www.ssbse.org/2011

immediately after FSE

and the first day is free

why not stay?
SBSE: Optimising Automation

Mark Harman
University College London
CREST Centre

Joint work with Yue Jia, Kiran Lakhotia, Bill Langdon, Spiros Mancoridis, Phil McMinn, Joachim Wegener, Shin Yoo, Yuanyuan Zhang ... and many others

Tuesday, 6 September 11
SBSE: Optimising Automation

Mark Harman
University College London
CREST Centre

Joint work with Yue Jia, Kiran Lakhota, Bill Langdon, Spiros Mancoridis, Phil McMinn, Joachim Wegener, Shin Yoo, Yuanyuan Zhang ... and many others

Tuesday, 6 September 11
Centre for Research on Evolution Search and Testing

1 admin
4 faculty
8 post docs
11 PhD students
1-4 resident visiting scholars
Research
Funding

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Three Repositories
Dr. Yuanyuan Zhang
SBSE Repository

This page collects the work which address the software engineering problems using metaheuristic search optimisation techniques (i.e., Genetic Algorithms) into the Repository of Publications on Search Based Software Engineering.

- SBSE repository is maintained by Yuanyuan Zhang
- 936 relevant publications are included
- Last updated on 23 July 2011

Who’s Who

- The number of publications in the year from 1976 to 2011.
- The ratio of SE research fields that involved SBSE.
- The ratio of publications number in the world countries.

Mark Harman, UCL CREST centre
SBSE Repository

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
how to tell you have the new version?
how to tell you have the new version?

ucl
not
kcl
how to tell you have the new version?

ucl
not kcl

it's the one with a who's who
The Genetic Programming Bibliography

The bibliography is part of the Collection of Computer Science Bibliographies. It is maintained and managed by William Langdon, Steven Gustafson, and John Koza.

Add single entries or email William Langdon or firstname D0T lastname AT-SYMBOl research D0T ge D0T com

Bibtex file
gp-bibliography.bib (Compressed) Over 6000 GP references

Other formats
refer format (Compressed)
text file (Compressed)

Search Interfaces
The GP bibliography is one of the many online computer science bibliographies. There are several online search tools for these bibliographies. For example:
- WWW form based search
- The coauthor graphs can also be used for searching by author name.
- You can find all entries for an author using a URL like this one: http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/WilliamBLangdon.html
  Naturally you must replace WilliamBLangdon.html by the author's name.

Other Resources
- Bibliographies of evolutionary computation conferences.
- Hints on using bibliographies with PC based word processing packages
- Links to other information on bibliographies.
- Subdirectory containing some tools and templates for maintaining bibliographies.
- Links to other bibliographies.
- Home pages of some GP researchers
Mutation Testing Repository

Welcome to Mutation Testing repository

Mutation Testing is a fault-based software testing technique that has been widely studied for over three decades. The literature on mutation testing has contributed a set of approaches, tools and empirical studies. This repository aims to provide a full coverage of the publications in the literature on Mutation Testing.

Mutation Testing Survey

Using this repository, a comprehensive analysis and survey of Mutation Testing work has been conducted. The paper presents the results of several development trend analyses. These analyses provide evidence that mutation testing techniques and tools are reaching a state of maturity and applicability, while the topic of mutation testing itself is the subject of increasing interest. This paper is currently published as a technical report, available from here. If you want to cite results from the survey, here is the BibTeX entry:

```latex
@ARTICLE{Jia2010b, 
  author = {Yue Jia and Mark Harman}, 
  title = {A Survey of the Development of Mutation Testing}, 
  journal = {IEEE Transactions of Software Engineering}, 
  year = {1999}, 
  volume = {To appear}, 
  number = {1}, 
  pages = {1} 
}
```

Terms of Use:

Unless stated differently, all material on this repository is the property of its own author. All downloads from this website are provided as freeware for personal use only. You must not sell, redistribute or modify any items (in whole, and/or in part) that have originated from this site.

Copyright © 2007 - 2010 Yue Jia and Mark Harman all rights reserved.

Mutation Testing remains an active research area with growing interest...

- READ MORE
COWs

CREST Open Workshop
Roughly one per month

Discussion based
Recorded and archived
COWs

CREST Open Workshop

Roughly one per month

Discussion based

Recorded and archived
COWs

CREST Open Workshop

Roughly one per month

Discussion based

Recorded and archived
CREST Open Workshop
Roughly one per month

Discussion based
Recorded and archived
COWs

CREST Open Workshop
Roughly one per month

Discussion based

Recorded and archived
COW Attendance

> 220 different researchers and practitioners
> 100 different organisations
> 20 countries
Previous 14 COWs

Nov 2009: 35 attendees: Search Based Software Engineering
Dec 2009: 21 attendees: Software Testing
Jan 2010: 34 attendees: Using Static Analysis for Fault Prediction
Feb 2010: 31 attendees: Operational Research for Software Engineering Methods
Mar 2010: 26 attendees: Information Theory for Search Based Software Engineering
Apr 2010: 24 attendees: Dependence Analysis and Slicing for Programs and Models
May 2010: 24 attendees: Information Flow and Security
Oct 2010: 34 attendees: Mutation testing
Nov 2010: 36 attendees: Code Provenance and Clone Detection
Jan 2011: 54 attendees: Program Slicing and Dependence
Feb 2011: 34 attendees: SBSE for Early Lifecycle Software Engineering
Apr 2011: 25 attendees: Security and Code
May 2011: 51 attendees: SBSE (with focus on Testing)
Jul 2011: 38 attendees: Genetic Programming for Software Engineering
Previous 14 COWs

Nov 2009: 35 attendees: Search Based Software Engineering
Dec 2009: 21 attendees: Software Testing
Jan 2010: 34 attendees: Using Static Analysis for Fault Prediction
Feb 2010: 31 attendees: Operational Research for Software Engineering Methods
Mar 2010: 26 attendees: Information Theory for Search Based software Engineering
Apr 2010: 24 attendees: Dependence Analysis and Slicing for Programs and Models
May 2010: 24 attendees: Information Flow and Security
Oct 2010: 34 attendees: Mootation testing
Nov 2010: 36 attendees: Code Provenance and Clone Detection
Jan 2011: 54 attendees: Program Slicing and Dependence
Feb 2011: 34 attendees: SBSE for Early Lifecycle Software Engineering
Apr 2011: 25 attendees: Security and Code
May 2011: 51 attendees: SBSE (with focus on Testing)
Jul 2011: 38 attendees: Genetic Programming for Software Engineering

typically 25 .. 50 people
Previous 14 COWs

Nov 2009: 35 attendees: Search Based Software Engineering
Dec 2009: 21 attendees: Software Testing
Jan 2010: 34 attendees: Using Static Analysis for Fault Prediction
Feb 2010: 31 attendees: Operational Research for Software Engineering Methods
Mar 2010: 26 attendees: Information Theory for Search Based Software Engineering
Apr 2010: 24 attendees: Dependence Analysis and Slicing for Programs and Models
May 2010: 24 attendees: Information Flow and Security
Oct 2010: 34 attendees: Mutation testing
Nov 2010: 36 attendees: Code Provenance and Clone Detection
Jan 2011: 54 attendees: Program Slicing and Dependence
Feb 2011: 34 attendees: SBSE for Early Lifecycle Software Engineering
Apr 2011: 25 attendees: Security and Code
May 2011: 51 attendees: SBSE (with focus on Testing)
Jul 2011: 38 attendees: Genetic Programming for Software Engineering

lots of SBSE

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Previous 14 COWs

Nov 2009: 35 attendees: Search Based Software Engineering
Dec 2009: 21 attendees: Software Testing
Jan 2010: 34 attendees: Using Static Analysis for Fault Prediction
Feb 2010: 31 attendees: Operational Research for Software Engineering Methods
Mar 2010: 26 attendees: Information Theory for Search Based Software Engineering
Apr 2010: 24 attendees: Dependence Analysis and Slicing for Programs and Models
May 2010: 24 attendees: Information Flow and Security
Oct 2010: 34 attendees: Mootation testing
Nov 2010: 36 attendees: Code Provenance and Clone Detection
Jan 2011: 54 attendees: Program Slicing and Dependence
Feb 2011: 34 attendees: SBSE for Early Lifecycle Software Engineering
Apr 2011: 25 attendees: Security and Code
May 2011: 51 attendees: SBSE (with focus on Testing)
Jul 2011: 38 attendees: Genetic Programming for Software Engineering

lots of testing too

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Previous 14 COWs

Nov 2009: 35 attendees: Search Based Software Engineering
Dec 2009: 21 attendees: Software Testing
Jan 2010: 34 attendees: Using Static Analysis for Fault Prediction
Feb 2010: 31 attendees: Operational Research for Software Engineering Methods
Mar 2010: 26 attendees: Information Theory for Search Based software Engineering
Apr 2010: 24 attendees: Dependence Analysis and Slicing for Programs and Models
May 2010: 24 attendees: Information Flow and Security
Oct 2010: 34 attendees: Mootation testing
Nov 2010: 36 attendees: Code Provenance and Clone Detection
Jan 2011: 54 attendees: Program Slicing and Dependence
Feb 2011: 34 attendees: SBSE for Early Lifecycle Software Engineering
Apr 2011: 25 attendees: Security and Code
May 2011: 51 attendees: SBSE (with focus on Testing)
Jul 2011: 38 attendees: Genetic Programming for Software Engineering

... and source code analysis
Previous 14 COWs

<table>
<thead>
<tr>
<th>Date</th>
<th>Attendees</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nov 2009</td>
<td>35</td>
<td>Search Based Software Engineering</td>
</tr>
<tr>
<td>Dec 2009</td>
<td>21</td>
<td>Software Testing</td>
</tr>
<tr>
<td>Jan 2010</td>
<td>34</td>
<td>Using Static Analysis for Fault Prediction</td>
</tr>
<tr>
<td>Feb 2010</td>
<td>31</td>
<td>Operational Research for Software Engineering Methods</td>
</tr>
<tr>
<td>Mar 2010</td>
<td>26</td>
<td>Information Theory for Search Based Software Engineering</td>
</tr>
<tr>
<td>Apr 2010</td>
<td>24</td>
<td>Dependence Analysis and Slicing for Programs and Models</td>
</tr>
<tr>
<td>May 2010</td>
<td>24</td>
<td>Information Flow and Security</td>
</tr>
<tr>
<td>Oct 2010</td>
<td>34</td>
<td>Mootation testing</td>
</tr>
<tr>
<td>Nov 2010</td>
<td>36</td>
<td>Code Provenance and Clone Detection</td>
</tr>
<tr>
<td>Jan 2011</td>
<td>54</td>
<td>Program Slicing and Dependence</td>
</tr>
<tr>
<td>Feb 2011</td>
<td>34</td>
<td>SBSE for Early Lifecycle Software Engineering</td>
</tr>
<tr>
<td>Apr 2011</td>
<td>25</td>
<td>Security and Code</td>
</tr>
<tr>
<td>May 2011</td>
<td>51</td>
<td>SBSE (with focus on Testing)</td>
</tr>
<tr>
<td>Jul 2011</td>
<td>38</td>
<td>Genetic Programming for Software Engineering</td>
</tr>
</tbody>
</table>

a little security

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Upcoming COWs

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Upcoming COWs

places limited to 34 ...

