ssbse

symposium on search based software engineering

www.ssbse.org/2011

introduction

Mark Harman
UCL
London

not a repeat of my FSE TB

not a repeat of my FSE TB

- What is UCL CREST ?
- What is SBSE ?
- Why SBSE ?
  - Insight-rich
  - Generic
  - Software: the ideal engineering material to optimize
- Why SSBSE ?

not a repeat of my FSE TB

What is UCL CREST ?

What is SBSE ?

Why SBSE ?

- Insight-rich
- Generic
- Software: the ideal engineering material to optimize

Why SSBSE ?

not a repeat of my FSE TB

What is UCL CREST ?

What is SBSE ?

Why SBSE ?

... well OK, maybe some overlap

- Insight-rich
- Generic
- Software: the ideal engineering material to optimize

Why SSBSE ?

not a repeat of my FSE TB

What is UCL CREST ?

What is SBSE ?

Why SBSE ?

... well OK, maybe some overlap

Insight-rich

... I will *seek* to *minimize* it

Generic

Software: the ideal engineering material to optimize

Why SSBSE ?

not a repeat of my FSE TB

What is UCL CREST ?

What is SBSE ?

Why SBSE ?

... well OK, maybe some overlap

- Insight-rich

... I will *seek* to *minimize* it

- Generic

... this is *SSBSE*

- Software: the ideal engineering material to optimize

Why SSBSE ?

not a repeat of my FSE TB

What is UCL CREST ?

What is SBSE ?

Why SBSE ?

- Insight-rich
- Generic
- Software: the ideal engineering material to optimize

Why SSBSE ?

Centre for Research on Evolution Search and Testing *Est. 2006*

CREST

Centre for Research on Evolution Search and Testing *Est. 2006*

1 admin

4 faculty

8 post docs

11 PhD students

1-4 resident visiting scholars

CREST

Research

Research

Service Oriented Computing

Dependence Analysis

Clone Detection

Search Based Software Engineering

Quantitative Information Flow

Testing

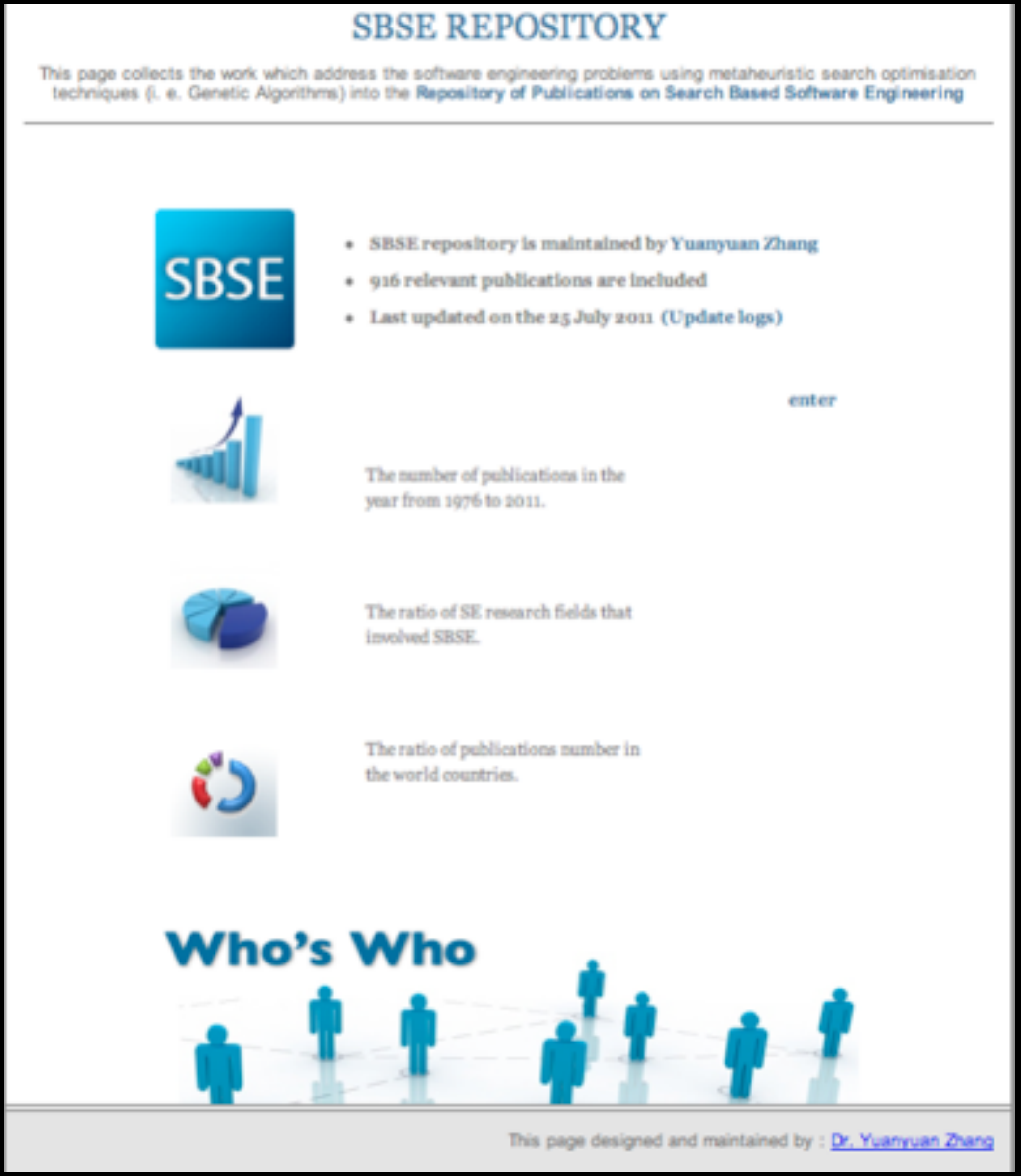
Requirements Engineering

Digital Humanities

Three Repositories

SBSE Repository

SBSE Repository



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

The number of publications in the year from 1976 to 2011.

The ratio of SE research fields that involved SBSE.

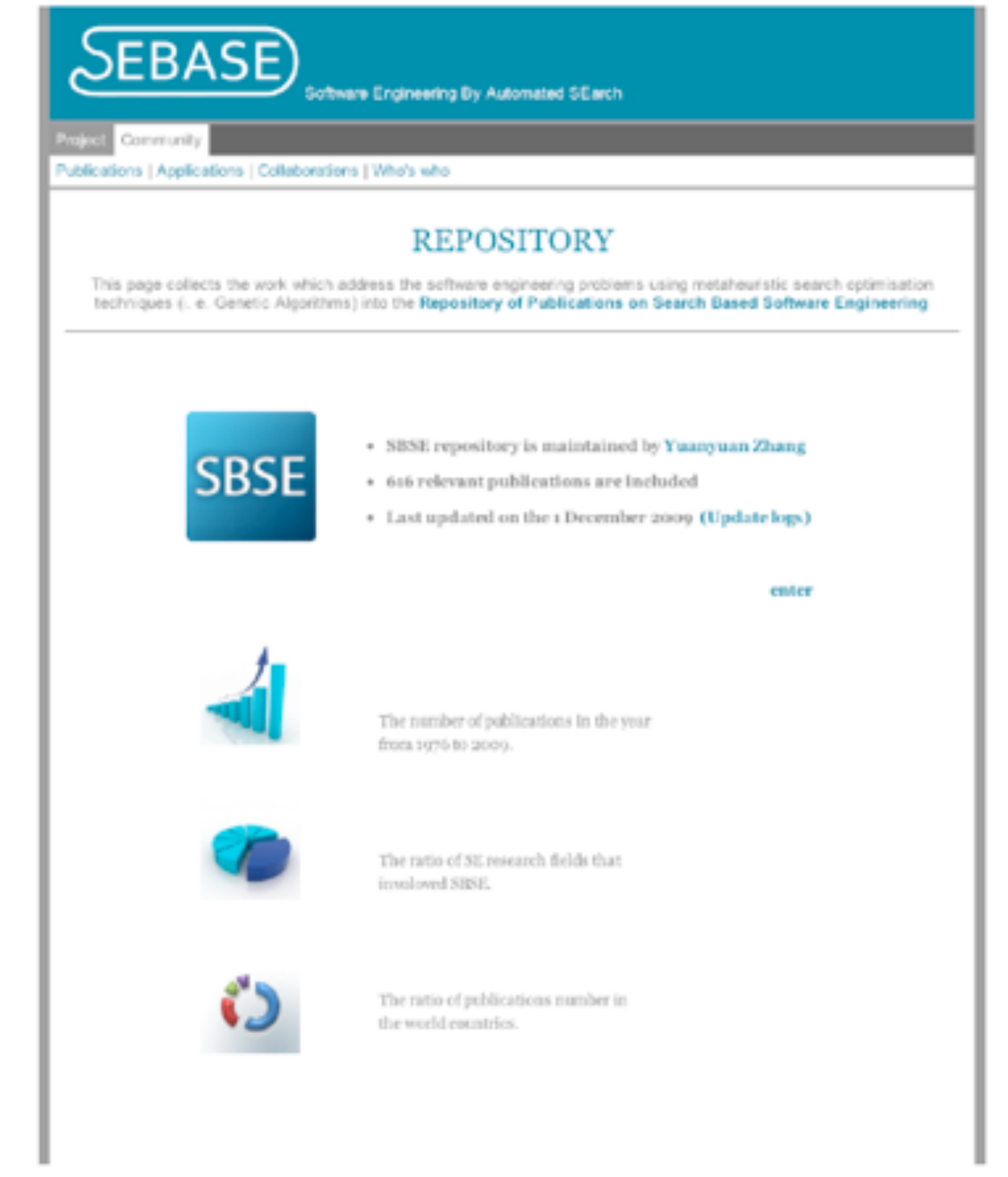
The ratio of publications number in the world countries.

Who's Who

This page designed and maintained by : Dr. Yuanyuan Zhang

Dr. Yuanyuan Zhang

SBSE Repository



SEBASE Software Engineering By Automated SEarch

Project | Community

Publications | Applications | Collaborations | Who's who

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**

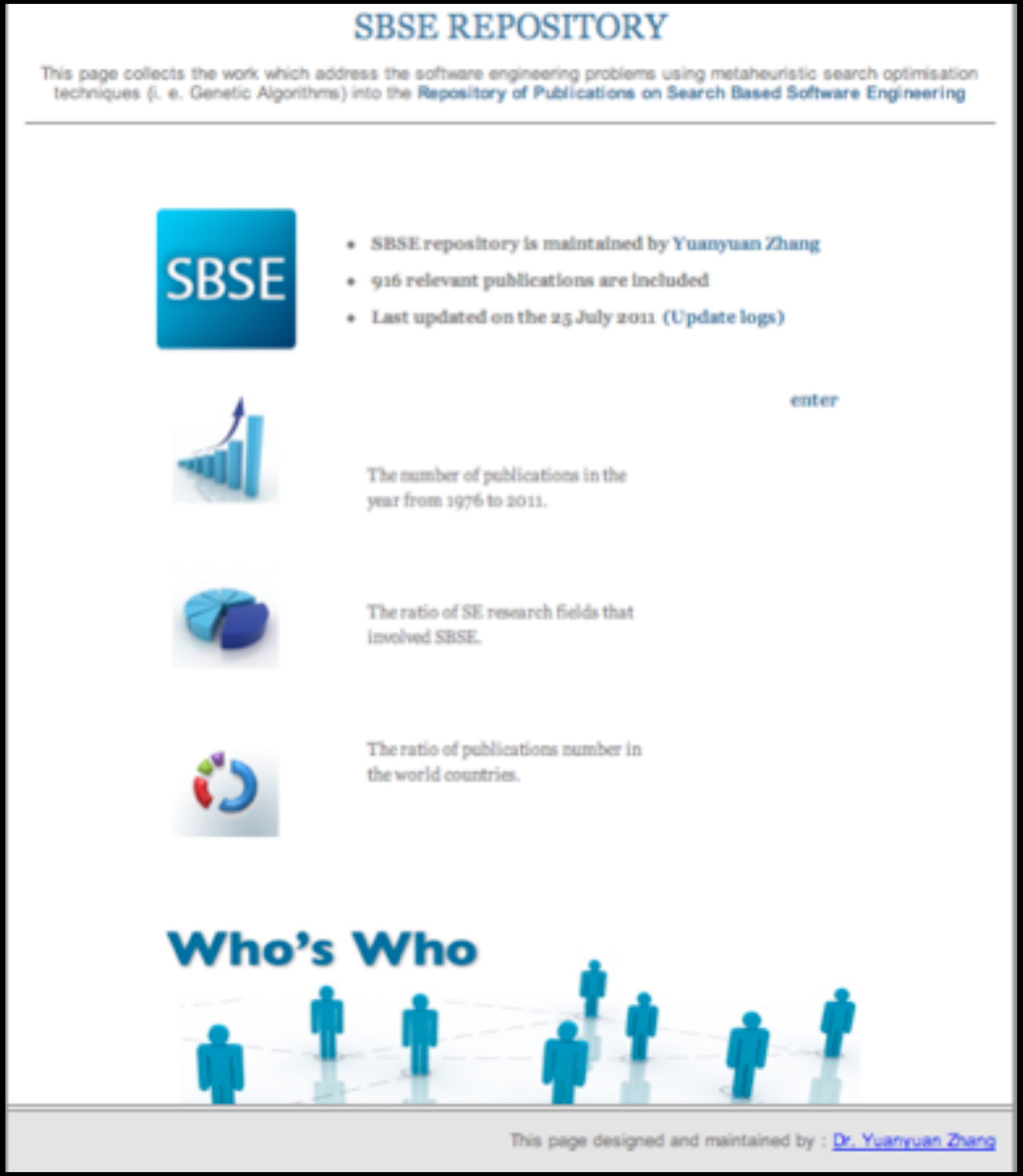
- SBSE repository is maintained by Yuanyuan Zhang
- 616 relevant publications are included
- Last updated on the 1 December 2009 (Update logs)

enter

The number of publications in the year from 1976 to 2009.

The ratio of SE research fields that involved SBSE.

The ratio of publications number in the world countries.



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

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.

