UNIVERSITÄT PADERBORN

Software Engineering

Landscape depending parameter tuning for search-based software testing

Overview

1. Introduction
2. Fundamentals
3. Experimental
4. Adaptive parameter control
5. Evaluation
6. Conclusion
7. Conclusion

Introduction

- Unit tests
- maximize coverage (line, branch, exception)
- lack of sufficient tests
- costly and time-consuming
- => use search-based software testing

Motivation

- Tools... => EvoSuite state-of-the-art
- may not terminate => search budget
- optimal only with optimal configuration
- No Free Lunch theorem
  - impossible to find optimal configuration for all problems
- EvoSuite's default configuration is fairly good, but not perfect

Research goal

- wide variety of problem-cases
- concept landscape depending
- adaptive parameter control
- increase coverage of EvoSuite

State-of-the-art

Challenges

-

Delimitation

-

1. Introduction

2. Fundamentals

3. Experimental

4. Adaptive parameter control

5. Evaluation

6. Conclusion

7. Conclusion

Search-based software testing

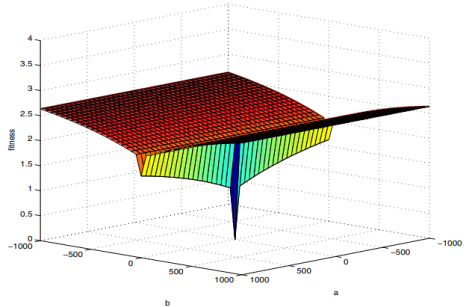
- tests for object oriented languages
- sequence of method calls
-
-
-

Fitness function

-
- function for e.g. coverage
- guidance for search algorithms
-
-

```
1 void bar(int x) {
2     if (x == 1) {
3         // uncovered code
4     }
5 }
```

Fitness landscape



Genetic algorithm

Heading

- start with random population
- iterate till termination condition
- return last generation

Heading

-
-
-

DynaMOSA

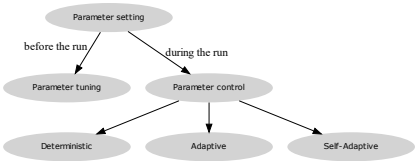
Heading

- start with random population
- multiple target
- keep track of target covering individuals

Heading

- iterate till termination condition
- update targets
- update archive
- select by rank
- return archive as last generation

Parameter tuning and control



1. Introduction

2. Fundamentals

3. Experimental

4. Adaptive parameter control

5. Evaluation

6. Conclusion

7. Conclusion

Corpus

SF110

- 110 open-source Java projects
- 23,894 Java Classes
-
-
-

Panichella et al.

- 117 open-source Java projects
- 346 Java Classes
- non-trivial and complex
- often used

Prediction sample

- S_1
- 709 Java Classes
- randomly selected
- SF110

- 9.8 days on three machines
-
-
-

Evaluation sample

- S_2
- 346 Java Classes
- the whole Panichella corpus
-

- 4.8 days on tree machines
-
-
-

Sensitive sample

- S_3
- 20 Java Classes
- extracted from S_1
- high Standard Deviation

- 6.6 hours on tree machines
-
-
-

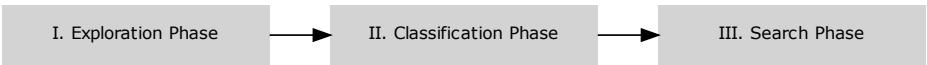
Comparisons

- 30 repeats for every Java Class
- Mann-Whitney U-test
- Vargha-Delaney effect size $\hat{A}_{12}$
-

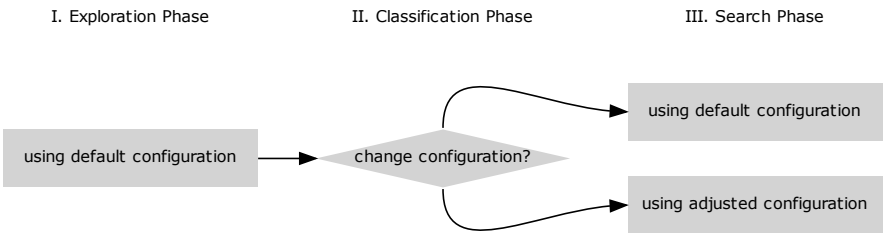
1. Introduction
2. Fundamentals
3. Experimental
4. Adaptive parameter control
5. Evaluation
6. Conclusion
7. Conclusion

Concept

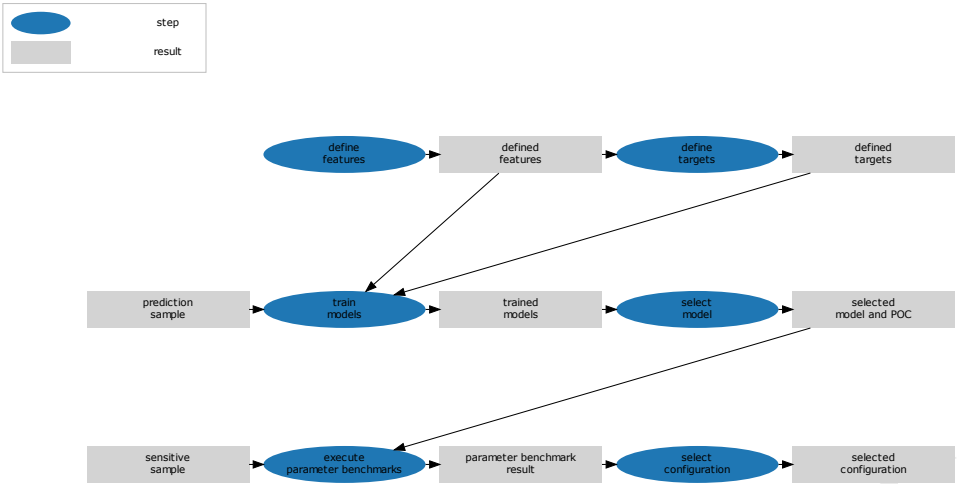
- use the beginning of the search to explore the search space
- classify the current search (”adjust” or ”default”)
- use configuration depending on class



Configuration selection



Components


step
result
define features
defined features
define targets
defined targets
prediction sample
train models
trained models
select model
selected model and POC
sensitive sample
execute parameter benchmarks
parameter benchmark result
select configuration
selected configuration

Landscape features

- Fitness and gradient
- Neutrality
- Neutrality Volume
- Information Content

Target approximation

-
-
-

Targets

- Low end coverage
- High standard deviation
- Relative low coverage

Classification

Heading

-
-
-

Heading

-
-
-

Parameter selection

Heading

-
-
-

Heading

-
-
-

1. Introduction

2. Fundamentals

3. Experimental

4. Adaptive parameter control

5. Evaluation

6. Conclusion

7. Conclusion

1. Introduction

2. Fundamentals

3. Experimental

4. Adaptive parameter control

5. Evaluation

6. Conclusion

7. Conclusion


UNIVERSITÄT
PADERBORN