Oct 24th to 25th 2011: Predictive Models and SBSE
Nov 28th 2011: Code provenance and clone detection
Jan 31st to Feb 1st 2012: Testing and Verification
Upcoming COWs

places limited to 34 ...

Oct 24th to 25th 2011: Predictive Models and SBSE
Nov 28th 2011: Code provenance and clone detection
Jan 31st to Feb 1st 2012: Testing and Verification
Upcoming COWs

places limited to 34 ...

Oct 24th to 25th 2011: Predictive Models and SBSE

Nov 28th 2011: Code provenance and clone detection

Jan 31st to Feb 1st 2012: Testing and Verification
Upcoming COWs

places limited to 34 ...

Oct 24th to 25th 2011: Predictive Models and SBSE
Nov 28th 2011: Code provenance and clone detection
Jan 31st to Feb 1st 2012: Testing and Verification
Outline

Philosophical Basis: Science and Engineering

What is SBSE?

Is it any good?

Some algorithms

Single Objective SBSE

Example: Temporal Testing

Multiple Objective SBSE

Examples: Requirements, Design, Regression Testing
Outline

Philosophical Basis: Science and Engineering

What is SBSE?

Is it any good?

Some algorithms

Single Objective SBSE

Example: Temporal Testing

Multiple Objective SBSE

Examples: Requirements, Design, Regression Testing
Outline

Philosophical Basis: Science and Engineering

What is SBSE?

Is it any good?

Some algorithms

Single Objective SBSE

Example: Temporal Testing

Multiple Objective SBSE

Examples: Requirements, Design, Regression Testing
Outline

Philosophical Basis: Science and Engineering

What is SBSE?

Is it any good?

Some algorithms

Single Objective SBSE

Example: Temporal Testing

Multiple Objective SBSE

Examples: Requirements, Design, Regression Testing
Outline

Philosophical Basis: Science and Engineering

What is SBSE?

Is it any good?

Some algorithms

**Single Objective** SBSE

Example: Temporal Testing

**Multiple Objective** SBSE

Examples: Requirements, Design, Regression Testing

Mark Harman, UCL CREST centre
Outline

Philosophical Basis: Science and Engineering
What is SBSE?
Is it any good?
Some algorithms
Single Objective SBSE
Example: Temporal Testing
Multiple Objective SBSE
Examples: Requirements, Design, Regression Testing
Outline

Philosophical Basis: Science and Engineering

What is SBSE?

Is it any good?

Some algorithms

Single Objective SBSE

Example: Temporal Testing

Multiple Objective SBSE

Examples: Requirements, Design, Regression Testing

Mature

Well studied

Applied in Practice

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
... not covered (but important)
... not covered (but important)

Combinatorial Testing
Stress Testing, QoS Testing, ...
Hybrids: DSE/SBST, Static Analysis and SBST
Many Recent Surveys


Many Recent Surveys


Many Recent Surveys


Many Recent Surveys


Many Recent Surveys


Scientists’ and Engineers’ viewpoints
Scientists’ and Engineers’ viewpoints
Scientists’ and Engineers’ viewpoints

scientist:
Scientists’ and Engineers’ viewpoints

scientist:
Scientists’ and Engineers’ viewpoints

scientist:

what is true
Scientists’ and Engineers’ viewpoints

scientist:

what is true

correctness
Scientists’ and Engineers’ viewpoints

scientist:
what is true
correctness
model the world
Scientists’ and Engineers’ viewpoints

scientist:

what is true

correctness

model the world
to understand
Scientists’ and Engineers’ viewpoints

scientist: what is true correctness model the world to understand

engineer:
Scientists’ and Engineers’ viewpoints

scientist:

what is true
correctness
model the world
to understand

engineer:

what is possible
<table>
<thead>
<tr>
<th>scientist:</th>
<th>engineer:</th>
</tr>
</thead>
<tbody>
<tr>
<td>what is true</td>
<td>what is possible</td>
</tr>
<tr>
<td>correctness</td>
<td>within tolerance</td>
</tr>
<tr>
<td>model the world</td>
<td></td>
</tr>
<tr>
<td>to understand</td>
<td></td>
</tr>
</tbody>
</table>
Scientists’ and Engineers’ viewpoints

scientist:
what is true
correctness
model the world
to understand

engineer:
what is possible
within tolerance
model the world
Scientists’ and Engineers’ viewpoints

scientist:  
what is true  
correctness  
model the world  
to understand  

engineer:  
what is possible  
within tolerance  
model the world  
to manipulate  

Mark Harman, UCL CREST centre
Scientists’ and Engineers’ viewpoints

computer scientist:
what is true about computation
prove correctness
make it perfect

software engineer:
what is possible with software
test for imperfection
find were to improve
Combining Science and Engineering

prove correctness
make it perfect
test for imperfection
find were to improve

Tuesday, 6 September 11
Combining Science and Engineering

prove correctness
make it perfect

test for imperfection
find where to improve
Combining Science and Engineering

prove correctness
make it perfect

where possible ...
... and where impossible ...

test for imperfection
find were to improve
Combining Science and Engineering

prove correctness
make it perfect

where possible ...
... and where impossible ...

test for imperfection
find where to improve
Engineering words

- optimise
- with acceptable bounds
- improve performance
- optimize
- reduce cost
- fit for purpose
- within constraints
- tolerance
Engineering words

- tolerance
- optimise
- optimize
- reduce cost
- fit for purpose
- with acceptable bounds
- improve performance
- within constraints

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Tuesday, 6 September 11
Engineering words

- optimise with acceptable bounds
- tolerance
- optimise
- reduce cost
- improve performance
- optimize
- fit for purpose
- optimize within constraints
Engineering words

- optimise
- with acceptable bounds
- improve performance
- optimise
- reduce cost
- optimize
- fit for purpose
- within constraints

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Engineering words

optimise with acceptable bounds

reduce cost

fit for purpose

within constraints

improve performance

tolerance
Engineering words

- optimise
- with acceptable bounds
- improve performance
- optimise
- optimize
- reduce cost
- fit for purpose
- within constraints
- tolerance

Mark Harman, UCL CREST centre

SBSE: Optimising Automation

Tuesday, 6 September 11
Engineering words

- optimise
- with acceptable bounds
- optimize
- improve performance
- reduce cost
- fit for purpose
- tolerate
- within constraints

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Engineering words

- optimise
- optimize
- with acceptable bounds
- improve performance
- reduce cost
- fit for purpose
- within constraints
- tolerance

Mark Harman, UCL CREST centre

SBSE: Optimising Automation

Tuesday, 6 September 11
Engineering words

- optimise
- with acceptable bounds
- tolerance
- improve performance
- reduce cost
- fit for purpose
- optimize
- within constraints

Tuesday, 6 September 11
Engineering words

optimise

reduce cost
fit for purpose
within constraints
improve performance
tolerance
with acceptable bounds
optimisation: so good they named it twice
Engineering words

- optimise
- optimize
- with acceptable bounds
- improve performance
- tolerance
- reduce cost
- fit for purpose
- within constraints
- optimisation: so good they named it twice
- First in English
Engineering words

optimise

optimisation: so good they named it twice

First in English ...
Engineering words

- optimise
- optimize
- with acceptable bounds
- improve performance
- tolerance
- reduce cost
- fit for purpose
- within constraints
- optimization: so good they named it twice

First in English ... Then in American

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
overheard at FSE ...
overheard at FSE ...

fix worst bug in least time
find quickest test suite that achieves coverage
best interface that meets constraints
order requirements to maximise early delivery
need a version that minimizes heat dissipation
increase throughput without hurting response time
overheard at FSE ...

fix worst bug in least time
find quickest test suite that achieves coverage
best interface that meets constraints
order requirements to maximise early delivery
need a version that minimizes heat dissipation
increase throughput without hurting response time
overheard at FSE ...

fix worst bug in least time
find quickest test suite that achieves coverage
best interface that meets constraints
order requirements to maximise early delivery
need a version that minimizes heat dissipation
increase throughput without hurting response time
overheard at FSE ...

- fix worst bug in least time
- find quickest test suite that achieves coverage
- best interface that meets constraints
- order requirements to maximise early delivery
- need a version that minimizes heat dissipation
- increase throughput without hurting response time
overheard at FSE ...

- fix worst bug in least time
- find quickest test suite that achieves coverage
- best interface that meets constraints
- order requirements to maximise early delivery
- need a version that minimizes heat dissipation
- increase throughput without hurting response time
overheard at FSE ...

fix worst bug in least time
find quickest test suite that achieves coverage
best interface that meets constraints
order requirements to maximise early delivery
need a version that minimizes heat dissipation
increase throughput without hurting response time
overheard at FSE ...

fix worst bug in least time
find quickest test suite that achieves coverage
best interface that meets constraints
order requirements to maximise early delivery
need a version that minimizes heat dissipation
increase throughput without hurting response time
What is SBSE

In SBSE we apply search techniques to search large search spaces, guided by a fitness function that captures properties of the acceptable software artefacts we seek.
What is SBSE

In SBSE we apply search techniques to search large search spaces, guided by a fitness function that captures properties of the acceptable software artefacts we seek.
What is SBSE

In SBSE we apply search techniques to search large search spaces, guided by a fitness function that captures properties of the acceptable software artefacts we seek.

like google search?
like code search?
like breadth first search?
What is SBSE

In SBSE we apply search techniques to search large search spaces, guided by a fitness function that captures properties of the acceptable software artefacts we seek.
What is SBSE

In SBSE we apply **search techniques** to search large search spaces, **guided by a fitness** function that captures properties of the acceptable software artefacts we seek.