Who's Who

This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository

SE SE Software Engineering By Automated SEarch

Project Community

Publications | Applicat ations | Who's who

REPOSITORY

This page collects the work software engineering problems using meta ch optimisation techniques (i. e. Genetic Algo pository of Publications on Search B re Engineering

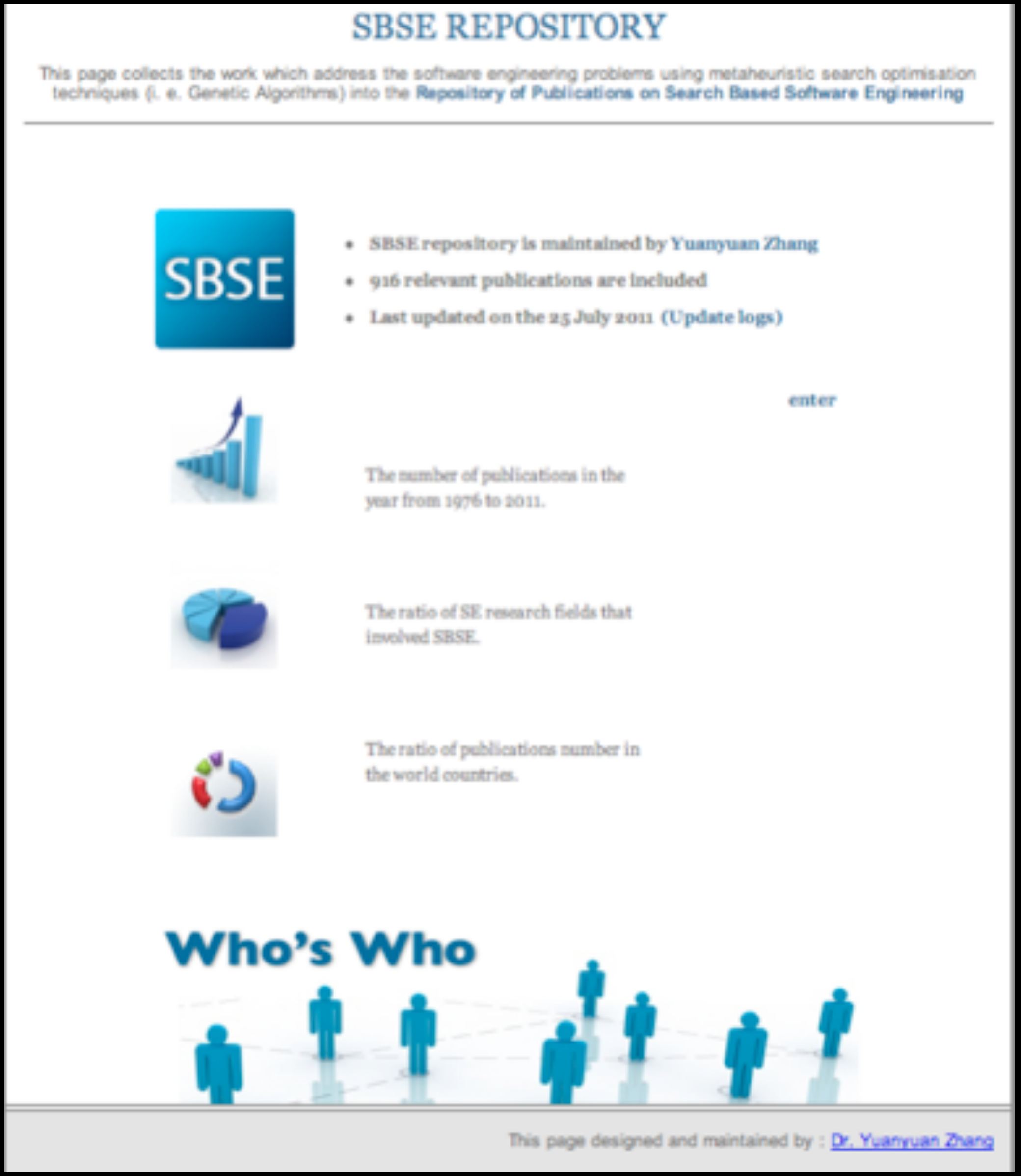
- SB uan Zhang
- 616 rele d
- Last updat 009 (Update logs)

enter

T ublications in the year 2009.

The ratio of SE research fields that involved SBSE.

The ratio of publications number in the world countries.



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

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.

Who's Who

This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository

SE SE
Software Engineering By Automated SEarch
Project Communit...
Publications | Applica... ...tions | Who's who

REPOSITORY

This page collects the work ... software engineering problems using metah... ...ch optimisation techniques (i. e. Genetic Algo... ...ository of Publications on Search B... ...e Engineering

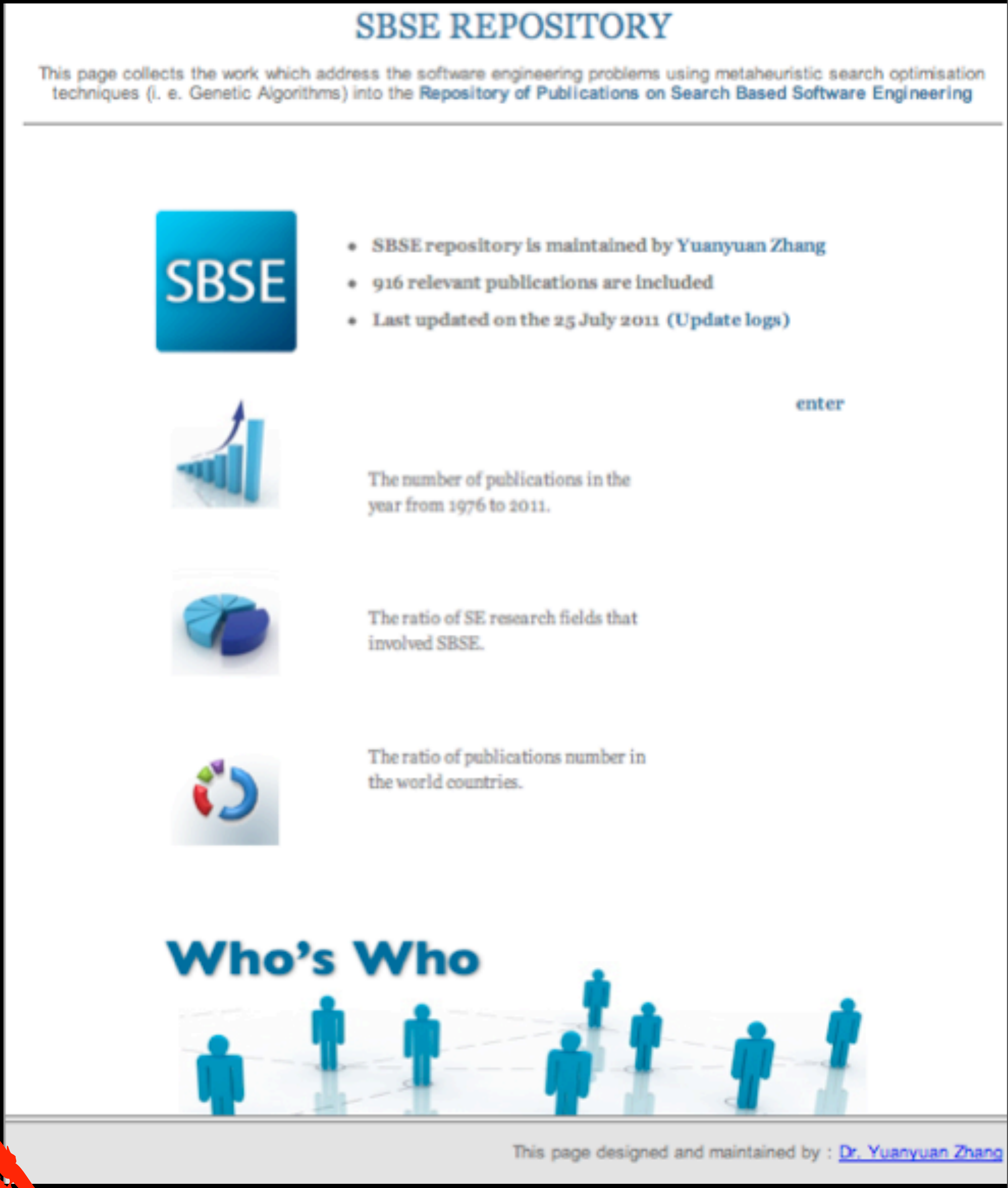
- SBS... ...uan Zhang
- 616 rele... ...d
- Last updat... ...009 (Update logs)

enter

...ublications in the year ...2009.

The ratio of SE research fields that involved SBSE.

The ratio of publications number in the world countries.



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

The number of publications in the year from 1976 to 2011.

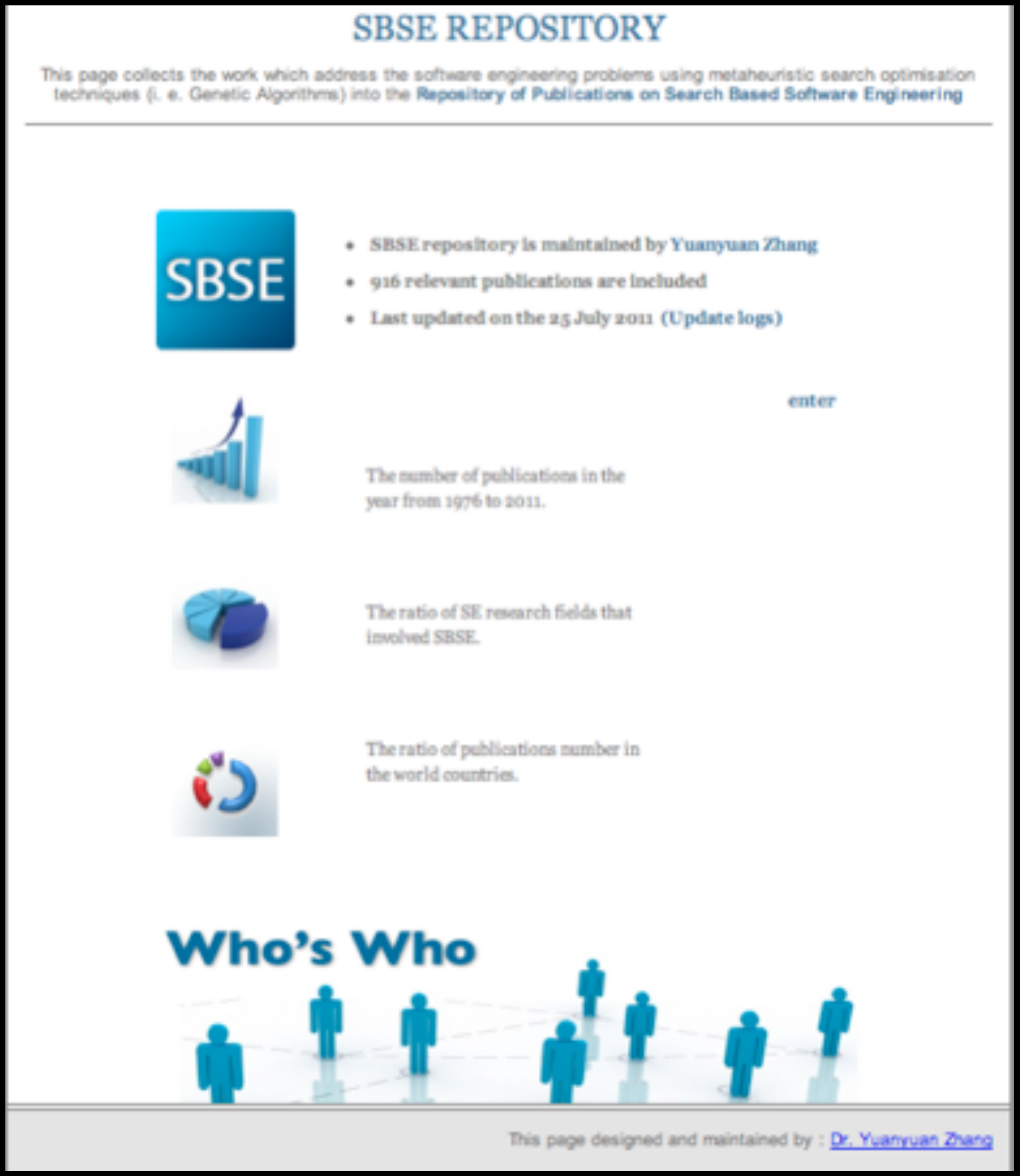
The ratio of SE research fields that involved SBSE.

The ratio of publications number in the world countries.

This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository

how to tell you
have the new
version?



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

The number of publications in the year from 1976 to 2011.

The ratio of SE research fields that involved SBSE.

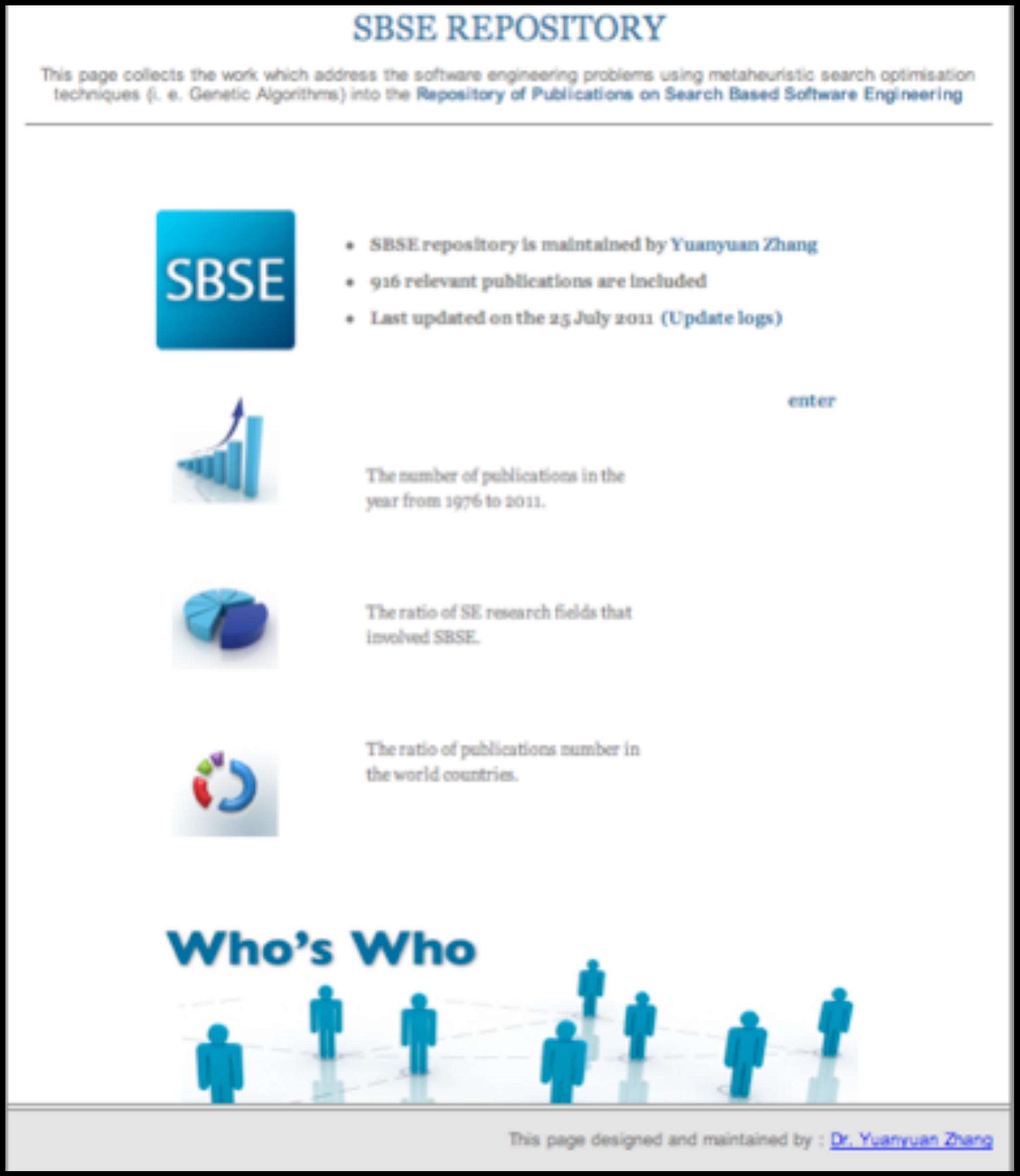
The ratio of publications number in the world countries.

This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository

how to tell you
have the new
version?

ucl
not
kcl



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

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.