- Tabu Search
- Ant Colonies
- Hill Climbing
- Simulated Annealing
- Particle Swarm Optimization
- Genetic Algorithms
- Genetic Programming
- Greedy
- LP
- Random
- Estimation of Distribution Algorithms
Why?
Eight Queens Problem
Eight Queens Problem
Eight Queens Problem
Eight Queens Problem

Perfect
Eight Queens Problem

Perfect Score 0
Eight Queens Problem
Eight Queens Problem
Eight Queens Problem
Eight Queens Problem
Eight Queens Problem

Two attacks
Eight Queens Problem

Two attacks

Score -2
Eight Queens Problem
Eight Queens Problem
Eight Queens Problem
Eight Queens Problem
Eight Queens Problem

Three attacks
Eight Queens Problem

Three attacks

Score -3
That was easy
Eight Queens Problem

Place 8 queens on the board

So that there are no attacks
Eight Queens Problem

Place 44 queens on the board
So that there are no attacks
Eight Queens Problem

Place 10 queens on the board so that there are no attacks.
Checking vs Generating

Task One:
Write a method to determine which is the better of two placements of N queens

Task Two:
Write a method to construct a board placement with N non-attacking queens
Checking vs Generating

Task One:
Write a method to determine which is the better of two placements of N queens

Task Two:
Write a method to construct a board placement with N non attacking queens
Checking vs Generating

Search Based Software Engineering
Write a method to determine which is the better of two solutions

Conventional Software Engineering
Write a method to construct a perfect solution
Checking vs Generating

**Search Based Software Engineering**
Write a method to determine which is the better of two solutions

**Conventional Software Engineering**
Write a method to construct a perfect solution
Checking vs Generating

Search Based Software Engineering
Write a method to determine which is the better of two solutions

Conventional Software Engineering
Write a method to construct a perfect solution
Checking vs Generating

Search Based Software Engineering
Write a **fitness function** to determine which is the better of two solutions

Conventional Software Engineering
Write a method to construct a perfect solution
Checking vs Generating

Search Based Software Engineering
Write a **fitness function** to guide a search

Conventional Software Engineering
Write a method to construct a perfect solution
Checking vs Generating

Search Based Software Engineering
Write a **fitness function** to guide *automated* search

Conventional Software Engineering
Write a method to construct a perfect solution
What is SBSE

Search Based Optimization

Software Engineering

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
What is SBSE

Search Based Optimization

Software Engineering

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
What is SBSE

Search Based Optimization

Software Engineering

Mark Harman, UCL CREST centre
What is SBSE

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Search Based Optimization
Software Engineering
example

single simple objective

Optimising Temporal Testing
Temporal testing

Verifying timing constraints of real-time systems by means of evolutionary testing

Evolutionary Testing
Evolutionary Testing
Evolutionary Testing
Evolutionary Testing
Evolutionary Testing

Fitness evaluation
Evolutionary Testing
Evolutionary Testing

Fitness evaluation

End?

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Evolutionary Testing

Fitness evaluation

Selection

End?

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Evolutionary Testing

Fitness evaluation

Selection

End?

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Evolutionary Testing

- Fitness evaluation
- Selection
- Recombination
- End?

Tuesday, 6 September 11

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Evolutionary Testing

- Fitness evaluation
- Selection
- Recombination

End?
Evolutionary Testing

- Mutation
- Fitness evaluation
- Recombination
- Selection
- End?
Evolutionary Testing

- Mutation
- Fitness evaluation
- Recombination
- Selection
- End?
Evolutionary Testing
Evolutionary Testing

- Insertion
- Test cases
- Execution
- Fitness evaluation
- Monitoring
- Recombination
- Mutation
- Selection
- End?
Daimler Temporal Testing

optimal point of time for triggering the airbag igniter

deceleration

t

t_{\text{min}}

t_{\text{max}}
Daimler Temporal Testing

- Deceleration
- $t_{\text{min}}$ to $t_{\text{max}}$
- Optimal point of time for triggering the airbag igniter

Insertion
Mutation
Recombination
Fitness evaluation
Test cases
Execution
Monitoring
End?
Selection

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Daimler Temporal Testing

Deceleration

optimal point of time for triggering the airbag igniter

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Daimler Temporal Testing

Deceleration

t_min

t_max

Optimal point of time for triggering the airbag igniter

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Evolution vs Random for Temporal Testing

Evolutionary Test

Random Test

Execution Time (in cycles)

Generation

Frequency

Execution Time (cycles)
The advantages of SBSE
The advantages of SBSE
The advantages of SBSE

- Insight-rich
- Scalable
- Robust
- Generic
- Realistic

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
The advantages of SBSE

- Insight-rich
- Scalable
- Robust
- Generic
- Realistic

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
The advantages of SBSE

- Insight-rich
- Scalable
- Robust
- Generic
- Realistic

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
The advantages of SBSE

- Insight-rich
- Scalable
- Robust
- Generic
- Realistic

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
The advantages of SBSE

- Insight-rich
- Scalable
- Robust
- Generic
- Realistic

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
The advantages of SBSE

- Insight-rich
- Scalable
- Robust
- Generic
- Realistic

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Evolutionary Algorithms

about 70% of all SBSE uses evolutionary computation
some other
search based
optimisation algorithms
Hill Climbing

pick a random point
evaluate fitness of near neighbours
move to a fitter neighbour
stop when no neighbour is fitter
Hill Climbing

pick a random point

evaluate fitness of near neighbours

move to a fitter neighbour

stop when no neighbour is fitter
Hill Climbing

pick a random point
evaluate fitness of near neighbours
move to a fitter neighbour
stop when no neighbour is fitter
Hill Climbing

pick a random point

evaluate fitness of near neighbours

move to a fitter neighbour

stop when no neighbour is fitter
Hill Climbing

- pick a random point
- evaluate fitness of near neighbours
- move to a fitter neighbour
- stop when no neighbour is fitter
Hill Climbing

pick a random point
evaluate fitness of near neighbours
move to a fitter neighbour
stop when no neighbour is fitter
Hill Climbing

pick a random point
evaluate fitness of near neighbours
move to a fitter neighbour
stop when no neighbour is fitter
Hill Climbing

pick a random point
evaluate fitness of near neighbours
move to a fitter neighbour
stop when no neighbour is fitter
Hill Climbing

- pick a random point
- evaluate fitness of near neighbours
- move to a fitter neighbour
- stop when no neighbour is fitter
Hill Climbing

- pick a random point
- evaluate fitness of near neighbours
- move to a fitter neighbour
- stop when no neighbour is fitter

stuck at local optima
**Iterative Hill Climbing**

pick a random point

evaluate fitness of near neighbours

move to a fitter neighbour

stop when no neighbour is fitter

*restart*
Hill Climbing

pick a random point

evaluate fitness of near neighbours

move to a fitter neighbour

stop when no neighbour is fitter
Hill Climbing

pick a random point

evaluate fitness of near neighbours

maybe move to an unfit neighbour while warm

stop when no neighbour is fitter
Simulated Annealing

pick a random point

evaluate fitness of near neighbours

maybe move to a unfit neighbour while warm

don’t stop when no neighbour is fitter
Simulated Annealing

pick a random point

evaluate fitness of near neighbours

maybe move to an unfit neighbour while warm

stop when cold
Ingredients for an optimising search algorithm

Representation

Neighbourhood

Fitness Function
Search-Based Algorithms

**Representation**
encoding all possible solutions
you *must* have one, surely

**Neighbourhood**

**Fitness Function**
Search-Based Algorithms

Representation

Neighbourhood

Part of our understanding of the problem

it’s useful to know our near neighbours

Fitness Function
Search-Based Algorithms

Representation

Neighbourhood

**Fitness Function** indicates how ‘good’ an input is an *ordinal scale* metric
Essential Ingredients for SBSE
Essential Ingredients for SBSE Representation
Essential Ingredients for SBSE

Representation

Fitness Function
Essential Ingredients for SBSE

Representation

Fitness Function

erh ... that’s it!
yet more search algorithms
yet more search algorithms

- Tabu Search
- Particle Swarm Optimisation
- Ant Colony Optimisation
- Genetic Programming
- Estimation of Distribution Algorithms
yet more search algorithms

- Tabu Search
- Particle Swarm Optimisation
- Ant Colony Optimisation
- Genetic Programming
- Estimation of Distribution Algorithms

SBSE: Optimising Automation  Mark Harman, UCL CREST centre

Tuesday, 6 September 11
yet more search algorithms

Tabu Search

Particle Swarm Optimisation

Ant Colony Optimisation

Genetic Programming

Growth Trends
SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Author statistics

> 800 different authors
> 270 different institutions
> 40 countries

source: SBSE repository, 28th July 2011.
Just some of the many SBSE applications
Just some of the many SBSE applications

Agent Oriented
Aspect Oriented
Assertion Generation
Bug Fixing
Component Oriented
Design
Effort Estimation
Heap Optimisation
Model Checking
Predictive Modelling
Probe distribution
Program Analysis
Program Comprehension
Program Transformation
Project Management
Protocol Optimisation
QoS
Refactoring
Regression Testing
Requirements
Reverse Engineering
SOA
Software Maintenance and Evolution
Test Generation
UIO generation
Emergent Areas in SBSE
Emergent Areas in SBSE

- Year 2001:
  - Testing and Debugging: 13%
  - Distribution, Maintenance, and Enhancement: 6%
  - Management: 5%
  - Design Tools and Techniques: 5%
  - Requirements/Specifications: 4%
  - Software/Program Verification: 4%
  - General Aspects: 2%
  - Other: 5%

- Year 2004:
  - Testing and Debugging: 6%
  - Distribution, Maintenance, and Enhancement: 11%
  - Management: 6%
  - Design Tools and Techniques: 5%
  - Requirements/Specifications: 4%
  - Software/Program Verification: 4%
  - General Aspects: 5%
  - Other: 1%