Who's Who

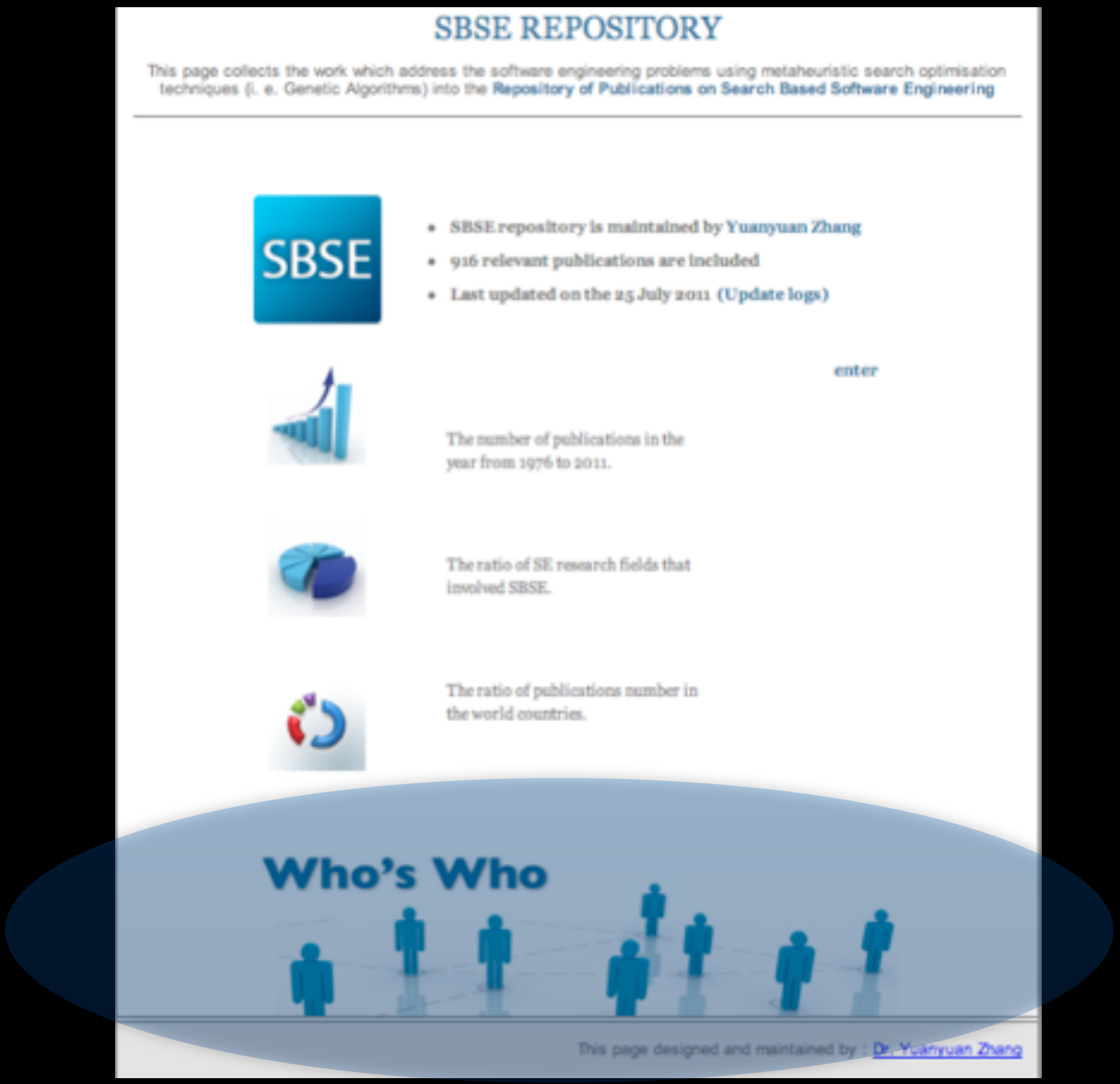
This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository

how to tell you
have the new
version?

ucl
not
kcl

it's the one with
a who's who



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

The number of publications in the year from 1976 to 2011.

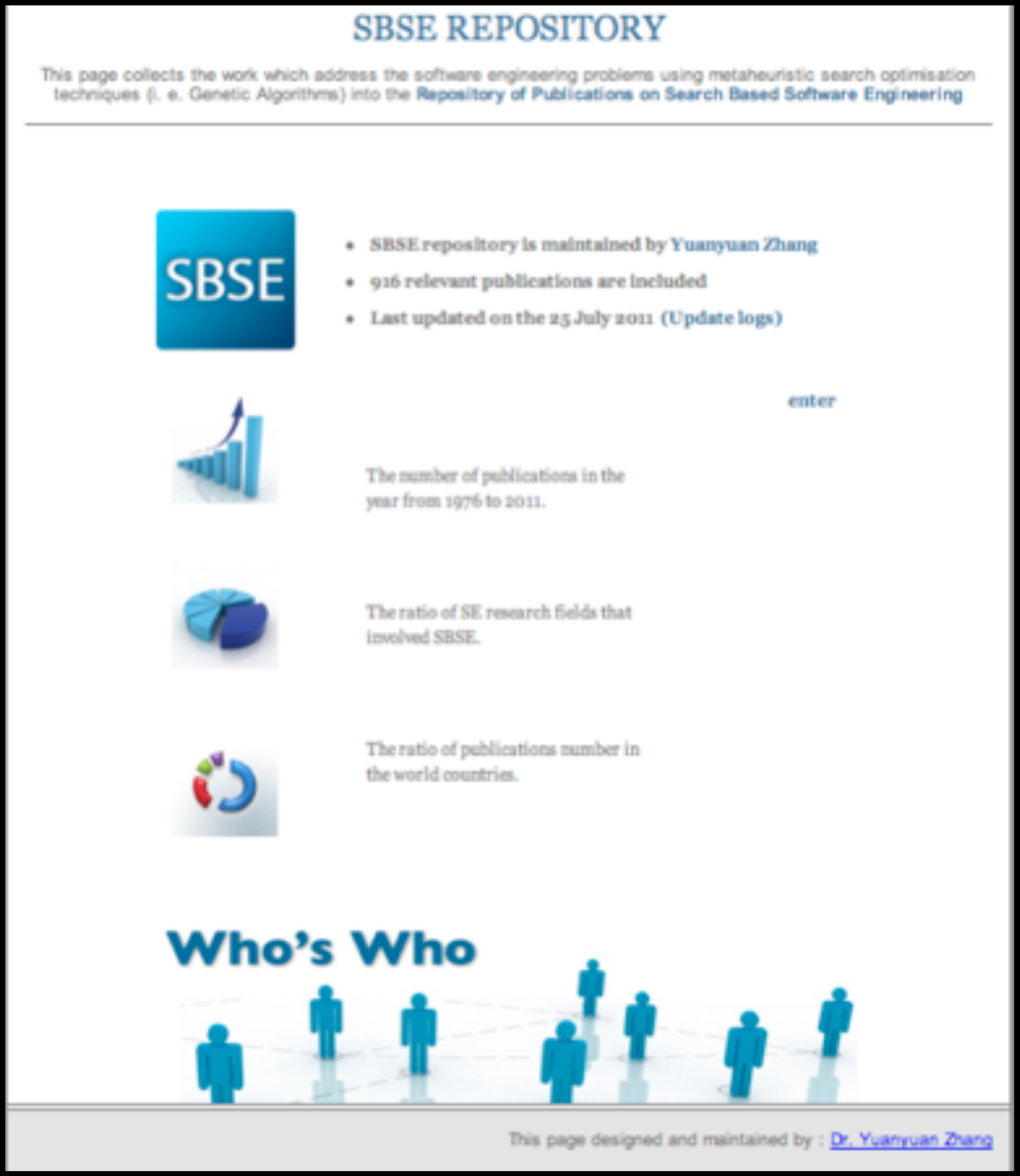
The ratio of SE research fields that involved SBSE.

The ratio of publications number in the world countries.

Who's Who

This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

The number of publications in the year from 1976 to 2011.

The ratio of SE research fields that involved SBSE.

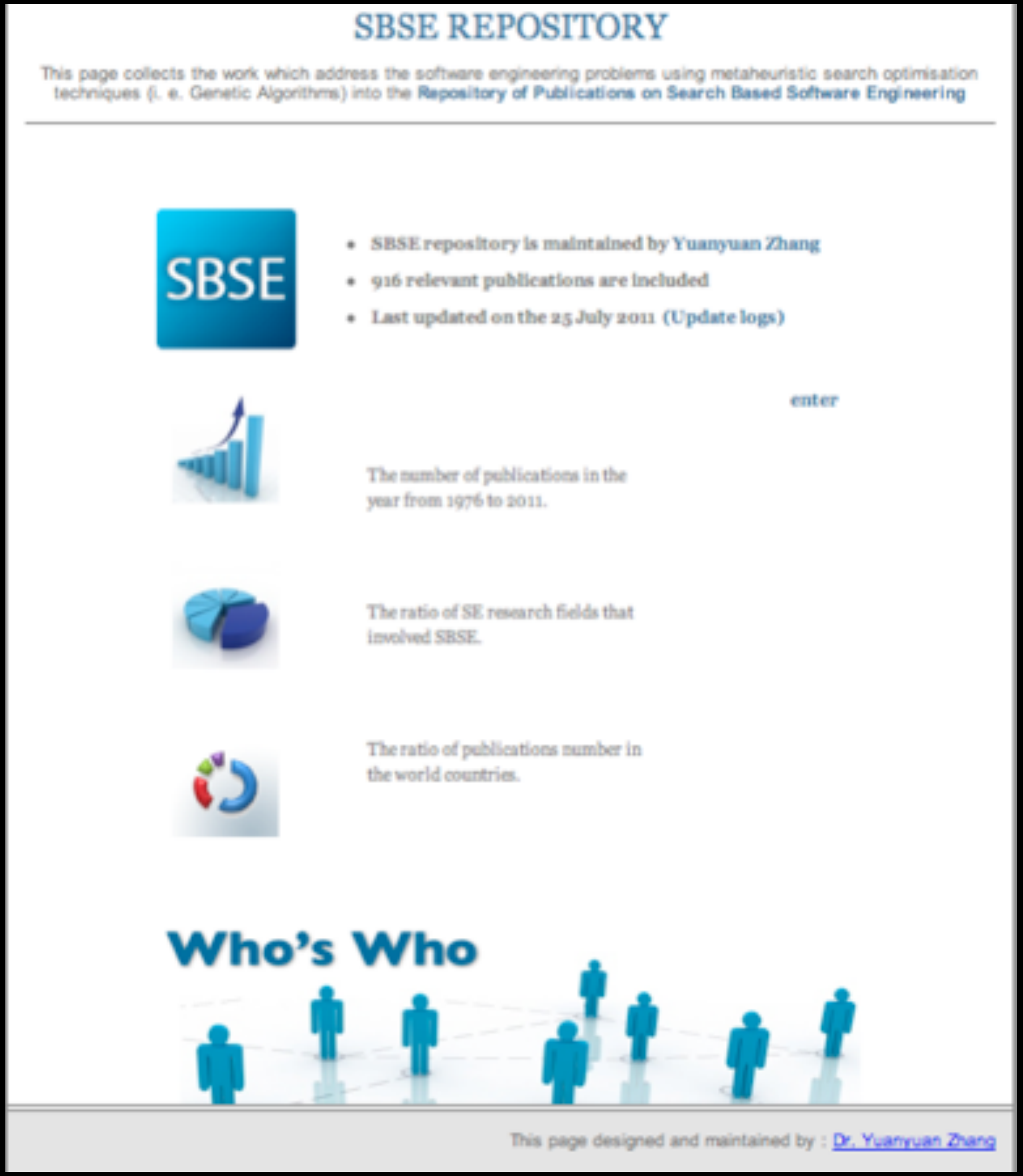
The ratio of publications number in the world countries.

Who's Who

This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository

Analysis



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

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.

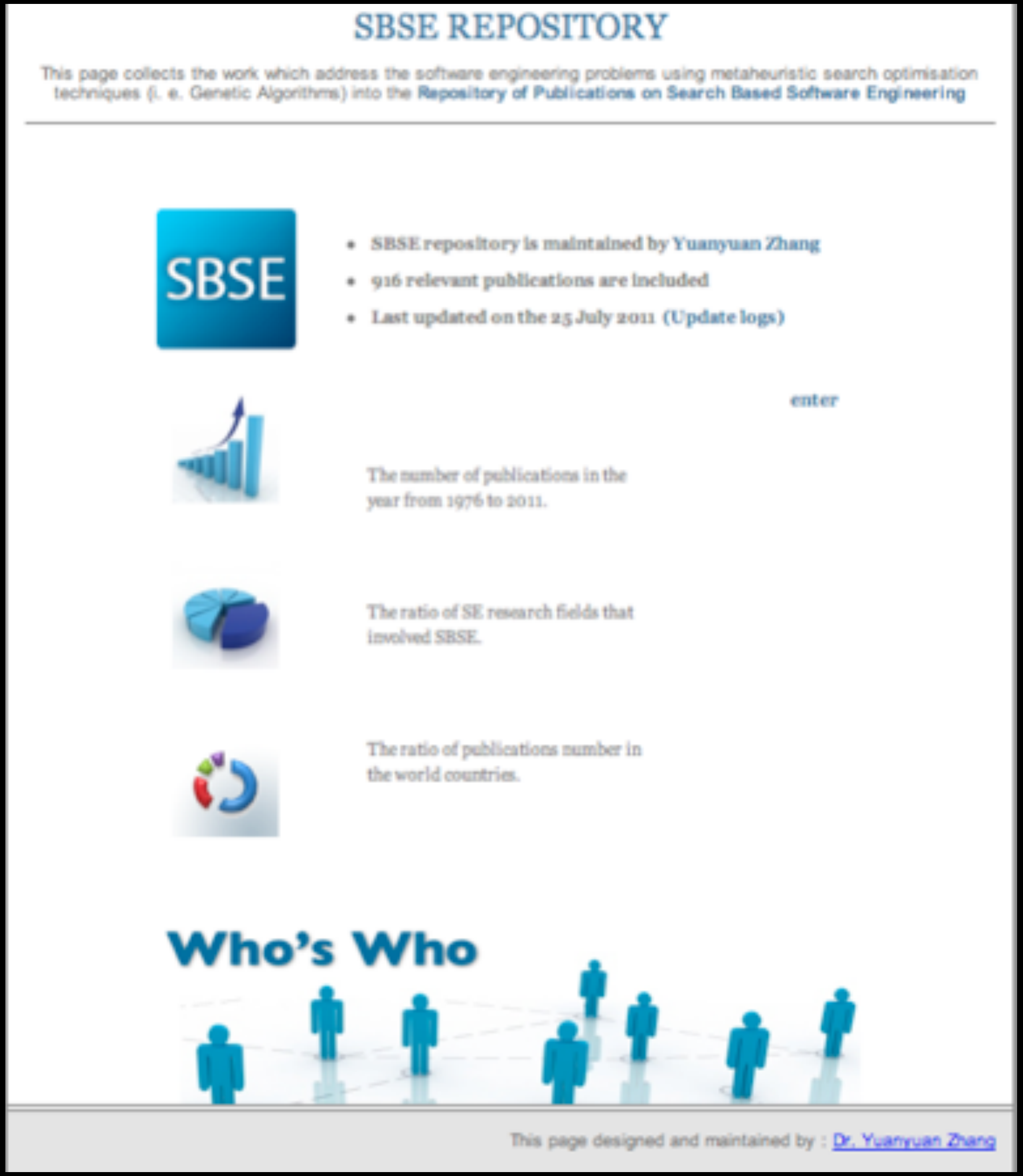
Who's Who

This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository

Analysis

extended who's who



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

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.

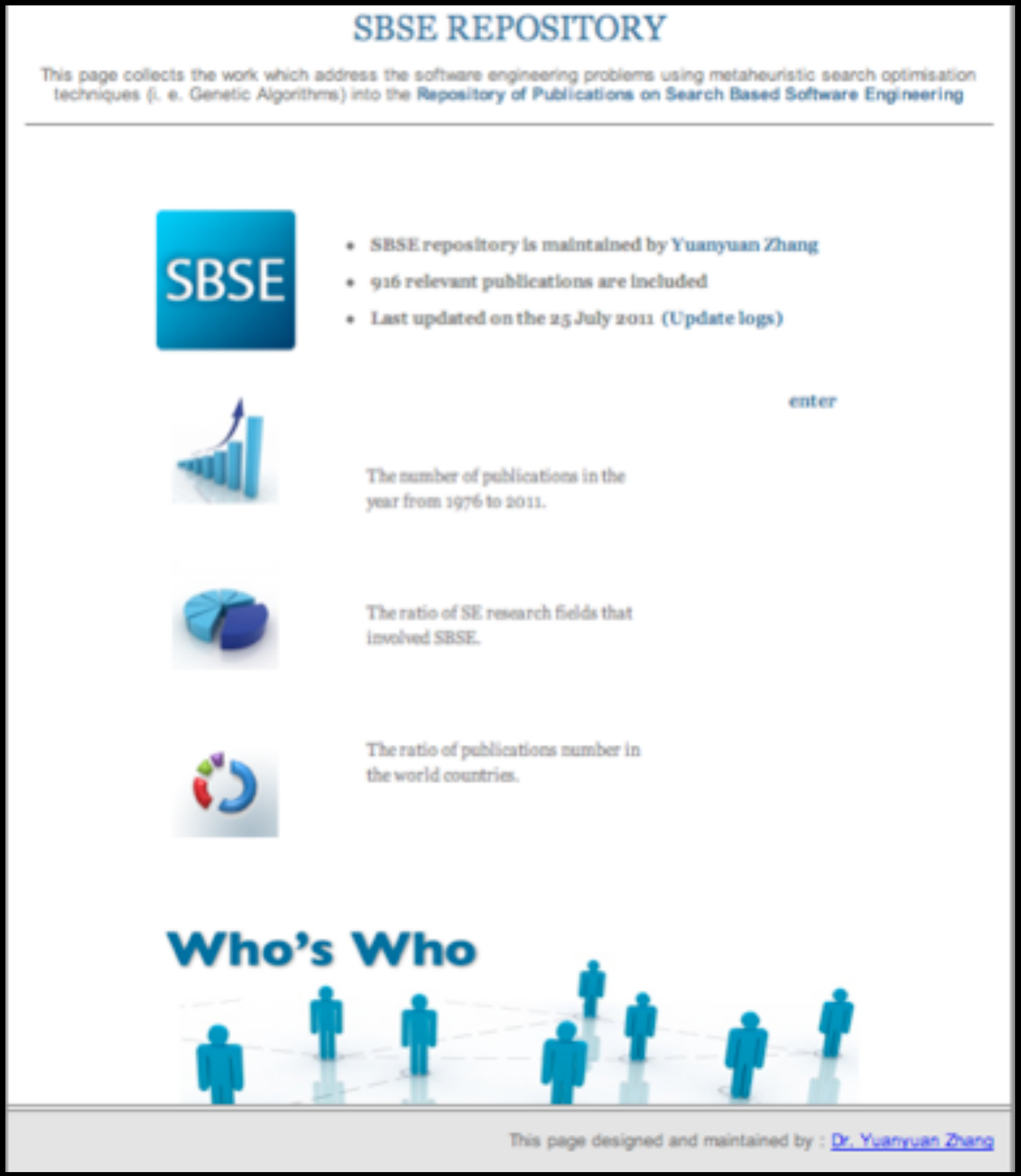
This page designed and maintained by : Dr. Yuanyuan Zhang

SBSE Repository

Analysis

extended who's who

Animations



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
- 916 relevant publications are included
- Last updated on the 25 July 2011 (Update logs)

enter

The number of publications in the year from 1976 to 2011.

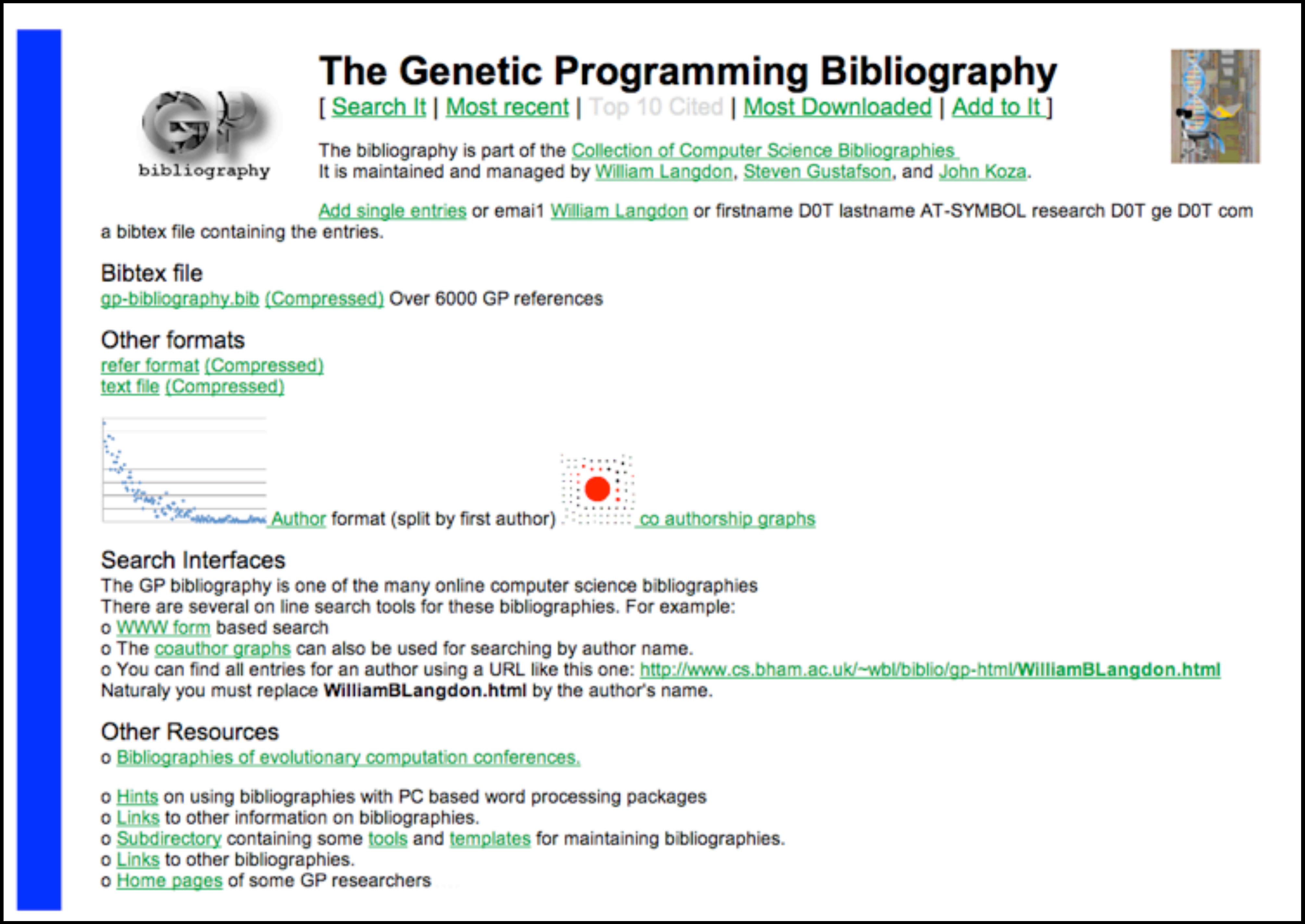
The ratio of SE research fields that involved SBSE.

The ratio of publications number in the world countries.

Who's Who

This page designed and maintained by : Dr. Yuanyuan Zhang

GP Bibliography



The Genetic Programming Bibliography

[Search It | Most recent | Top 10 Cited | Most Downloaded | Add to It]

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 emai1 William Langdon or firstname D0T lastname AT-SYMBOL research D0T ge D0T com a bibtex file containing the entries.

Bibtex file

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

Other formats

refer format (Compressed)
text file (Compressed)

Author format (split by first author) co authorship graphs

Search Interfaces

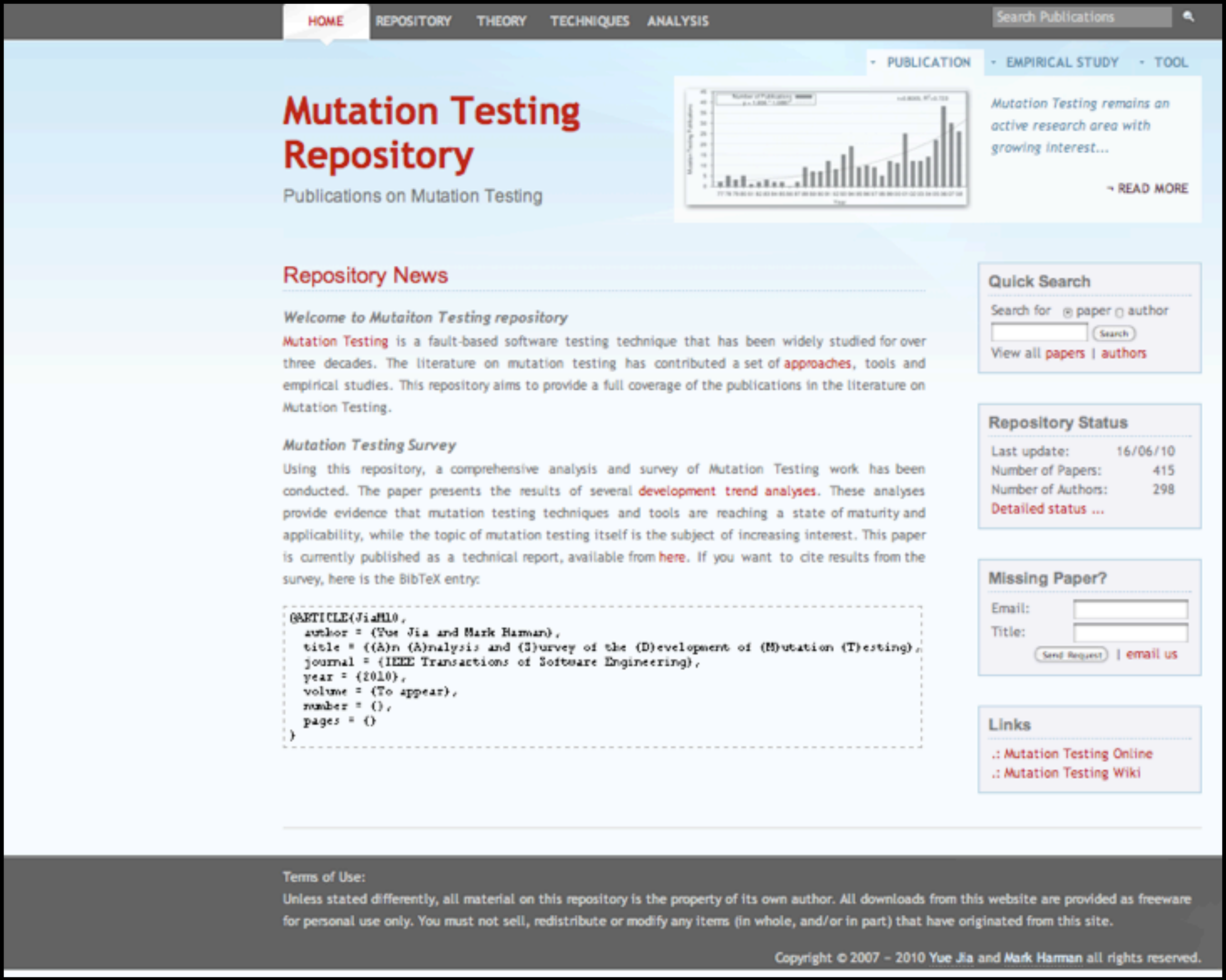
The GP bibliography is one of the many online computer science bibliographies
There are several on line search tools for these bibliographies. For example:
o WWW form based search
o The coauthor graphs can also be used for searching by author name.
o You can find all entries for an author using a URL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

Other Resources

o Bibliographies of evolutionary computation conferences.

o Hints on using bibliographies with PC based word processing packages
o Links to other information on bibliographies.
o Subdirectory containing some tools and templates for maintaining bibliographies.
o Links to other bibliographies.
o Home pages of some GP researchers

Mutation Testing Repository



Mutation Testing Repository

Publications on Mutation Testing

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

→ READ MORE

Repository News

Welcome to Mutaiton 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:

```
@ARTICLE{JiaH10,
  author = {Yue Jia and Mark Harman},
  title = {{A}n {A}nalysis and {S}urvey of the {D}evelopment of {M}utation {T}esting},
  journal = {IEEE Transactions of Software Engineering},
  year = {2010},
  volume = {To appear},
  number = {},
  pages = {}
}
```

Quick Search

Search for ● paper ○ author

Search

View all papers | authors

Repository Status

Last update:	16/06/10
Number of Papers:	415
Number of Authors:	298

Detailed status ...

Missing Paper?

Email:

Title:

Send Request | email us

Links

.: Mutation Testing Online
.: Mutation Testing Wiki

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.

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

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

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: 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

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: 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 SBSE

Upcoming COWs

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

What is SBSE

What is SBSE

let’s listen to software engineers ...

... what sort of things do they say?

Software Engineers Say

Software Engineers Say

Software Engineers Say

We need to satisfy business and technical concerns

We need to reduce risk while maintaining completion time

We need increased cohesion and decreased coupling

We need fewer tests that find more nasty bugs

We need to optimise for all metrics M1,..., Mn

Software Engineers Say

We need to satisfy business and technical concerns

We need to reduce risk while maintaining completion time

We need increased cohesion and decreased coupling

We need fewer tests that find more nasty bugs

We need to optimise for all metrics M1,..., Mn

Software Engineers Say

We need to satisfy business and technical concerns

We need to reduce risk while maintaining completion time

We need increased cohesion and decreased coupling

We need fewer tests that find more nasty bugs

We need to optimise for all metrics M1,..., Mn

Software Engineers Say

We need to satisfy business and technical concerns

We need to reduce risk while maintaining completion time

We need increased cohesion and decreased coupling

We need fewer tests that find more nasty bugs

We need to optimise for all metrics M1,..., Mn

Software Engineers Say

We need to satisfy business and technical concerns

We need to reduce risk while maintaining completion time

We need increased cohesion and decreased coupling

We need fewer tests that find more nasty bugs

We need to optimise for all metrics M1,..., Mn

Software Engineers Say

We need to satisfy business and technical concerns

We need to reduce risk while maintaining completion time

We need increased cohesion and decreased coupling

We need fewer tests that find more nasty bugs

We need to optimise for all metrics M1,..., Mn

Software Engineers Say

We need to satisfy business and technical concerns

We need to reduce risk while maintaining completion time

We need increased cohesion and decreased coupling

We need fewer tests that find more nasty bugs

We need to optimise for all metrics M1,..., Mn

Software Engineers Say

We need to satisfy business and technical concerns

We need to reduce risk while maintaining completion time

We need increased cohesion and decreased coupling

We need fewer tests that find more nasty bugs

We need to optimise for all metrics M1,..., Mn

Software Engineers Say

Requirements: We need to satisfy business and technical concerns

Management: We need to reduce risk while maintaining completion time

Design: We need increased cohesion and decreased coupling

Testing: We need fewer tests that find more nasty bugs

Refactoring: We need to optimise for all metrics M1,..., Mn

Software Engineers Say

Requirements: We need to satisfy business and technical concerns

Management: We need to reduce risk while maintaining completion time

Design: We need increased cohesion and decreased coupling

Testing: We need fewer tests that find more nasty bugs

Refactoring: We need to optimise for all metrics M1,..., Mn

All have been addressed in the SBSE literature

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.

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.

like google search?
like code search?
like breadth first search?

potentially
exhaustive

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?

potentially
exhaustive

pick one at
random

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?