- Year 2008:
  - Testing and Debugging: 10%
  - Distribution, Maintenance, and Enhancement: 9%
  - Management: 57%
  - General Aspects: 3%
  - Other: 11%

Mark Harman, UCL CREST centre

SBSE: Optimising Automation
Emergent Areas in SBSE

- Year 2001
  - Testing and Debugging: 5%
  - Distribution, Maintenance, and Enhancement: 13%
  - Management: 6%
  - 64%

- Year 2004
  - Design Tools and Techniques: 11%
  - Requirements/Specifications: 4%
  - Software/Program Verification: 5%
  - 57%

- Year 2008
  - General Aspects: 5%
  - Other: 11%
  - 54%

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Emergent Areas in SBSE

Year 2001

- Testing and Debugging: 13%
- Distribution, Maintenance, and Enhancement: 5%
- Management: 6%
- Design Tools and Techniques: 64%
- Requirements/Specifications: 11%
- Software/Program Verification: 11%

Year 2004

- Testing and Debugging: 5%
- Distribution, Maintenance, and Enhancement: 5%
- Management: 6%
- Design Tools and Techniques: 57%
- Requirements/Specifications: 9%
- Software/Program Verification: 11%

Year 2008

- Testing and Debugging: 3%
- Distribution, Maintenance, and Enhancement: 5%
- Management: 4%
- Design Tools and Techniques: 10%
- Requirements/Specifications: 4%
- Software/Program Verification: 9%
- General Aspects: 11%
- Other: 54%

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Emergent Areas in SBSE
SBSE challenge

pick a Software Engineering problem

see if I can “SBSE” it ...
why is SBSE so widely used?
Software Engineering Problems
Capture requirements  Generate tests  Explore designs  Maintain evolve  Regression test

Minimize

Maximize

SBSE: Optimising Automation  Mark Harman, UCL CREST centre
Generate tests

Minimize
- Development time

Maximize
- Satisfaction
- Fairness

Number of test

Capture requirements
Explore designs
Maintain evolve
Regression test

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Number of test

Execution time

Maximize

Code coverage

Fairness

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
**SBSE: Optimising Automation**

**Mark Harman, UCL CREST centre**

**Tuesday, 6 September 11**

**Capture requirements**
**Generate tests**
**Explore designs**
**Maintain evolve**
**Regression test**

**Minimize**
- Number of test
- Execution time

**Maximize**
- Code coverage
- Fault coverage
Capture requirements
Generate tests
Explore designs
Maintain evolve
Regression test

Minimize

Coupling

Maximize

Cohesion

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Emergent paradigms
Emergent paradigms
Emergent paradigms

Cloud

Agile

Quantum

Ambient

Non functional properties

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Emergent paradigms

Objectives:
- leakage/sharing
- fairness
- response

Cloud
Emergent paradigms

Objectives:
- cost
- value
- refactoring
- split points

Agile

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Emergent paradigms

Objectives:
coverage
response
fit

Ambient

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Emergent paradigms

Objectives:
- heat dissipation
- WCET
- resource use

Non functional properties

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Emergent paradigms

Objectives:
- functionality
- entanglement
- interference

Quantum
Optimising Multiple Objectives
Multi Objective Search

Many problems have multiple objectives
Often we have many metrics
Recent SBSE work has been multi objective
Pareto optimality can yield insight
Multi Objective Search

Many problems have multiple objectives

Often we have many metrics

Recent SBSE work has been multi objective

Pareto optimality can yield insight
Multi Objective Search
Multi Objective Search

Fitness function A

Fitness function B
Multi Objective Search

Fitness function A

Fitness function B

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective Search

Fitness function A

Fitness function B
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front
Multi Objective Search

Fitness function A

Pareto Front

Fitness function B

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Multi Objective Search

Fitness function A

Fitness function B

Pareto Front
Direction of front growth

Fitness function A

Fitness function B

Pareto Front

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Direction of front growth

Fitness function A

Fitness function B

Pareto Front
Literature


SBSE for Requirements
Customers

Requirements

Cost

Customers’ value for each requirement

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
The Next Release Problem

\[ C = \{c_1, \ldots, c_j, \ldots, c_m\} \quad R = \{r_1, \ldots, r_i, \ldots, r_n\} \]

\[ \text{Weight} = \{w_1, \ldots, w_j, \ldots, w_m\} \quad \text{Cost} = \{\text{cost}_1, \ldots, \text{cost}_n\} \]

\[ \text{value}(r_i, c_j) \]

\[ \text{score}_i = \sum_{j=1}^{m} w_j \times \text{value}(r_i, c_j) \]
Motorola Cell Phone Requirements
Motorola Cell Phone Requirements

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Motorola Cell Phone Requirements
Motorola Cell Phone Requirements
Motorola Cell Phone Requirements
Motorola Cell Phone Requirements

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
The Next Release Problem. 

M S. Feather & T Menzies. 
Converging on the Optimal Attainment of Requirements 
RE 2002.

Y. Zhang M. Harman & A. Mansouri. 
The Multi-Objective Next Release Problem 
GECCO 2007.

M. O. Saliu & G. Ruhe. 
Bi-Objective Release Planning for Evolving Software Systems. 
FSE 2007.

A Search Based Approach to Fairness Analysis in Requirements Assignments to Aid 
Negotiation, Mediation & Decision Making. 

The Human Competitiveness of Search Based Software Engineering. 
SSBSE 2010.

An Ant Colony Optimization Approach to the Software Release Planning with Dependent Requirements. 
SSBSE 2011.
literature

The Next Release Problem.

M S. Feather & T Menzies.
Converging on the Optimal Attainment of Requirements
RE 2002.

Y. Zhang M. Harman & A. Mansouri.
The Multi-Objective Next Release Problem
GECCO 2007.

M. O. Saliu & G. Ruhe.
Bi-Objective Release Planning for Evolving Software Systems.
FSE 2007.

A Search Based Approach to Fairness Analysis in Requirements Assignments to Aid
Negotiation, Mediation & Decision Making.

The Human Competitiveness of Search Based Software Engineering.
SSBSE 2010.

An Ant Colony Optimization Approach to the Software Release Planning with Dependent Requirements.
SSBSE 2011.
The Next Release Problem.

M.S. Feather & T Menzies.
Converging on the Optimal Attainment of Requirements
RE 2002.

Y. Zhang, M. Harman & A. Mansouri.
The Multi-Objective Next Release Problem
GECCO 2007.

M. O. Saliu & G. Ruhe.
Bi-Objective Release Planning for Evolving Software Systems.
FSE 2007.

A Search Based Approach to Fairness Analysis in Requirements Assignments to Aid
Negotiation, Mediation & Decision Making.

The Human Competitiveness of Search Based Software Engineering.
SSBSE 2010.

An Ant Colony Optimization Approach to the Software Release Planning with Dependent Requirements.
SSBSE 2011.
literature

The Next Release Problem.

M S. Feather & T Menzies.
Converging on the Optimal Attainment of Requirements
RE 2002.

Y. Zhang M. Harman & A. Mansouri.
The Multi-Objective Next Release Problem
GECCO 2007.

M. O. Saliu & G. Ruhe.
Bi-Objective Release Planning for Evolving Software Systems.
FSE 2007.

A Search Based Approach to Fairness Analysis in Requirements Assignments to Aid Negotiation, Mediation & Decision Making.

The Human Competitiveness of Search Based Software Engineering.
SSBSE 2010.

An Ant Colony Optimization Approach to the Software Release Planning with Dependent Requirements.
SSBSE 2011.

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
literature

The Next Release Problem.

M.S. Feather & T Menzies.
Converging on the Optimal Attainment of Requirements
RE 2002.

Y. Zhang, M. Harman & A. Mansouri.
The Multi-Objective Next Release Problem
GECCO 2007.

M. O. Saliu & G. Ruhe.
Bi-Objective Release Planning for Evolving Software Systems.
FSE 2007.

A Search Based Approach to Fairness Analysis in Requirements Assignments to Aid
Negotiation, Mediation & Decision Making.

The Human Competitiveness of Search Based Software Engineering.
SSBSE 2010.

An Ant Colony Optimization Approach to the Software Release Planning with Dependent Requirements.
SSBSE 2011.
SBSE for Software Design

Mark Harman, UCL CREST centre
Familiar?
Bunch System

MQ(Good Partition) > MQ(Bad Partition)
Bunch System

Mancoridis et al.  Drexel
Literature


M. Barros Evaluating Modularization Quality as an Extra Objective in Multiobjective Software Module Clustering. SSBSE 2011
Literature

Using Automatic Clustering to Produce High-Level System Organizations of Source Code
IWPC 1998.

M. Harman, R. M. Hierons & M. Proctor.
A New Representation and Crossover Operator for Search-based Optimization of Software
Modularization.
GECCO 2002.

A Multiple Hill Climbing Approach to Software Module Clustering.
ICSM 2003

B. S. Mitchell & S. Mancoridis.
On the Automatic Modularization of Software Systems using the Bunch Tool.

K. Praditwong, M. Harman & X. Yao
Software Module Clustering as a Multi-Objective Search Problem.

M. Barros
Evaluating Modularization Quality as an Extra Objective in Multiobjective Software Module Clustering.
SSBSE 2011
Literature

Using Automatic Clustering to Produce High-Level System Organizations of Source Code
IWPC 1998.

M. Harman, R. M. Hierons & M. Proctor.
A New Representation and Crossover Operator for Search-based Optimization of Software
Modularization.
GECCO 2002.

A Multiple Hill Climbing Approach to Software Module Clustering.
ICSM 2003

B. S. Mitchell & S. Mancoridis.
On the Automatic Modularization of Software Systems using the Bunch Tool.

K. Praditwong, M. Harman & X. Yao
Software Module Clustering as a Multi-Objective Search Problem.

M. Barros
Evaluating Modularization Quality as an Extra Objective in Multiobjective Software Module Clustering.
SSBSE 2011
Literature

Using Automatic Clustering to Produce High-Level System Organizations of Source Code
IWPC 1998.