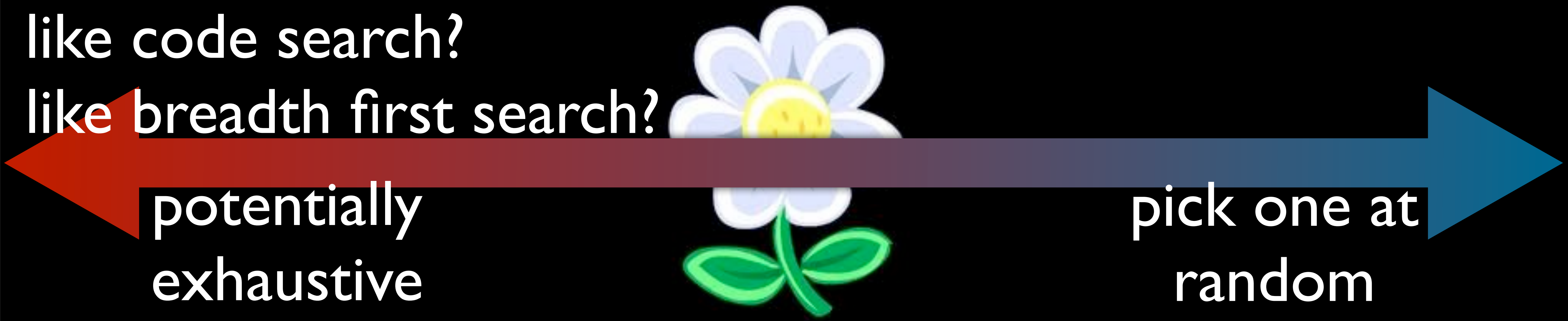
potentially exhaustive

pick one at random

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?



potentially exhaustive

pick one at random

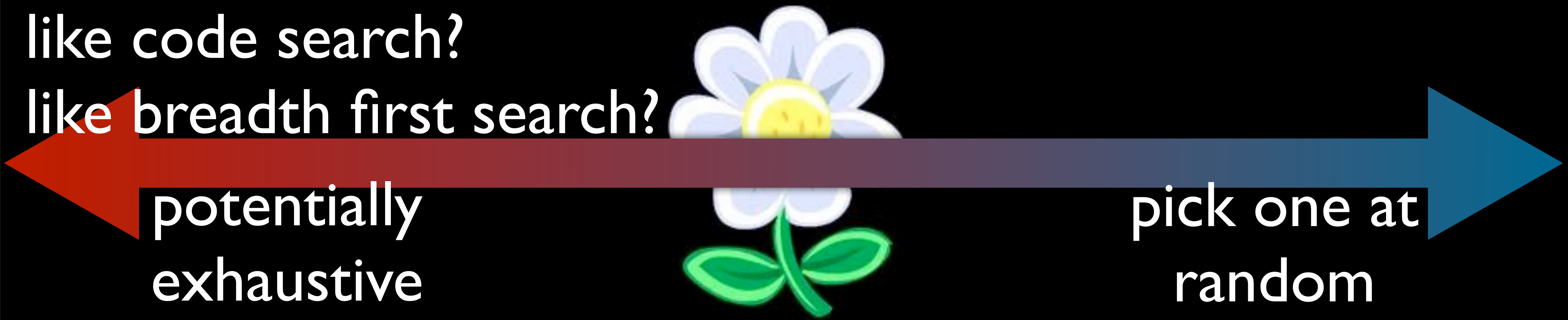
What is SBSE

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?



sweet spot

potentially exhaustive

pick one at random

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

Particle Swarm Optimization

Genetic Algorithms

Hill Climbing

Genetic Programming

Simulated Annealing

Greedy

LP

Random

Estimation of Distribution Algorithms

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

Particle Swarm Optimization

Genetic Algorithms

Hill Climbing

Genetic Programming

Simulated Annealing

Greedy

LP

Random

Estimation of Distribution Algorithms

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

Particle Swarm Optimization

Genetic Algorithms

Hill Climbing

Genetic Programming

Simulated Annealing

Random

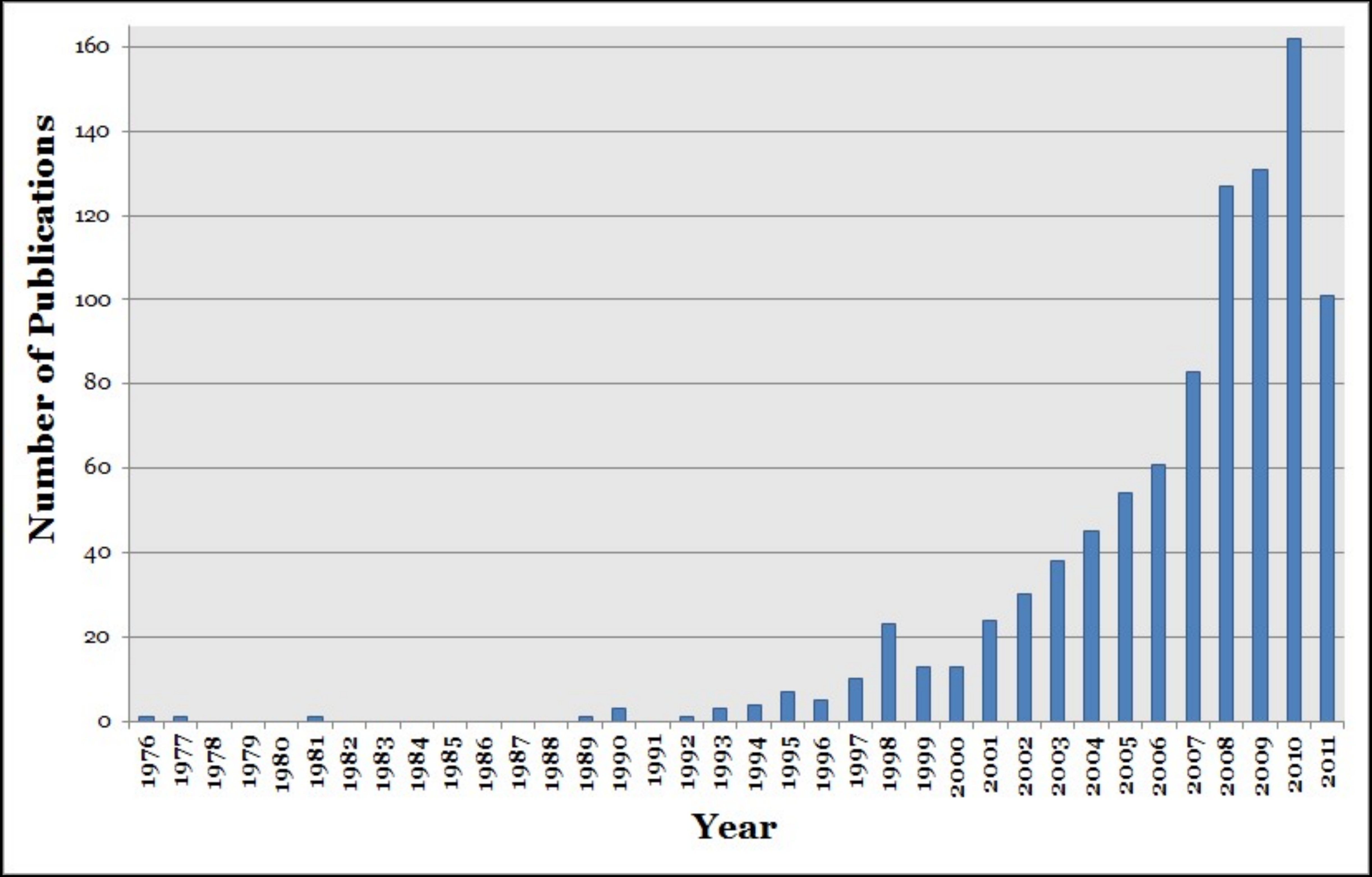
Greedy

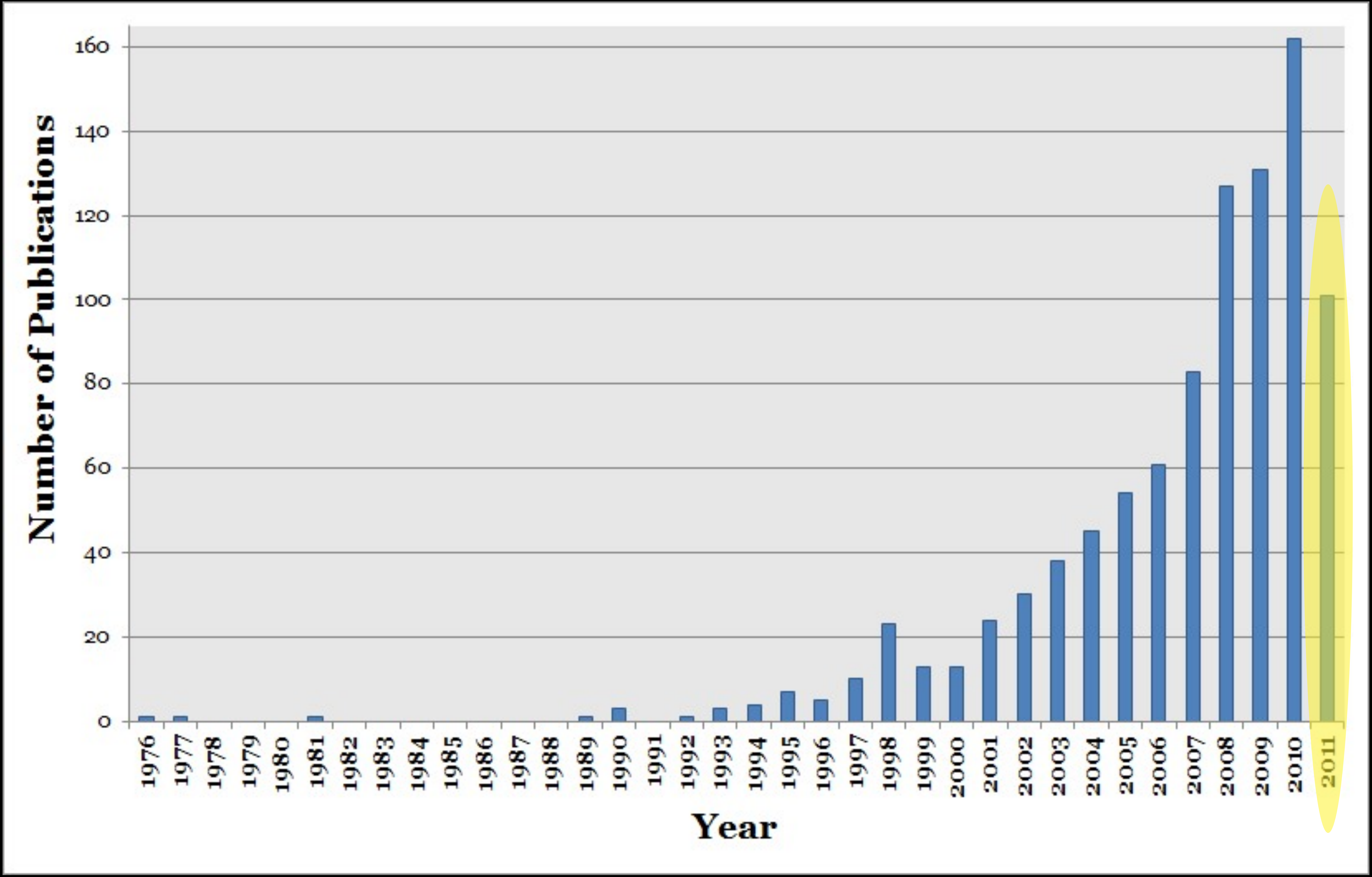
LP

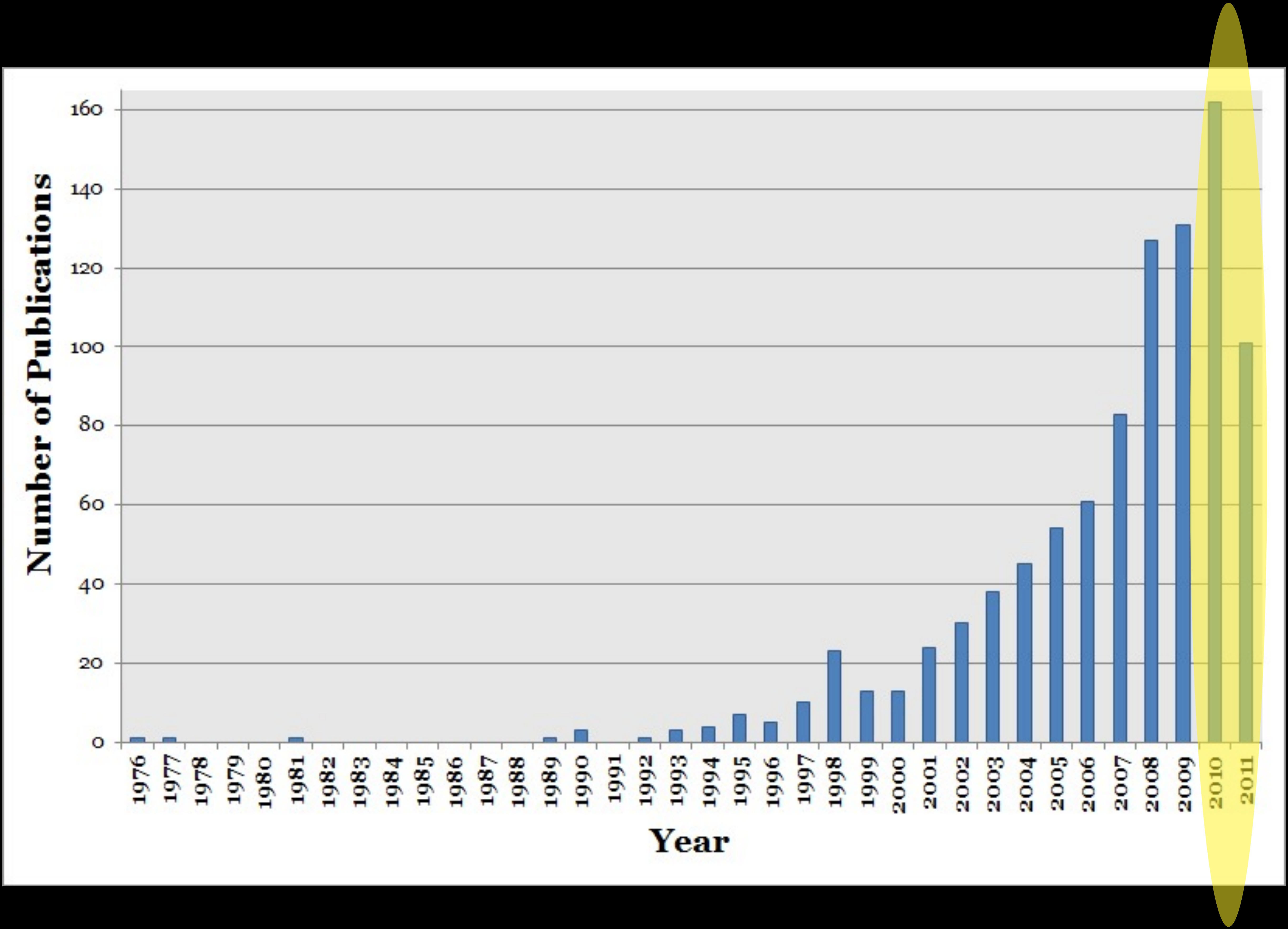
Estimation of Distribution Algorithms

Wes Weimer, ThanVu Nguyen, Claire Le Goues, Stephanie Forrest.
Automatically Finding Patches Using Genetic Programming. ICSE 2009 best paper.

Growth Trends


Number of Publications
160
140
120
100
80
60
40
20
0
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
Year


Number of Publications
0
20
40
60
80
100
120
140
160
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
Year


Number of Publications
160
140
120
100
80
60
40
20
0
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
Year

Essential Ingredients for SBSE

Essential Ingredients for SBSE

Representation

Essential Ingredients for SBSE

Representation

Fitness Function

Essential Ingredients for SBSE

Representation

Fitness Function

The advantages of SBSE

The advantages of SBSE

The advantages of SBSE

Insight-rich

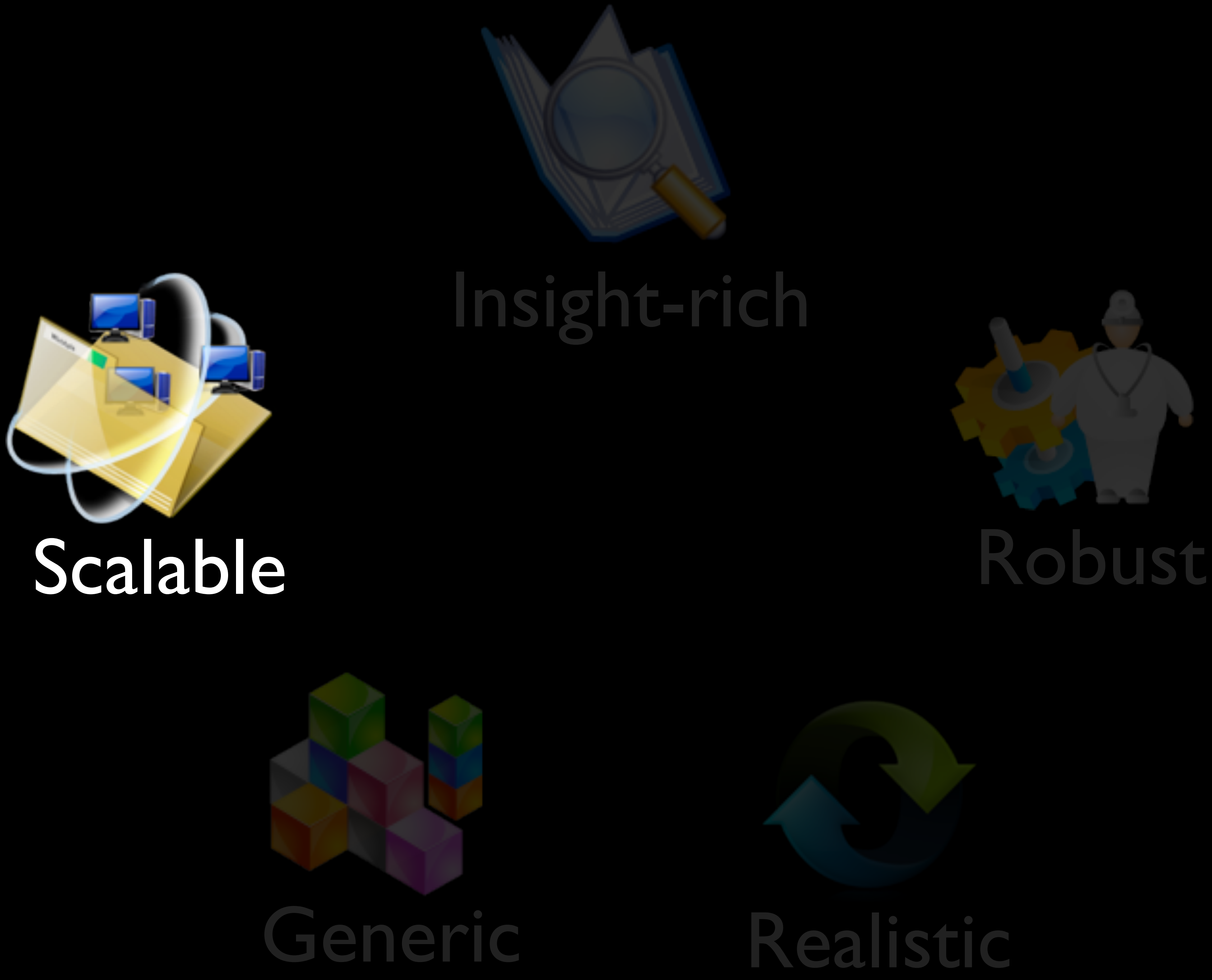
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Insight-rich

Scalable

Robust

Generic

Realistic

The advantages of SBSE

Insight-rich

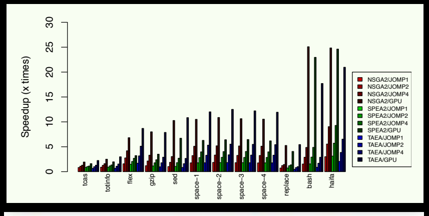
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Insight-rich

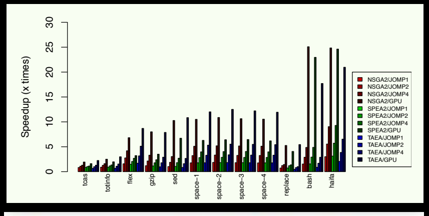
Scalable

Robust

Generic

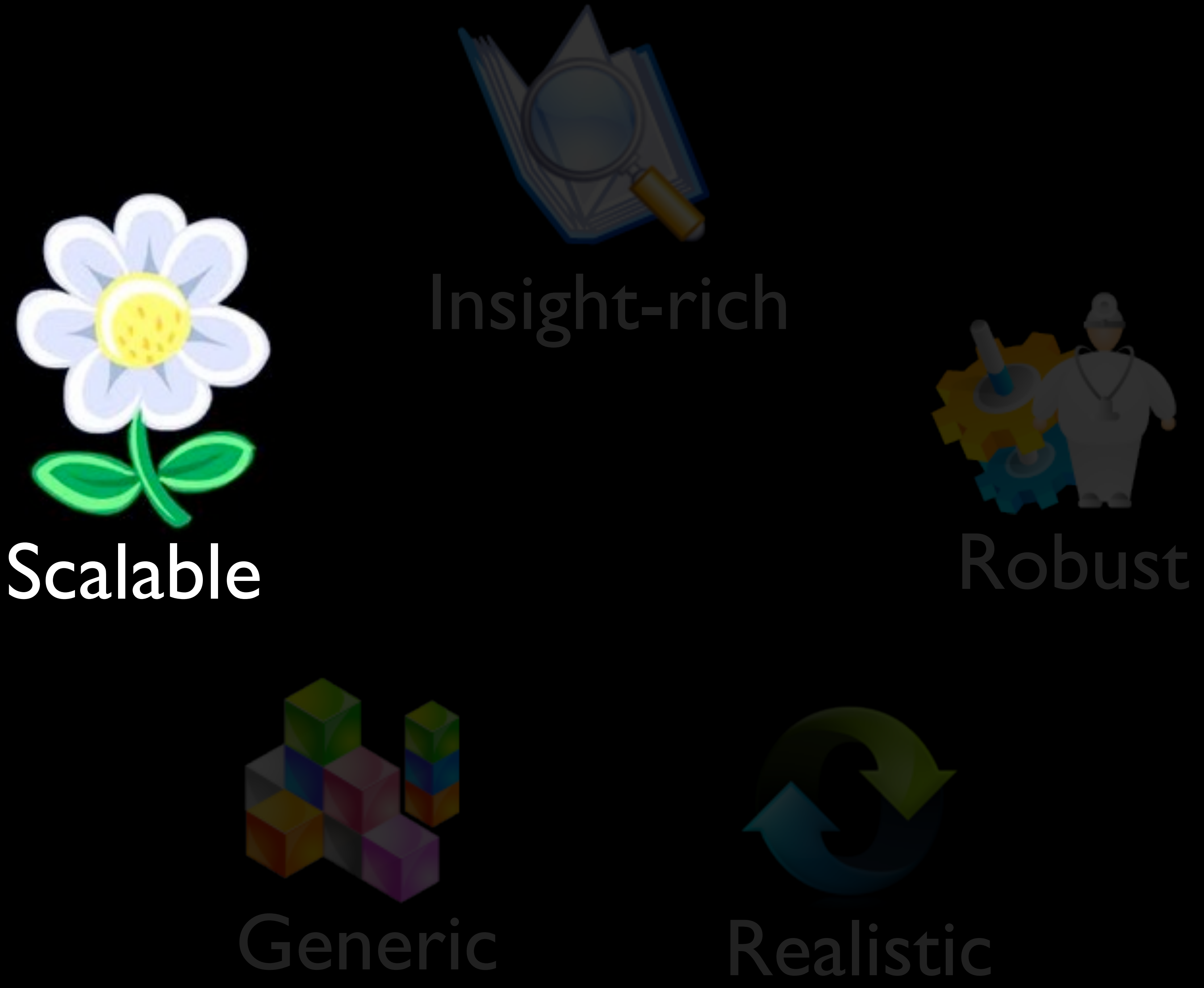
Realistic

The advantages of SBSE



IBM®

Shin Yoo



Scalable

Insight-rich

Robust

Generic

Realistic

The advantages of SBSE

Insight-rich

Scalable

Robust

Generic

Realistic

The advantages of SBSE

Insight-rich

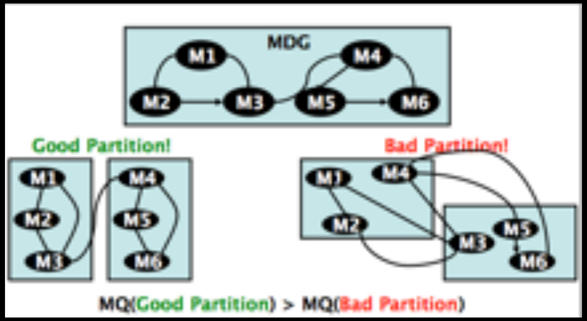
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Insight-rich

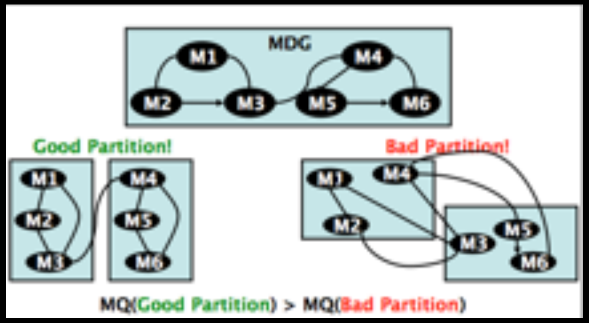
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Márcio Barros

Insight-rich

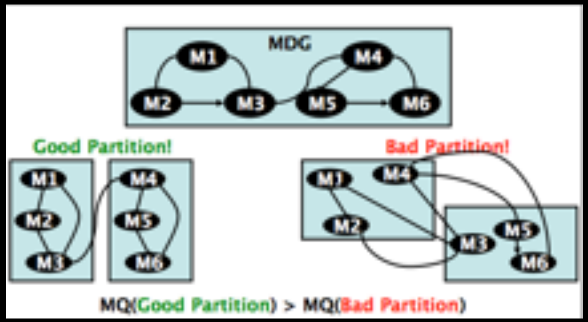
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Márcio Barros

Insight-rich

$$\operatorname*{avg}_{y \in N(x)}\{f(y)\} = \frac{1}{|N(x)|} \sum_{y \in N(x)} f(y)$$

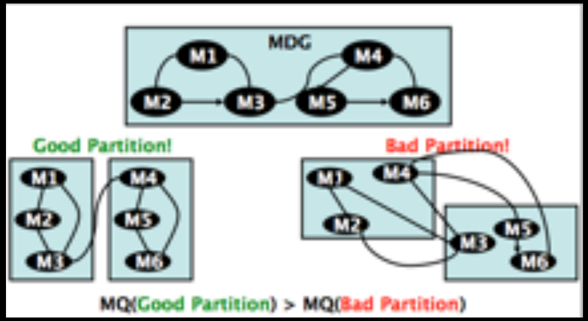
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Márcio Barros

$$\operatorname{avg}_{y \in N(x)}\{f(y)\} = \frac{1}{|N(x)|} \sum_{y \in N(x)} f(y)$$

Javier Ferrer

Insight-rich

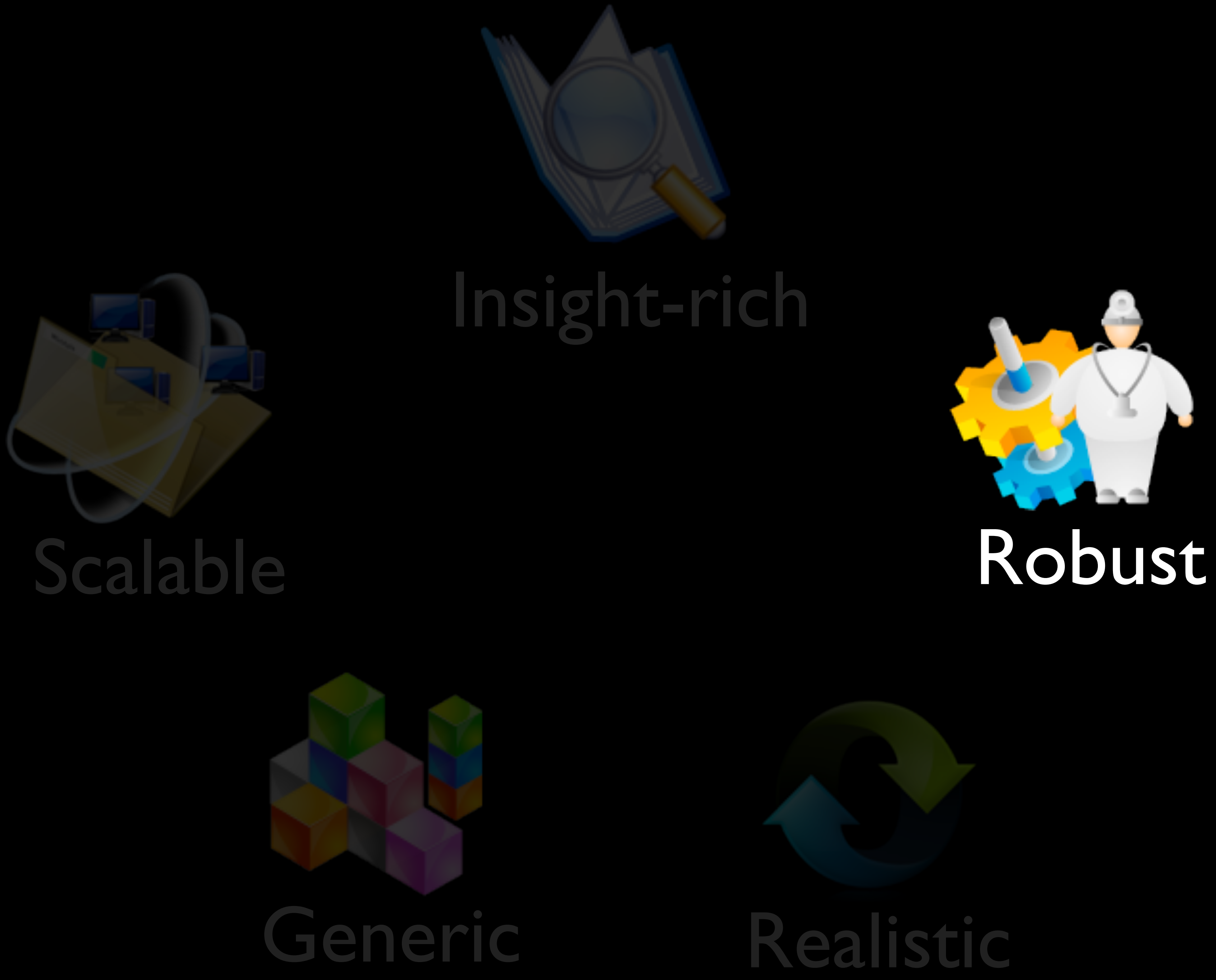
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Insight-rich