M. Harman, R. M. Hierons & M. Proctor.
A New Representation and Crossover Operator for Search-based Optimization of Software
Modularization.
GECCO 2002.

A Multiple Hill Climbing Approach to Software Module Clustering.
ICSM 2003

B. S. Mitchell & S. Mancoridis.
On the Automatic Modularization of Software Systems using the Bunch Tool.

K. Praditwong, M. Harman & X. Yao
Software Module Clustering as a Multi-Objective Search Problem.

M. Barros
Evaluating Modularization Quality as an Extra Objective in Multiobjective Software Module Clustering.
SSBSE 2011
Literature


M. Barros Evaluating Modularization Quality as an Extra Objective in Multiobjective Software Module Clustering. SSBSE 2011
SBSE for Regression Testing
Regression Testing
Regression Testing
Large Test Suites

Regression Testing

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Tuesday, 6 September 11
Selection

Large Test Suites

Limited Resource

Uses impact analysis to precisely identify the changed parts of the program and only test those parts.
Selection

Large Test Suites

Limited Resource

Uses impact analysis to precisely identify the changed parts of the program and only test those parts.

Often not the answer because:
- requires static analysis
- not enough reduction in effort

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Minimisation

Large Test Suites

Seeks to reduce the size of test suites while satisfying test adequacy goals

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Minimisation

Large Test Suites

Seeks to reduce the size of test suites while satisfying test adequacy goals
Minimisation

Large Test Suites

Seeks to reduce the size of test suites while satisfying test adequacy goals

R1

R2

R3

R4

T1

Mark Harman, UCL CREST centre
Minimisation

Large Test Suites

Seeks to reduce the size of test suites while satisfying test adequacy goals

R1 → T1
R2 → T1
R3 → T2
R4 → T2

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Minimisation

Large Test Suites

Seeks to reduce the size of test suites while satisfying test adequacy goals

R1

R2

R3

R4

T1

T2

T3

Mark Harman, UCL CREST centre
Minimisation

Large Test Suites

Seeks to reduce the size of test suites while satisfying test adequacy goals

R1
R2
R3
R4

T1
T2
T3

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Minimisation

Large Test Suites

Seeks to reduce the size of test suites while satisfying test adequacy goals
Prioritisation

Limited Resource

Seeks to achieve as much of adequacy measure as possible earlier on
Prioritisation

Limited Resource

Seeks to achieve as much of adequacy measure as possible earlier on

0 25.00 50.00 75.00 100.00
A-B-C-D-E

C-E-B-A-D

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
literature
Selection

Insights into regression testing
H.K.N. Leung and Lee White. ICSM 1989

An approach to regression testing using slicing
Rajiv Gupta, Mary Jean Harrold and Mary Lou Soffa. ICSM 1992

TestTube: A system for selective regression testing
Yih-Farn Chen, David Rosenblum and Kiem-Phong Vo. ICSE 1994

A safe, efficient regression test selection technique
Gregg Rothermel and Mary Jean Harrold. TOSEM 1997
Selection

Insights into regression testing
H.K.N. Leung and Lee White. ICSM 1989

An approach to regression testing using slicing
Rajiv Gupta, Mary Jean Harrold and Mary Lou Soffa. ICSM 1992

TestTube: A system for selective regression testing
Yih-Farn Chen, David Rosenblum and Kiem-Phong Vo. ICSE 1994

A safe, efficient regression test selection technique
Gregg Rothermel and Mary Jean Harrold. TOSEM 1997
Selection

Insights into regression testing
H.K.N. Leung and Lee White. ICSM 1989

An approach to regression testing using slicing
Rajiv Gupta, Mary Jean Harrold and Mary Lou Soffa. ICSM 1992

TestTube: A system for selective regression testing
Yih-Farn Chen, David Rosenblum and Kiem-Phong Vo. ICSE 1994

A safe, efficient regression test selection technique
Gregg Rothermel and Mary Jean Harrold. TOSEM 1997
Minimization

Effect of test set minimization on fault detection effectiveness
Eric Wong, Joseph Horgan, Saul London and Aditya Mathur.
ICSE 1995

An empirical study of the effects of minimization on the fault
detection capabilities of test suites
Gregg Rothermel, Mary Jean Harrold, Jeffery Ostrin and
Christie Hong. ICSM 1998

Test-suite reduction and prioritization for modified condition/
decision coverage
James A. Jones and Mary Jean Harrold. TSE 2003.
Minimization

Effect of test set minimization on fault detection effectiveness
Eric Wong, Joseph Horgan, Saul London and Aditya Mathur. ICSE 1995

An empirical study of the effects of minimization on the fault
detection capabilities of test suites
Gregg Rothermel, Mary Jean Harrold, Jeffery Ostrin and Christie Hong. ICSM 1998

Test-suite reduction and prioritization for modified condition/
decision coverage
James A. Jones and Mary Jean Harrold. TSE 2003.
Minimization

Effect of test set minimization on fault detection effectiveness
Eric Wong, Joseph Horgan, Saul London and Aditya Mathur. ICSE 1995

An empirical study of the effects of minimization on the fault
detection capabilities of test suites
Gregg Rothermel, Mary Jean Harrold, Jeffery Ostrin and Christie Hong. ICSM 1998

Test-suite reduction and prioritization for modified condition/
decision coverage
James A. Jones and Mary Jean Harrold. TSE 2003.
Prioritisation

Test case prioritization: An empirical study
Gregg Rothermel, Roland Untch, Chengyun Chu and Mary Jean Harrold. ICSM 1999.

Prioritizing test cases for regression testing
Gregg Rothermel, Roland Untch, Chengyun Chu and Mary Jean Harrold. TSE 2001.

Prioritizing test cases for regression testing

Test case prioritization: A family of empirical studies
Sebastian Elbaum, Alexey Malishevsky and Gregg Rothermel. TSE 2002.

Search algorithms for regression test case prioritization
Prioritisation

Test case prioritization: An empirical study
Gregg Rothermel, Roland Untch, Chengyun Chu and Mary Jean Harrold. ICSM 1999.

Prioritizing test cases for regression testing
Gregg Rothermel, Roland Untch, Chengyun Chu and Mary Jean Harrold. TSE 2001.

Prioritizing test cases for regression testing

Test case prioritization: A family of empirical studies
Sebastian Elbaum, Alexey Malishevsky and Gregg Rothermel. TSE 2002.

Search algorithms for regression test case prioritization
Prioritisation

Test case prioritization: An empirical study
Gregg Rothermel, Roland Untch, Chengyun Chu and Mary Jean Harrold. ICSM 1999.

Prioritizing test cases for regression testing
Gregg Rothermel, Roland Untch, Chengyun Chu and Mary Jean Harrold. TSE 2001.

Prioritizing test cases for regression testing

Test case prioritization: A family of empirical studies
Sebastian Elbaum, Alexey Malishevsky and Gregg Rothermel. TSE 2002.

Search algorithms for regression test case prioritization
Prioritisation

Test case prioritization: An empirical study
Gregg Rothermel, Roland Untch, Chengyun Chu and Mary Jean Harrold. ICSM 1999.

Prioritizing test cases for regression testing
Gregg Rothermel, Roland Untch, Chengyun Chu and Mary Jean Harrold. TSE 2001.

Prioritizing test cases for regression testing

Test case prioritization: A family of empirical studies
Sebastian Elbaum, Alexey Malishevsky and Gregg Rothermel. TSE 2002.

Search algorithms for regression test case prioritization
Growth Trends
Relative Interest in Different Subjects

Figure 3. Relative research interest in each subject. Papers that consider more than one subject were counted multiple times.

Taken from forthcoming STVR survey by Shin Yoo and Mark Harman
Potential Impact

Survey of 157 papers on regression testing techniques

Multi-Objective Regression Testing

“the minimised test suite is still too big?!”

“I care not just code coverage”
<table>
<thead>
<tr>
<th>Test Case</th>
<th>Program Blocks</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1  2  3  4  5  6  7  8  9  10</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>x   x   x   x   x   x   x   x</td>
<td>4</td>
</tr>
<tr>
<td>T2</td>
<td>x   x   x   x   x   x   x   x   x</td>
<td>5</td>
</tr>
<tr>
<td>T3</td>
<td>x   x   x   x   x   x   x   x</td>
<td>3</td>
</tr>
<tr>
<td>T4</td>
<td>x   x   x   x   x   x   x   x</td>
<td>3</td>
</tr>
</tbody>
</table>
### Test Case vs Program Blocks

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Program Blocks</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1   2   3   4   5   6   7   8   9   10</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>x  x  x  x  x  x     x</td>
<td>4</td>
</tr>
<tr>
<td>T2</td>
<td>x  x          x  x  x     x</td>
<td>5</td>
</tr>
<tr>
<td>T3</td>
<td>x  x          x  x</td>
<td>x</td>
</tr>
<tr>
<td>T4</td>
<td>x  x          x  x</td>
<td>x</td>
</tr>
</tbody>
</table>

### Single Objective

*SBSE: Optimising Automation*

Mark Harman, UCL CREST centre
<table>
<thead>
<tr>
<th>Test Case</th>
<th>Program Blocks</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1  2  3  4  5  6  7  8  9  10</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>x  x  x  x  x  x  x  x  x  x</td>
<td>4</td>
</tr>
<tr>
<td>T2</td>
<td>x  x  x  x  x  x  x  x  x  x</td>
<td>5</td>
</tr>
<tr>
<td>T3</td>
<td>x  x  x  x  x  x  x  x  x</td>
<td>3</td>
</tr>
<tr>
<td>T4</td>
<td>x  x  x  x  x  x  x  x</td>
<td>3</td>
</tr>
</tbody>
</table>

### Single Objective

Choose test case with highest block per time ratio as the next one.
## Single Objective

Choose test case with highest block per time ratio as the next one

1. T1 (ratio = 2.0)
2. T2 (ratio = 2 / 5 = 0.4)

∴ \{T1, T2\} (takes 9 hours)
Single Objective

Choose test case with highest block per time ratio as the next one

1) T1 (ratio = 2.0)
2) T2 (ratio = 2 / 5 = 0.4)

∴ {T1, T2} (takes 9 hours)

“But we only have 7 hours...?”
Single Objective

Choose test case with highest block per time ratio as the next one

1) T1 (ratio = 2.0)
2) T2 (ratio = 2 / 5 = 0.4)

∴ {T1, T2} (takes 9 hours)

“But we only have 7 hours...?”

---

Multi Objective

Choose test case with highest block per time ratio as the next one

1) T1 (ratio = 2.0)
2) T2 (ratio = 2 / 5 = 0.4)

∴ {T1, T2} (takes 9 hours)

“But we only have 7 hours...?”

---

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Trade-off

2 Objectives, schedule

2 Objectives, printtokens2

Code Coverage vs. Cost graphs for different methods:
- Reference
- Additional Greedy
- vNSGA-II
- NSGA-II

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
look ahead
Quick peek ahead

Tomorrow,
Wednesday 7th September
FSE Lecture room 1, 4pm slot: Industrial track 3 - Software Testing

Shin Yoo, Robert Nilsson and Mark Harman
Faster Fault Finding at Google using
Multi Objective Regression Test Optimisation
Google CLs (FSE 2011)
Another quick peek ... slightly further ahead

Monday 12th September at SSBSE, 4pm slot
Shin Yoo, Mark Harman and Shmuel Ur
*Highly Scalable Multi Objective Test Suite Minimisation Using Graphics Cards*
GPGPU scale up (SSBSE 2011)

![Speedup Chart](image)
Overviews and Survey Papers

Overviews and Survey Papers

Overviews and Survey Papers


Overviews and Survey Papers

Overviews and Survey Papers

Overviews and Survey Papers

Overviews and Survey Papers


Overviews and Survey Papers

Overviews and Survey Papers

Tutorial Paper


google: search based software engineering tutorial
Tutorial Paper


googole: search based software engineering tutorial
Tutorial Paper


google: search based software engineering tutorial
Tutorial Paper


google: search based software engineering tutorial
SBSE challenge

pick a Software Engineering problem
see if I can “SBSE” it ...
You gotta ask yourself...

How often does a tester have only one objective?
MORTO agenda

Cost
Value
Constraints
MORTO agenda

Mark Harman
Making the Case for MORTO: Multi Objective Regression Test Optimization (invited paper)
The 1st International Workshop on Regression Testing (Regression 2011)
Berlin, Germany, March 2011.
MORTO agenda

Cost: *pick at least one*

Value: *pick at least one*

Constraints
MORTO agenda

Cost
Value
Constraints
Cost Objectives

Execution Time
Cost Objectives

Execution Time
Data Access Costs
Cost Objectives

Execution Time
Data Access Costs
Third Party Costs
Cost Objectives

Execution Time
Data Access Costs
Third Party Costs
Technical resource Costs
Cost Objectives

Execution Time
Data Access Costs
Third Party Costs
Technical resource Costs
Set up Costs
Cost Objectives

- Execution Time
- Data Access Costs
- Third Party Costs
- Technical resource Costs
- Set up Costs
- Simulation Costs
MORTO agenda

Cost

Value

Constraints
Value Objectives
Value Objectives

Code Coverage
Value Objectives

Code Coverage
Non Code Coverage
Value Objectives

Code Coverage
Non Code Coverage
Fault Model Sensitive
Value Objectives

Code Coverage
Non Code Coverage
Fault Model Sensitive
Fault history Sensitive
Value Objectives

Code Coverage
Non Code Coverage
Fault Model Sensitive
Fault history Sensitive
Human Sensitive
Value Objectives

Code Coverage
Non Code Coverage
Fault Model Sensitive
Fault history Sensitive
Human Sensitive
Business Sensitive
MORTO agenda

Cost

Value

Constraints
Constraints

Precedence
Conjunction
Exclusion
Dependence

... add constraints to your taste
Constraints

Precedence

Conjunction

Exclusion

Dependence

... some tests come before others
Constraints

Precedence

Conjunction

Exclusion

Dependence

... some tests just go together
Constraints

- Precedence
- Conjunction
- Exclusion
- Dependence

... some tests can’t go together
Constraints

Precedence
Conjunction
Exclusion
Dependence

... some tests depend on others
SBSE Unites ...
SBSE is so generic
SBSE is so generic

Test generation
Fitness function: time ...
Representation: input vector
SBSE is so generic

Test generation
  Fitness function: time ...
  Representation: input vector

Requirements
  Fitness function: cost, value ...
  Representation: bitset of requirements
SBSE is so generic

Test generation
  Fitness function: time ...
  Representation: input vector

Requirements
  Fitness function: cost, value ...
  Representation: bitset of requirements

Regression
  Fitness function: coverage, time, faults
  Representation: bitset of test cases
SBSE is so generic

Test generation
Fitness function: time ...
Representation: input vector

Requirements
Fitness function: cost, value ...
Representation: bitset of requirements

Regression
Fitness function: coverage, time, faults
Representation: bitset of test cases
Selection Problems

Requirements Assignment

Select

Requirements

\[ R_1, R_2, \ldots, R_n \]

Customers

\[ C_1, C_2, \ldots, C_m \]

Regression Testing

Select

Test Cases

\[ T_1, T_2, \ldots, T_n \]

Covered Items in Code (e.g. statements)

Prioritization Problems

Requirements

\[ R_1, R_2, \ldots, R_n \]

Permutation \rightarrow \[ R_4, R_7, R_{12}, \ldots \]

Optimal Order of Development

Test Cases

\[ T_1, T_2, \ldots, T_n \]

Permutation \rightarrow \[ T_3, T_5, T_2, \ldots \]

Optimal Order of Re-testing

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Requirements and regression testing: really different?

Alone
Requirements and regression testing: really different?
Requirements and regression testing: really different?
... but ...

why is

Software Engineering different?
Traditional Engineering Artifact
Traditional Engineering Artifact
Traditional Engineering Artifact

Optimization goal

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Traditional Engineering Artifact

Optimization goal

Maximize compression
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption
Traditional Engineering Artifact

Optimization goal

Maximize compression

Minimize fuel consumption

Fitness computed on a representation
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Fitness computed on a representation
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Software Engineering Artifact

Fitness computed on a representation
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Fitness computed on a representation

Software Engineering Artifact

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Fitness computed on a representation

Software Engineering Artifact

Optimization goal

Maximize cohesion

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Fitness computed on a representation

Software Engineering Artifact

Optimization goal

Maximize cohesion
Minimize coupling
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Software Engineering Artifact

Optimization goal

Maximize cohesion
Minimize coupling

Fitness computed on a representation
Fitness computed directly
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Software Engineering Artifact

Optimization goal

Maximize cohesion
Minimize coupling

Fitness computed on a representation

Fitness computed Directly
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Software Engineering Artifact

Optimization goal

Maximize cohesion
Minimize coupling

Fitness computed on a representation

Fitness computed directly

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Traditional Engineering Artifact

Optimization goal

Maximize compression
Minimize fuel consumption

Software Engineering Artifact

Optimization goal

Maximize cohesion
Minimize coupling

Fitness computed on a representation

Fitness computed Directly

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Traditional Engineering Artifact

Optimization goal

Fitness computed on a representation

Maximize compression
Minimize fuel consumption

Software Engineering Artifact

Optimization goal

Fitness computed Directly

Maximize cohesion
Minimize coupling

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Human in the Loop
Regression Testing

“I don’t agree. This is the way it should be!”

“We need to prioritise business concerns”
Interleaved Clusters Prioritisation

T1  T2  T3  T6  T4  T5
Interleaved Clusters Prioritisation

Cluster

T1  T3  T6

T2  T4

T5

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Interleaved Clusters Prioritisation

Cluster

Intra-cluster Prioritisation

T3  T6  T1  T4  T2  T5
Interleaved Clusters Prioritisation

Cluster
Intra-cluster Prioritisation
Inter-cluster Prioritisation

T3 T6 T1
T5
T4 T2

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Interleaved Clusters Prioritisation

Cluster
Intra-cluster Prioritisation
Inter-cluster Prioritisation
Interleaving Clusters

T1
T3
T4
T5
T6

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Interleaved Clusters Prioritisation

Cluster
Intra-cluster Prioritisation
Inter-cluster Prioritisation
Interleaving Clusters

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Interleaved Clusters Prioritisation

- Cluster
  - Intra-cluster Prioritisation
  - Inter-cluster Prioritisation
  - Interleaving Clusters
Interleaved Clusters Prioritisation

Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Interleaved Clusters Prioritisation

Cluster
Intra-cluster Prioritisation
Inter-cluster Prioritisation
Interleaving Clusters

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Interleaved Clusters Prioritisation

Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters
Interleaved Clusters Prioritisation

Cluster
Intra-cluster Prioritisation
Inter-cluster Prioritisation
Interleaving Clusters

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Experimental Setup

- Prioritised Test Suite
- Simple Agglomerative Hierarchical Clustering (k=14)
- Hamming distance between stmt. coverage as dissimilarity metric
- A human user model with controlled error rate

- Human Pairwise Comparison
- Structural Coverage

- C1
- C2
- C13
- C14
Tolerance

This is what we initially expected to see.
Tolerance

This is what we initially expected to see.

But...
Tolerance

Some test suites are very resilient to errors
Some test suites are very resilient to errors.
Some test suites are very resilient to errors

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Some test suites are very resilient to errors

Mark Harman, UCL CREST centre
Optimising Mutation Testing
FOMs and HOMs

Original Program
FOMs and HOMs

Original Program
FOMs and HOMs

Original Program

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
FOMs and HOMs

Original Program  First Order Mutants

SBSE: Optimising Automation  Mark Harman, UCL CREST centre
FOMs and HOMs

Original Program

First Order Mutants

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
FOMs and HOMs

Original Program  First Order Mutants

SBSE: Optimising Automation  Mark Harman, UCL CREST centre

Tuesday, 6 September 11
FOMs and HOMs

Original Program

First Order Mutants

Higher Order Mutants

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Mutation Testing Repository

Repository News

Welcome to Mutation Testing repository

Mutation Testing is a fault-based software testing technique that has been widely studied for over three decades. The literature on mutation testing has contributed a set of approaches, tools and empirical studies. This repository aims to provide a full coverage of the publications in the literature on Mutation Testing.