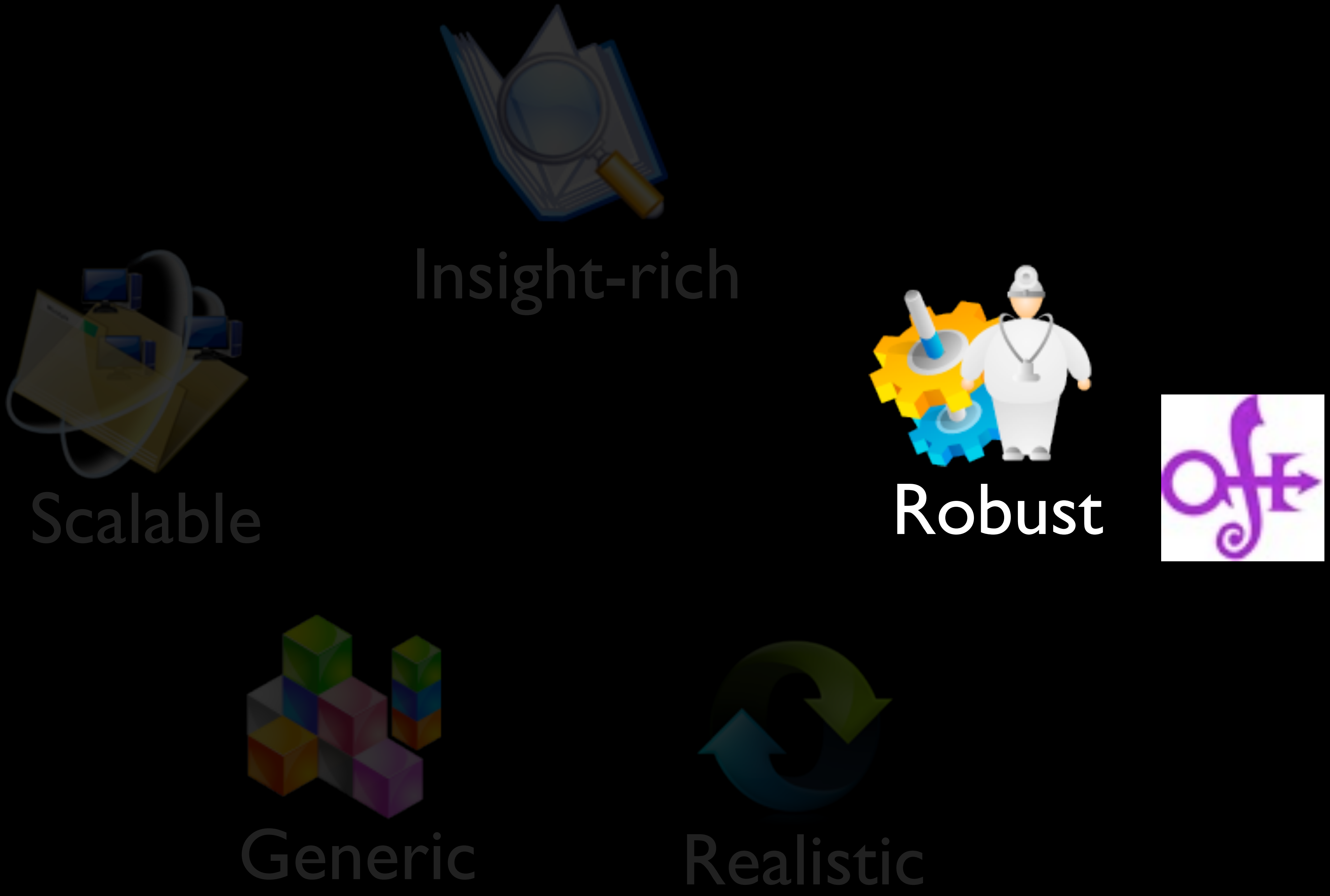
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Insight-rich

Scalable

Robust

Generic

Realistic

The advantages of SBSE

Insight-rich

Scalable

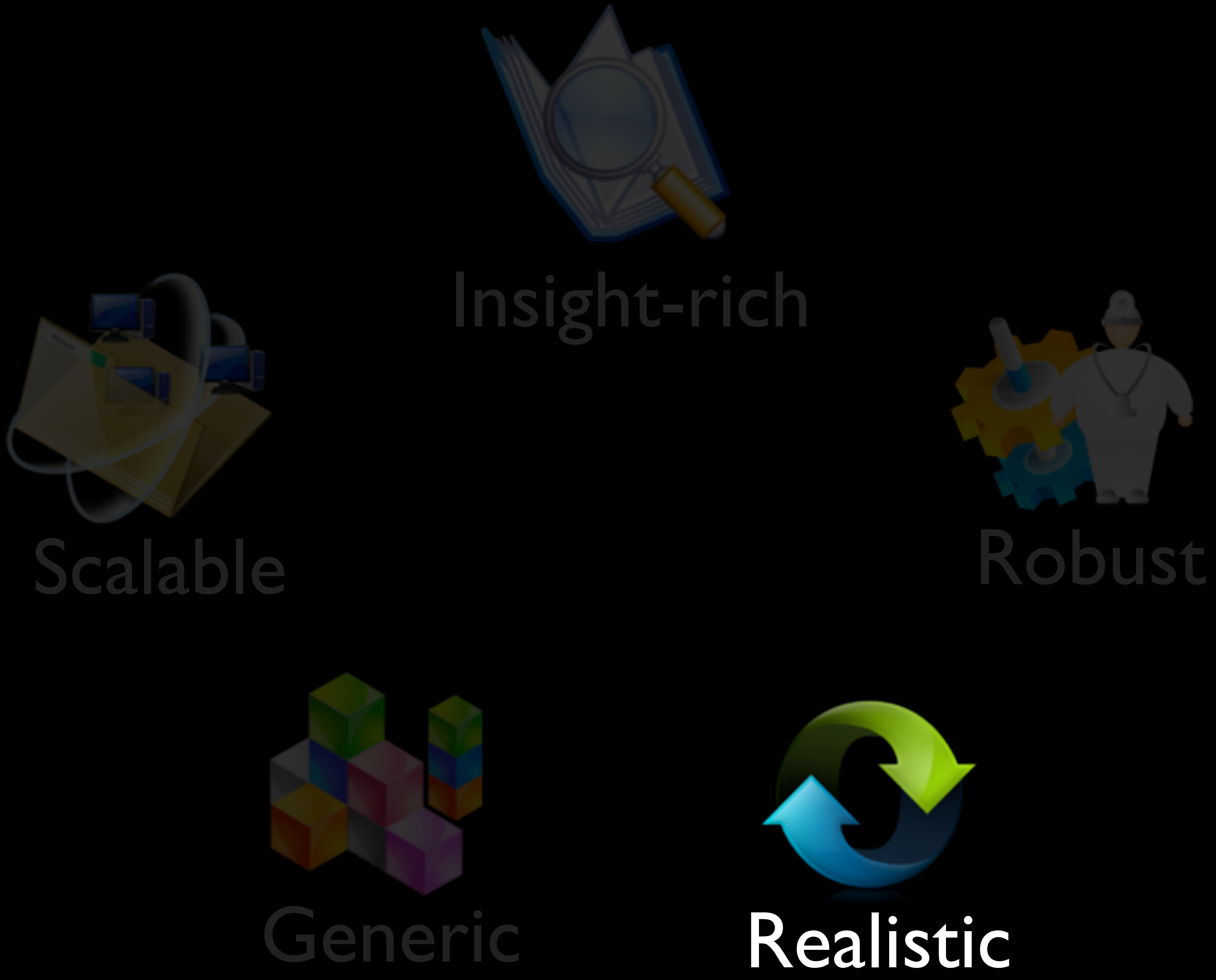
Robust

Gay, Menzies et al.

Generic

Realistic

The advantages of SBSE



Insight-rich

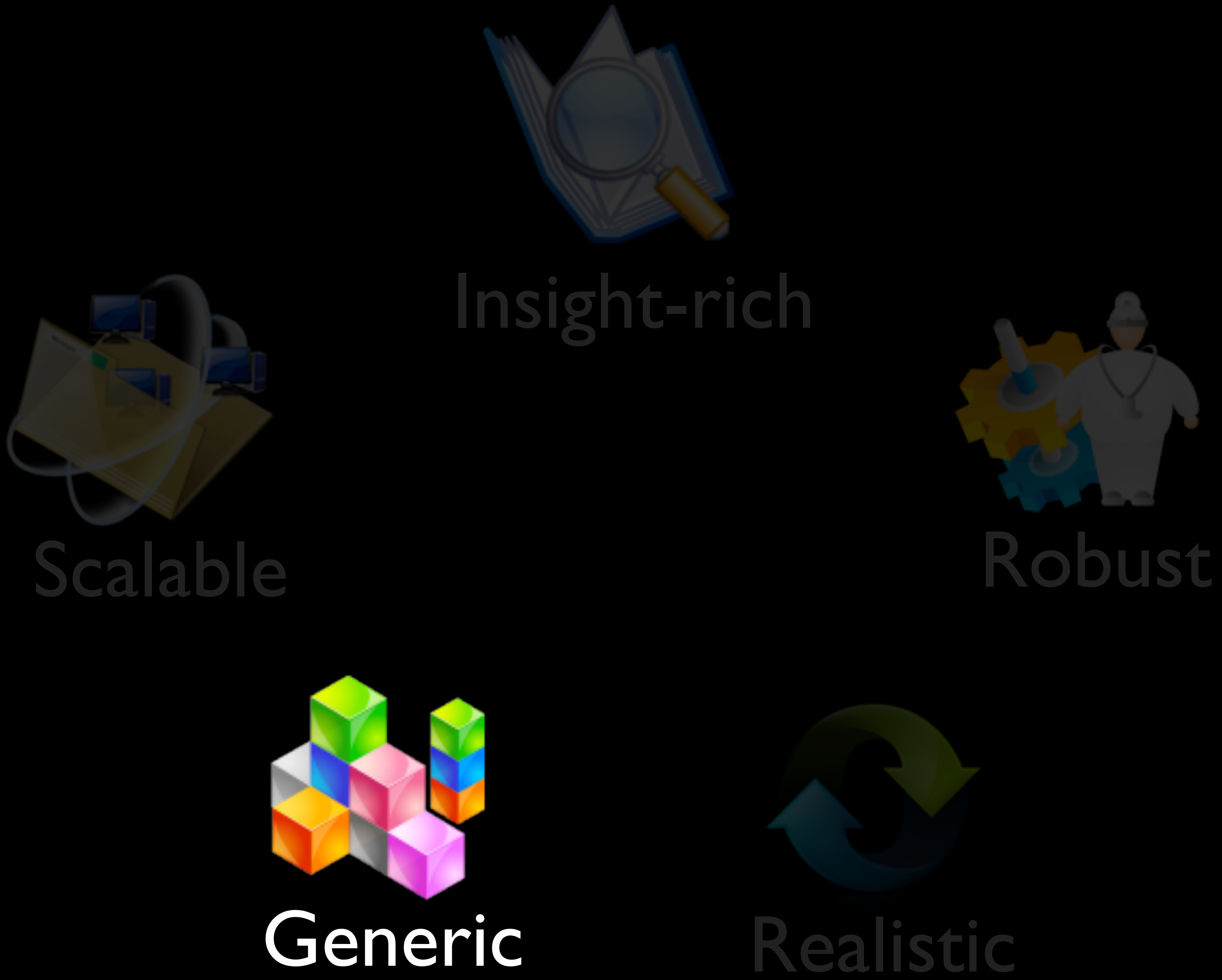
Scalable

Robust

Generic

Realistic

The advantages of SBSE



Insight-rich

Scalable

Robust

Generic

Realistic

Software Engineers Say

Requirements: We need to satisfy business and technical concerns

Management: We need to reduce risk while maintaining completion time

Design: We need increased cohesion and decreased coupling

Testing: We need fewer tests that find more nasty bugs

Refactoring: We need to optimise for all metrics M1,..., Mn

All have been addressed in the SBSE literature

Software Engineers Say

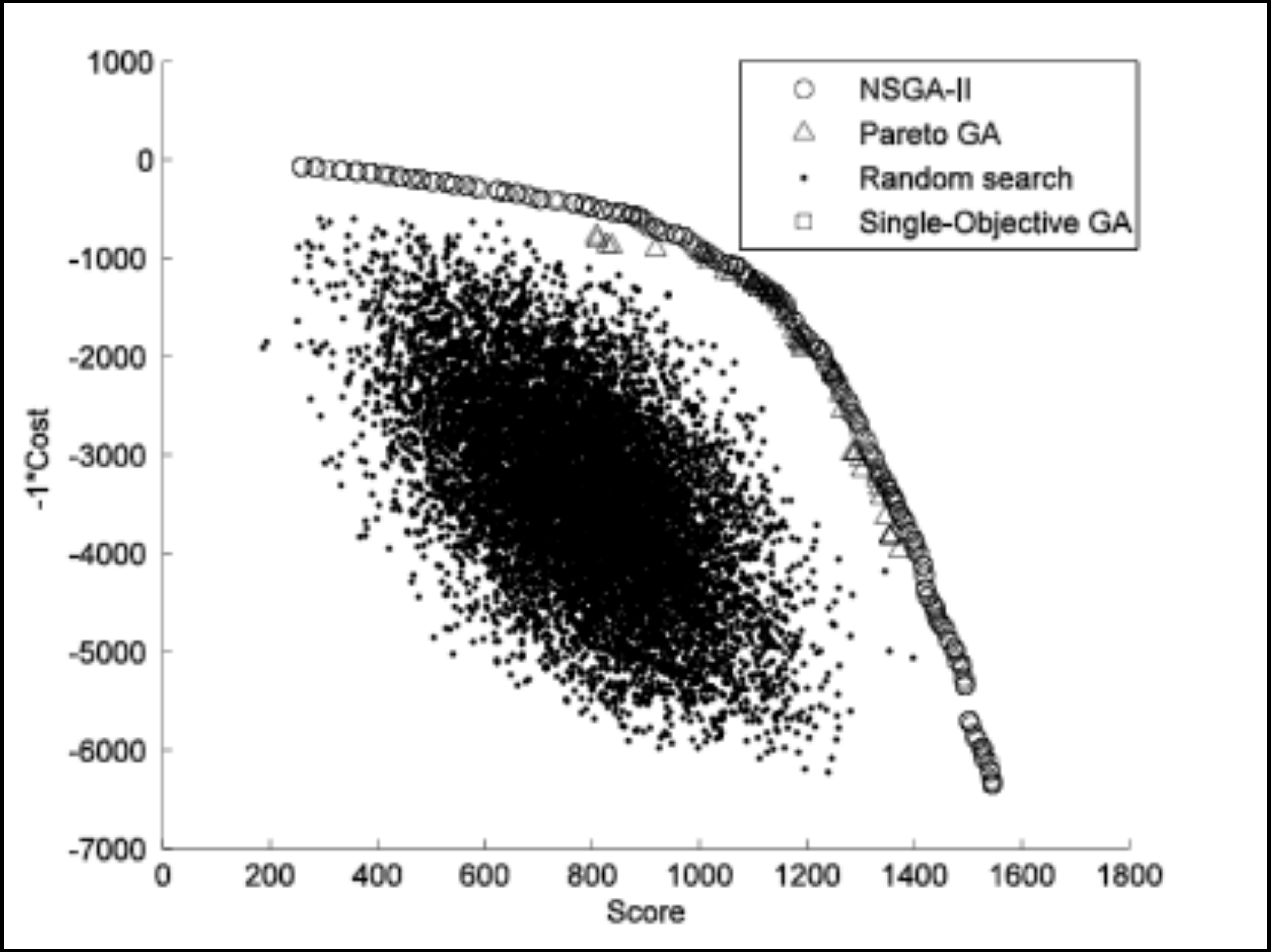
Requirements: We need to satisfy business and technical concerns