Mutation Testing Survey

Using this repository, a comprehensive analysis and survey of Mutation Testing work has been conducted. The paper presents the results of several development trend analyses. These analyses provide evidence that mutation testing techniques and tools are reaching a state of maturity and applicability, while the topic of mutation testing itself is the subject of increasing interest. This paper is currently published as a technical report, available from here. If you want to cite results from the survey, here is the BibTeX entry:

```
@TECHREPORT{Jia09b,  
author = {Yue Jia and Mark Harman},  
title = {An Analysis and Survey of the Development of Mutation Testing},  
institution = {CREST Centre, King’s College London},  
year = {2009},  
type = {Technical},  
number = {18-09},  
address = {London},  
month = {May},
```

Quick Search

Search for paper, author

View all papers | authors

Repository Status

Last update: 14/10/09
Number of Papers: 368
Number of Authors: 295
Detailed status...

Links

: Mutation Testing Online
: Mutation Testing...
Mutation Testing Repository

Yue Jia and Mark Harman
An Analysis and Survey of the Development of Mutation Testing (*TSE to appear*)
Research Publications

Number of Publications

77 79 81 83 85 87 89 91 93 95 97 99 01 03 05 07 09

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Research Publications

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Applications
Applications


SBSE: Optimising Automation
Applications

Algebraic Specification
Petri Nets
Finite State Machine
Calculus Specification

Network Protocol
Statecharts
Real Time System

Ada
C/C++
C#
Fortran
Java
Lustre
SQL

ASP
SQL

Security Policy
Spreadsheets
Web Service

Real Time System
Empirical Studies

Newly applied subject programs
Laboratory programs vs real programs
Size of the largest program
Laboratory vs Real

- Laboratory Programs (Cumulative view)
- Real Programs (Cumulative view)

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Summary

mutation testing techniques and tools are reaching a state of maturity and applicability

Yue Jia and Mark Harman
An Analysis and Survey of the Development of Mutation Testing (*TSE to appear*)

http://www.dcs.kcl.ac.uk/pg/jiayue/repository/
Mutant Reduction Techniques
Mutant Reduction Techniques
Mutant Reduction Techniques
Mutant Reduction Techniques

Strong Mutation
Firm Mutation
Weak Mutation
Interpreter
Compiler
SIMD
MIMD
Parallel
Byte Translation
Mutant Schemata Generation

Constraint Mutation
N-Selective Mutation
Mutation

Execution

Time line

78 80 82 84 86 88 90 92 94 96 98 00 02 04 06 08

Mutant Reduction Techniques

The Coupling Effect: Fact or Fiction (Offutt’89)

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
FOMs and HOMs

Original Program
FOMs and HOMs

Original Program

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
FOMs and HOMs

Original Program

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
FOMs and HOMs

Original Program

First Order Mutants

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
FOMs and HOMs

Original Program  First Order Mutants
FOMs and HOMs

Original Program

First Order Mutants

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
FOMs and HOMs

Original Program

First Order Mutants

Higher Order Mutants

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
FOMs and HOMs
FOMs and HOMs

Mutants
Domain
FOMs and HOMs

Mutants Domain

FOMs

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Traditional FOM Testing
Traditional FOM Testing

Higher Order Mutants are far too numerous
Traditional FOM Testing

Higher Order Mutants are far too numerous

Competent Programmer
Traditional FOM Testing

Higher Order Mutants are far too numerous

Competent Programmer

Mutation Coupling Effect
Mutation Coupling Effect
Mutation Coupling Effect
Mutation Coupling Effect
Mutation Coupling Effect

Simple / FOMs

Complex / HOMs

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Mutation Coupling Effect

If a test set kills all FOMs, it will kill a large percentage of the HOMs (Offutt 1992)

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Mutation Coupling Effect

If a test set kills all FOMs, it will kill a large percentage of the HOMs (Offutt 1992)
If a test set kills all FOMs, it will kill a large percentage of the HOMs (Offutt 1992)
If a test set kills all FOMs, it will kill a large percentage of the HOMs (Offutt 1992)
First Order Restriction

FOMs are easily killed

e.g. $+ \rightarrow -$
First Order Restriction

Are FOMs really like real faults?
First Order Restriction

real faults require several edits to fix them

AT&T 5ESS Telephone Switch (90%)
(Purushothaman and Perry, TSE 2005)

Ericsson Telecom Middleware (50%)
(Eldh et al., FATE 2007)
Search for HOMs

search space of mutants

search for good mutants
Search based MT

Why not search for mutants that are hard to kill?
Search based MT

Why not search for mutants that are hard to kill?

- Tabu Search
- Ant Colonies
- Hill Climbing
- Simulated Annealing
- Particle Swarm Optimization
- Genetic Algorithms
- Estimation of Distribution Algorithms
- Genetic Programming
- Greedy
- LP
- Random
Search based MT

Why not search for mutants that are hard to kill?
Aim

Searching avoids enumerating all mutants

SBSE: Optimising Automation
Mark Harman, UCL CREST Centre
Approaches

Single Objective
- Genetic Algorithm
- Hill Climbing
- Greedy

Multi Objective
- Genetic Programming
Single Objective

Data Representation

Index

Position
Single Objective

Data Representation

Index

Position

1 0 0 0
Single Objective

Data Representation

Index

Position

1 0 0 0

0 0 2 0
Single Objective

Data Representation

Index

Position

1 0 0 0
0 0 2 0
Single Objective

Data Representation

Index

Position

1 0 0 0
0 0 2 0
1 0 2 0

Mark Harman, UCL CREST centre
Single Objective

\[
\text{fitness} = \frac{\text{Test cases that kill the FOMs}}{\text{Test cases that kill the HOM}}
\]

- \( \text{fitness} \geq 1 \), easier to be killed
- \( \text{fitness} < 1 \), harder to be killed
- \( \text{fitness} = 0 \), cannot to be killed
Algorithms

- GA
- Crossover + Mutation
- Greedy
- HC
Algorithms

GA

Crossover + Mutation

Greedy

HC
Algorithms

- GA
- Crossover + Mutation
- Greedy
- HC

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Algorithms

GA

Crossover + Mutation

Greedy

HC
Algorithms

GA

Crossover + Mutation

Greedy

HC

SBSE: Optimising Automation  Mark Harman, UCL CREST centre
Algorithms

GA

Crossover + Mutation

Greedy

HC

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective
Multi Objective

source.c

Pareto Evolution

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Multi Objective

source.c → BNF Grammar

Pareto Evolution

SBSE: Optimising Automation

Mark Harman, UCL CREST centre
Multi Objective

source.c → BNF Grammar → GP → 10000 mutants

Pareto Evolution

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Multi Objective

source.c → BNF Grammar → GP → 10000 mutants

↓
gcc

population.exe ← Test Cases

Pareto Evolution
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Generation 0 Pareto front
No children
1 Child
2 Children

Number test cases with different output

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

HOM Testing
Classification
Future Work

Overview
FOM Restriction
HOM Testing
Approaches

Tuesday, 6 September 11
Multi Objective

Overview
FOM Restriction
HOM Testing
Approaches
Classification
Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
FOM Restriction
Approaches
Classification
Future Work

Tuesday, 6 September 11

CREST Centre, Mark Harman

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Generation 4 Pareto front
- No children
- 1 Child
- 2 Children

Number test cases with different output

Approaches
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
FOM Testing
HOM Testing
Multi Objective
Classification
Future Work

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
- HOM Testing

FOM Testing
- Classification
- Future Work

Tuesday, 6 September 11

CREST Centre, Mark Harman
MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
FOM Restriction
Classification
Future Work

Approaches

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
FOM Restrictions
HOM Testing
Approaches
Classification
Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
FOM Restriction
Approaches
Classification
Future Work

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
FOM Restriction
Future Work
Classification
Approaches

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

- Generation 11 Pareto front
- No children
- 1 Child
- 2 Children

Number test cases with different output

Approaches
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
Future Work

Approaches

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
Future Work
Approaches

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Number test cases with different output

Syntax distance
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Approaches
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Syntax distance

Number test cases with different output

Generation 16 Pareto front
No children
1 Child
2 Children

Approaches
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview

HOM Testing

Classification

Future Work

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

- Overview
- HOM Testing
- Approaches
- Classification
- Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Number test cases with different output

Syntax distance

Generation 20 Pareto front
No children
1 Child
2 Children

Overview
HOM Testing
Future Work
Classification
FOM Restriction
Approaches
Overview

Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Generation 21 Pareto front
No children
1 Child
2 Children

Number test cases with different output
Multi Objective
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Generation 23 Pareto front
- No children
- 1 Child
- 2 Children

Syntax distance

Number test cases with different output
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Generation 24 Pareto front
No children
1 Child
2 Children

Syntax distance

Number test cases with different output
Multi Objective

Overview

HOM Testing

Classification

Future Work

CREST Centre, Mark Harman

Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
Classifiers
FOM Testing
Future Work

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
Classification
Future Work

Approaches

Tuesday, 6 September 11

CREST Centre, Mark Harman
MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Generation 28 Pareto front
No children
1 Child
2 Children

Syntax distance

Number test cases with different output

Overview
FOM Restriction
HOM Testing
Approaches
Classification
Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

- Overview
- HOM Testing
- Approaches
- Classification
- Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

- Overview
- HOM Testing
- FOM Restriction
- Approaches
- Classification
- Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Approaches

Overview  FOM Restriction  HOM Testing  Classification  Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
Approaches
Future Work

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
Future Work

Approaches
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Number test cases with different output

Syntax distance

Generation 34 Pareto front
No children
1 Child
2 Children

Overview
FOM Restriction
HOM Testing
Approaches
Classification
Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview

HOM Testing

FOM Restriction

Classification

Future Work

Approaches
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Generation 36 Pareto front
No children
1 Child
2 Children

Number test cases with different output

Syntax distance

Approaches
Multi Objective
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

- Overview
- HOM Testing
- Approaches
- Classification
- Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview

HOM Testing

Classifications

Future Work

Approaches

Tuesday, 6 September 11

CREST Centre, Mark Harman

MUTATION 2010 Keynote
Multi Objective
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

- Overview
- HOM Testing
- Approaches
- Classification
- Future Work

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Number of test cases with different output

Syntax distance

Generation 42 Pareto front
No children
1 Child
2 Children

Approaches
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

- Overview
- FOM Restriction
- HOM Testing
- Approaches
- Classification
- Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview
HOM Testing
FOM Restriction
Approaches
Classification
Future Work

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Number test cases with different output

Syntax distance

Overview
FOM Restriction
HOM Testing
Classification
Future Work

CREST Centre, Mark Harman
Tuesday, 6 September 11

MUTATION 2010 Keynote
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview

FOM Restriction

HOM Testing

Approaches

Classification

Future Work

Tuesday, 6 September 11
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

- Overview
- HOM Testing
- Classification
- Future Work

Approaches
Multi Objective
Multi Objective
Multi Objective

Evolution of Multi-Objective Higher Order Mutants with NSGA-II and Genetic Programming

Overview

HOM Testing

FOM Restriction

Approaches

Future Work

Classification

MUTATION 2010 Keynote

Tuesday, 6 September 11
HOM Classification
HOM classification

Most common case

FOM a is killed by \{ 1, 2, 3, 4 \}

Test set T
HOM classification

Most common case

FOM a is killed by \{ 1, 2, 3, 4 \}

Test set T

\[
\begin{array}{c}
1 \\
2 \\
3 \\
4 \\
\end{array}
\]

Mark Harman, UCL CREST centre
HOM classification

Most common case

FOM a is killed by 
{ 1, 2, 3, 4 }

FOM b is killed by 
{ 3, 4, 5, 6 }

Test set T

1
2
3
4
Ta
HOM classification

Most common case

FOM a is killed by \{ 1, 2, 3, 4 \}
FOM b is killed by \{ 3, 4, 5, 6 \}

Test set T
HOM classification

Most common case

FOM a is killed by
\{ 1, 2, 3, 4 \}

FOM b is killed by
\{ 3, 4, 5, 6 \}

HOM ab is killed by
\{ 1, 2, 3, 4, 5, 6 \}
HOM classification

Most common case

FOM a is killed by \{ 1, 2, 3, 4 \}
FOM b is killed by \{ 3, 4, 5, 6 \}
HOM ab is killed by \{ 1, 2, 3, 4, 5, 6 \}
HOM classification

Most common case

FOM a is killed by \{ 1, 2, 3, 4 \}
FOM b is killed by \{ 3, 4, 5, 6 \}
HOM ab is killed by \{ 2, 3, 5 \}
HOM classification

Many types of HOM

Subsuming HOM

Test set $T$

$T_a$

$T_b$

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
HOM classification

Many types of HOM

Subsuming HOM

Strongly Subsuming

Test set $T$

$1$ $2$ $3$ $4$ $5$ $6$

$T_a$ $T_b$ $T_{ab}$
HOM classification

Many types of HOM

- Subsuming HOM
- Strongly Subsuming
- Anti-Coupling Effect

Test set $T$

$1 \ 2 \ 3 \ 4 \ 5 \ 6$

$T_a \ T_b$

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
HOM classification

Many types of HOM

- Subsuming HOM
- Strongly Subsuming
- Anti-Coupling Effect
- Equivalent

Test set $T$

- $1$
- $2$
- $3$
- $4$
- $5$
- $6$

Mark Harman, UCL CREST centre
# Single Objective Results

<table>
<thead>
<tr>
<th>Program</th>
<th>LoC</th>
<th>FOM</th>
<th>SHOM</th>
<th>SSHOMs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triangle</td>
<td>50</td>
<td>601</td>
<td>14.79%</td>
<td>0.24%</td>
</tr>
<tr>
<td>Tcas</td>
<td>150</td>
<td>744</td>
<td>10.21%</td>
<td>0.11%</td>
</tr>
<tr>
<td>Schedule2</td>
<td>350</td>
<td>1,603</td>
<td>32.81%</td>
<td>0.27%</td>
</tr>
<tr>
<td>Schedule</td>
<td>400</td>
<td>1,213</td>
<td>15.96%</td>
<td>0.39%</td>
</tr>
<tr>
<td>Totinfo</td>
<td>500</td>
<td>2,316</td>
<td>20.61%</td>
<td>0.24%</td>
</tr>
<tr>
<td>Replace</td>
<td>550</td>
<td>4,195</td>
<td>20.22%</td>
<td>0.31%</td>
</tr>
<tr>
<td>Printtokens2</td>
<td>600</td>
<td>1,714</td>
<td>16.54%</td>
<td>0.10%</td>
</tr>
<tr>
<td>Printtokens</td>
<td>750</td>
<td>1,237</td>
<td>24.33%</td>
<td>0.01%</td>
</tr>
<tr>
<td>Gzip</td>
<td>5,500</td>
<td>12,027</td>
<td>12.38%</td>
<td>0.08%</td>
</tr>
<tr>
<td>Space</td>
<td>6,000</td>
<td>68,843</td>
<td>7.29%</td>
<td>0.21%</td>
</tr>
</tbody>
</table>
Optimising Structural Testing
Program Coverage

control flow graph of a program containing a structure we want to cover
Fitness evaluation

The test data executes the ‘wrong’ path
Analysing control flow
Analysing control flow

The outcomes at key decision statements matter.

These are the decisions on which the target is control dependent.
Approach Level

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Approach Level

TARGET = 2

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Approach Level

TARGET = 2

TARGET = 1

TARGET = 0

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
Approach Level

TARGET = 2

minimisation

TARGET = 1

TARGET = 0

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Analysing predicates

Approach level alone gives us coarse values

```java
if (a == b) {
    // ...
}
```

- a = 50, b = 0
- a = 45, b = 5
- a = 40, b = 10
- a = 35, b = 15
- a = 30, b = 20
- a = 25, b = 25

Mark Harman, UCL CREST centre
Analysing predicates

Approach level alone gives us coarse values

```
if (a == b) {
    // ....
}
```

(a = 50, b = 0)
(a = 45, b = 5)
(a = 40, b = 10)
(a = 35, b = 15)
(a = 30, b = 20)
(a = 25, b = 25)

going 'closer' to being true
Branch distance

Associate a distance formula with different relational predicates

```java
if (a == b) {
   // ....
}
```

branch distance

- a = 50, b = 0, branch distance = 50
- a = 45, b = 5, branch distance = 40
- a = 40, b = 10, branch distance = 30
- a = 35, b = 15, branch distance = 20
- a = 30, b = 20, branch distance = 10
- a = 25, b = 25, branch distance = 0

getting ‘closer’ to being true
Putting it all together

Fitness = approach Level + *normalised* branch distance

```c
void f1(int a, int b, int c, int d)
{
    if (a > b)
    {
        if (b > c)
        {
            if (c > d)
            {
                // target
            }
        }
    }
}
```
Putting it all together

Fitness = approach Level + *normalised* branch distance

```c
void f1(int a, int b, int c, int d) {
    if (a > b)
    {
        if (b > c)
        {
            if (c > d)
            {
                // target
            }
        }
    }
...}
```

normalised branch distance between 0 and 1
indicates how close approach level is to being penetrated

TARGET MISSED
Approach Level = 2
Branch Distance = b - a
Putting it all together

Fitness = approach Level + *normalised* branch distance

```c
void f1(int a, int b, int c, int d) {
    if (a > b) {
        if (b > c) {
            if (c > d) {
                // target
            }
        }
    }
    ... 
}
```

*normalised* branch distance between 0 and 1 indicates how close approach level is to being penetrated
Putting it all together

Fitness = approach Level + *normalised* branch distance

```c
void f1(int a, int b, int c, int d)
{
    if (a > b)
    {
        if (b > c)
        {
            if (c > d)
            {
                // target
            }
        }
    }
    ...
}
```

normalised branch distance between 0 and 1
indicates how close approach level is to being penetrated

TARGET MISSED
Approach Level = 2
Branch Distance = b - a

TARGET MISSED
Approach Level = 1
Branch Distance = c - b

TARGET MISSED
Approach Level = 0
Branch Distance = d - c

if a >= b
true
false

if b >= c
true
false

if c >= d
true
false

TARGET

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing

Fitness vs. Input

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing

Input

Fitness

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing

Fitness

Input

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing

Fitness vs. Input

No better solution in neighbourhood
Stuck at a local optima

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing - Restarts

Fitness

Input

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing - Restarts

Fitness vs. Input

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing - Restarts

Fitness

Input

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing - Restarts

Fitness

Input

SBSE: Optimising Automation  Mark Harman, UCL CREST centre
Hill Climbing - Restarts

Fitness

Input

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Hill Climbing - Restarts

Fitness

Input

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Simulated Annealing

Fitness

Input

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Simulated Annealing

Fitness

Input

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Simulated Annealing

Worse solutions temporarily accepted
Simulated Annealing

SBSE: Optimising Automation
Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Evolutionary Algorithm

Fitness

Input

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Evolutionary Algorithm

Fitness

Input

SBSE: Optimising Automation

Mark Harman, UCL CREST centre

Tuesday, 6 September 11
Crossover

```c
void test_me(int a, int b, int c, int d) {
    if (a == b) {
        if (c == d) {
            // branch we want to execute
        }
    }
    ...
}
```
Crossover

```c
void test_me(int a, int b, int c, int d) {
    if (a == b) {
        if (c == d) {
            // branch we want to execute
        }
    }
    ...
}
```
Crossover

```c
void test_me(int a, int b, int c, int d) {
    if (a == b) {
        if (c == d) {
            // branch we want to execute
        }
    }
    ...
}
```

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>20</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>-5</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SBSE: Optimising Automation
Mark Harman, UCL CREST centre
void test_me(int a, int b, int c, int d) {
    if (a == b) {
        if (c == d) {
            // branch we want to execute
        }
    }
...
Mutation

```c
void test_me(int a, int b, int c, int d) {
    if (a == b) {
        if (c == d) {
            // branch we want to execute
        }
    }
    ...
}
```

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>20</td>
<td>40</td>
</tr>
</tbody>
</table>
void test_me(int a, int b, int c, int d) {
    if (a == b) {
        if (c == d) {
            // branch we want to execute
        }
    }
    ...
}
void test_me(int a, int b, int c, int d) {
    if (a == b) {
        if (c == d) {
            // branch we want to execute
        }
    }
    ...
}
void test_me(int a, int b, int c, int d) {
    if (a == b) {
        if (c == d) {
            // branch we want to execute
        }
    }
...