Refactoring: We need to optimise for all metrics M1,..., Mn

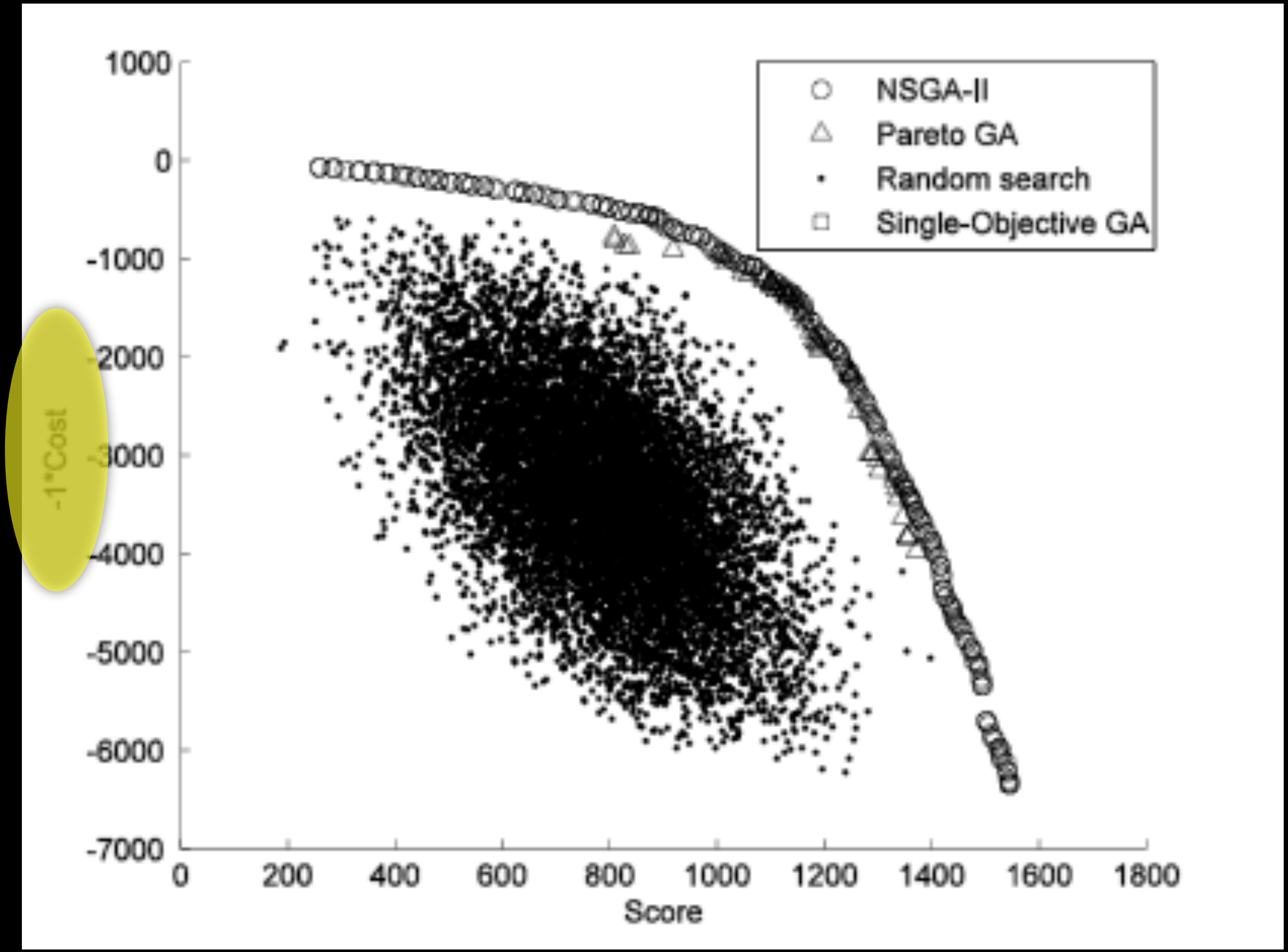
quick look at these two ...

SBSE for Requirements

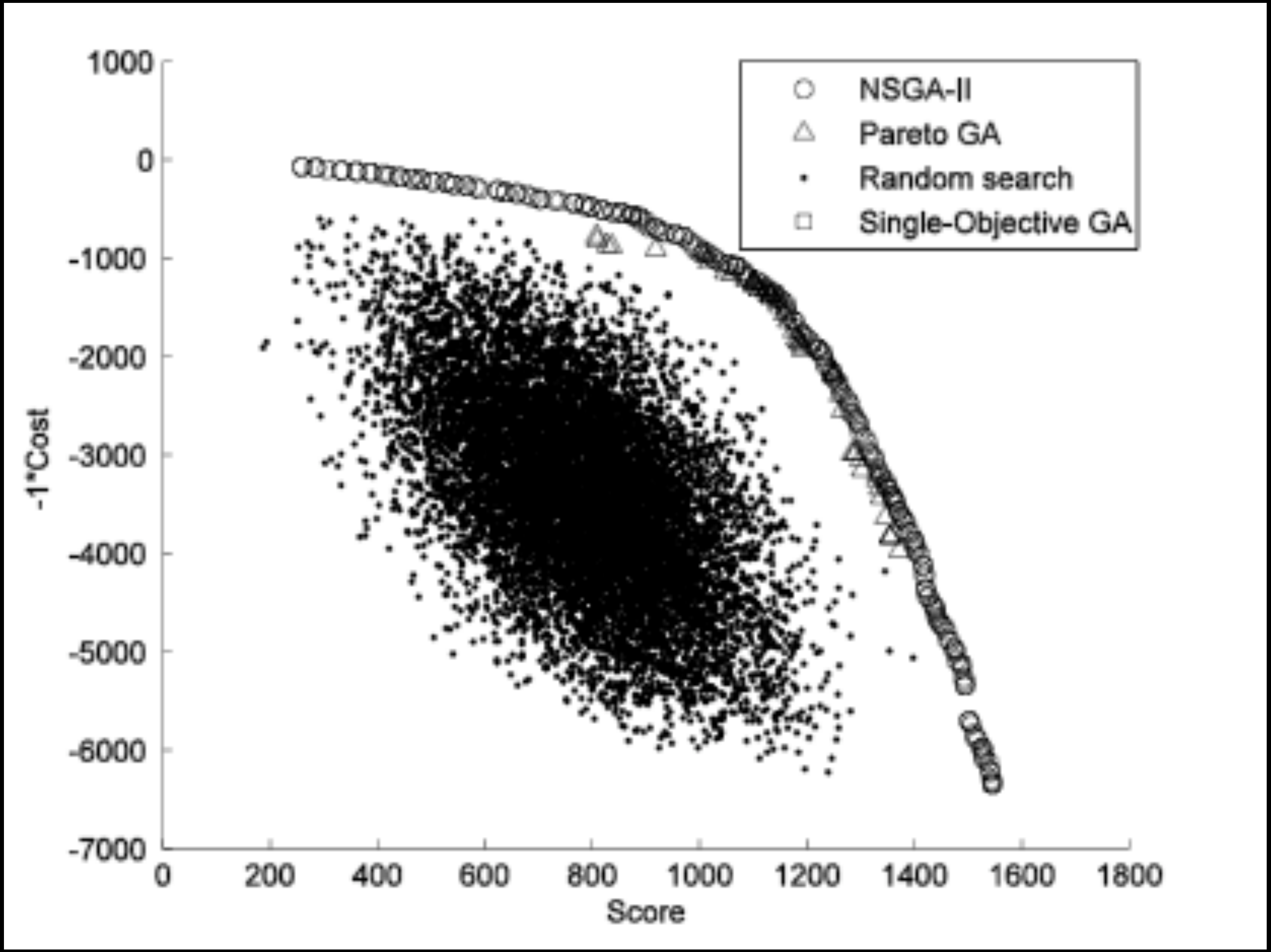
Motorola Cell Phone Requirements



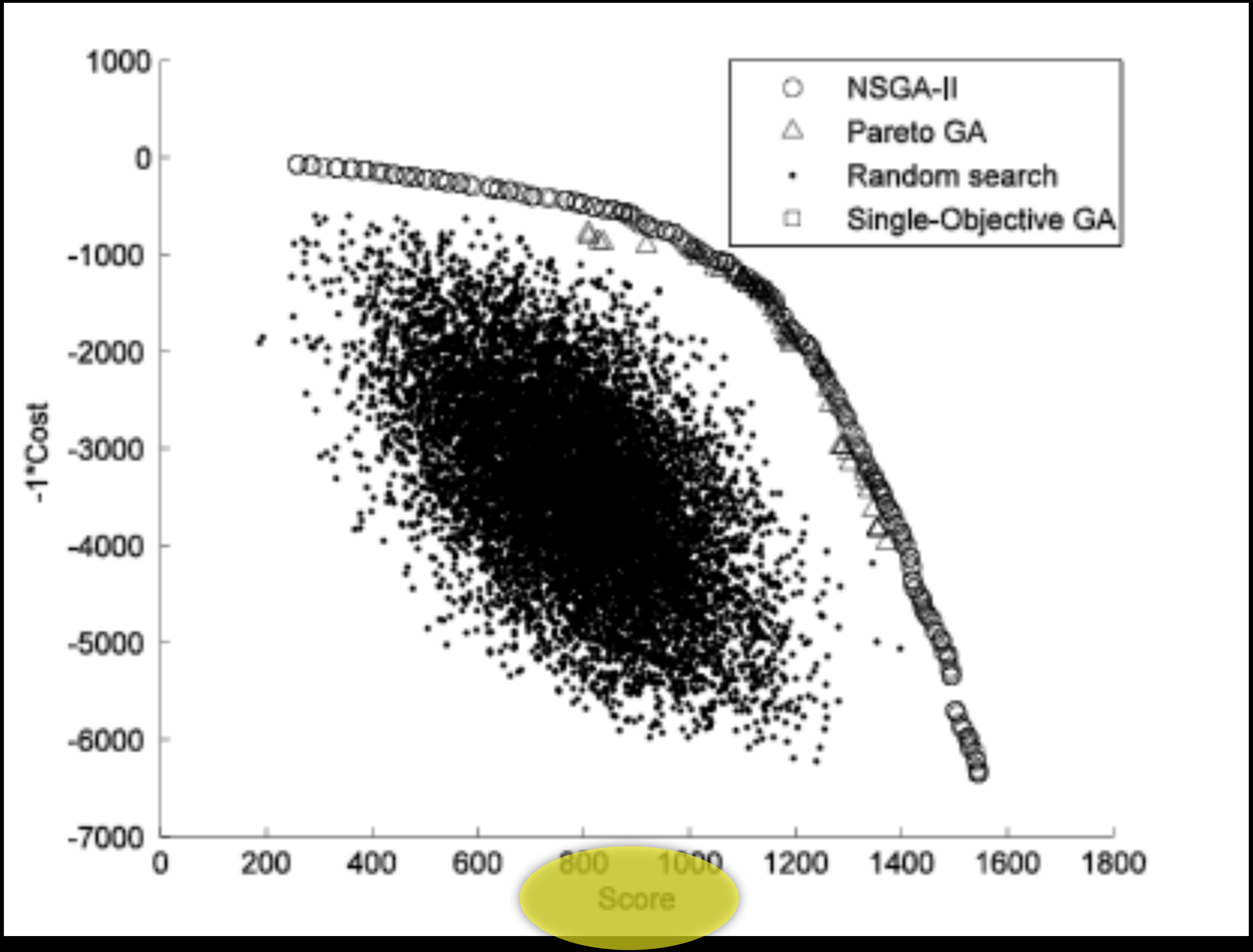
Motorola Cell Phone Requirements



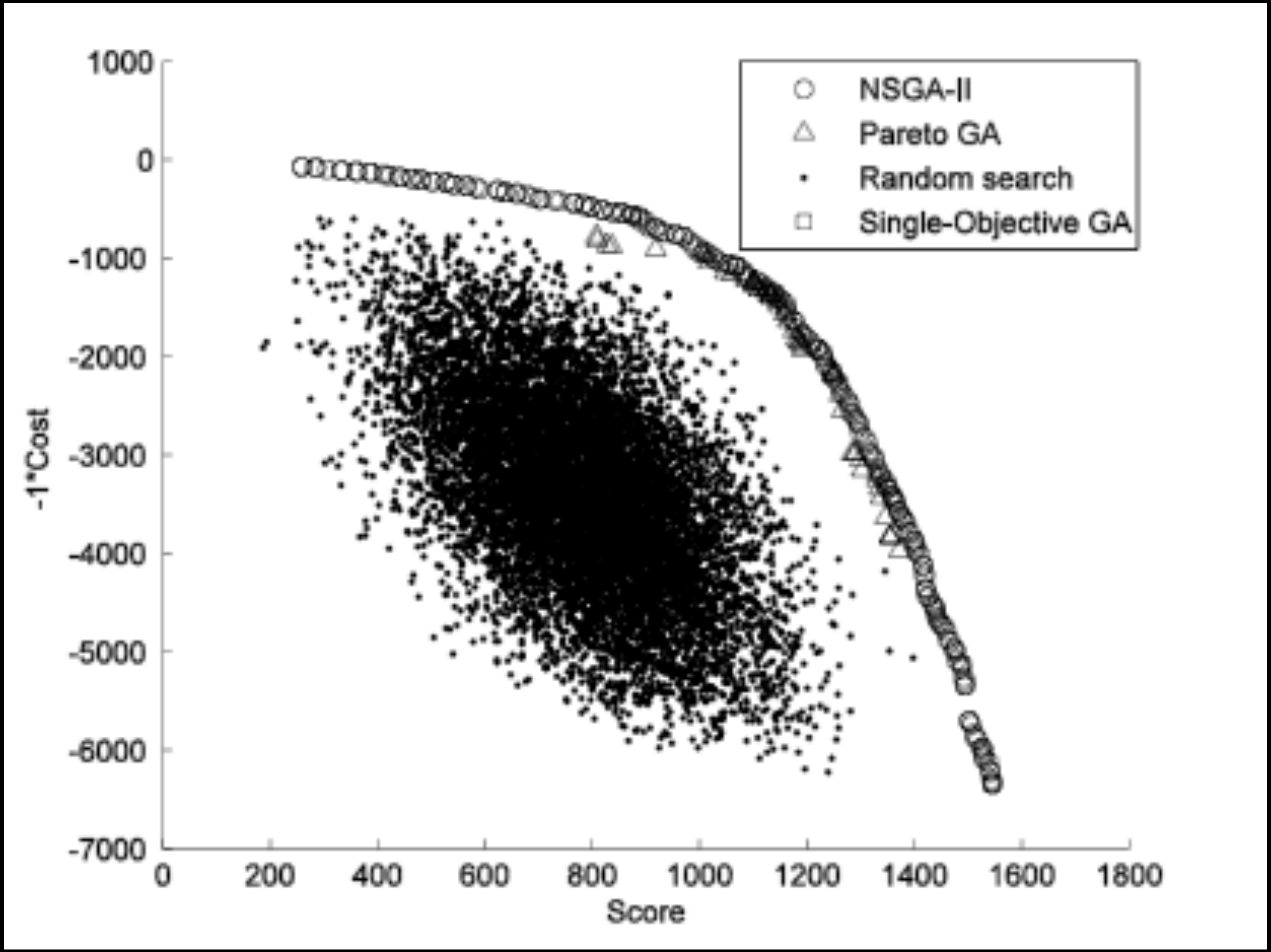
Motorola Cell Phone Requirements



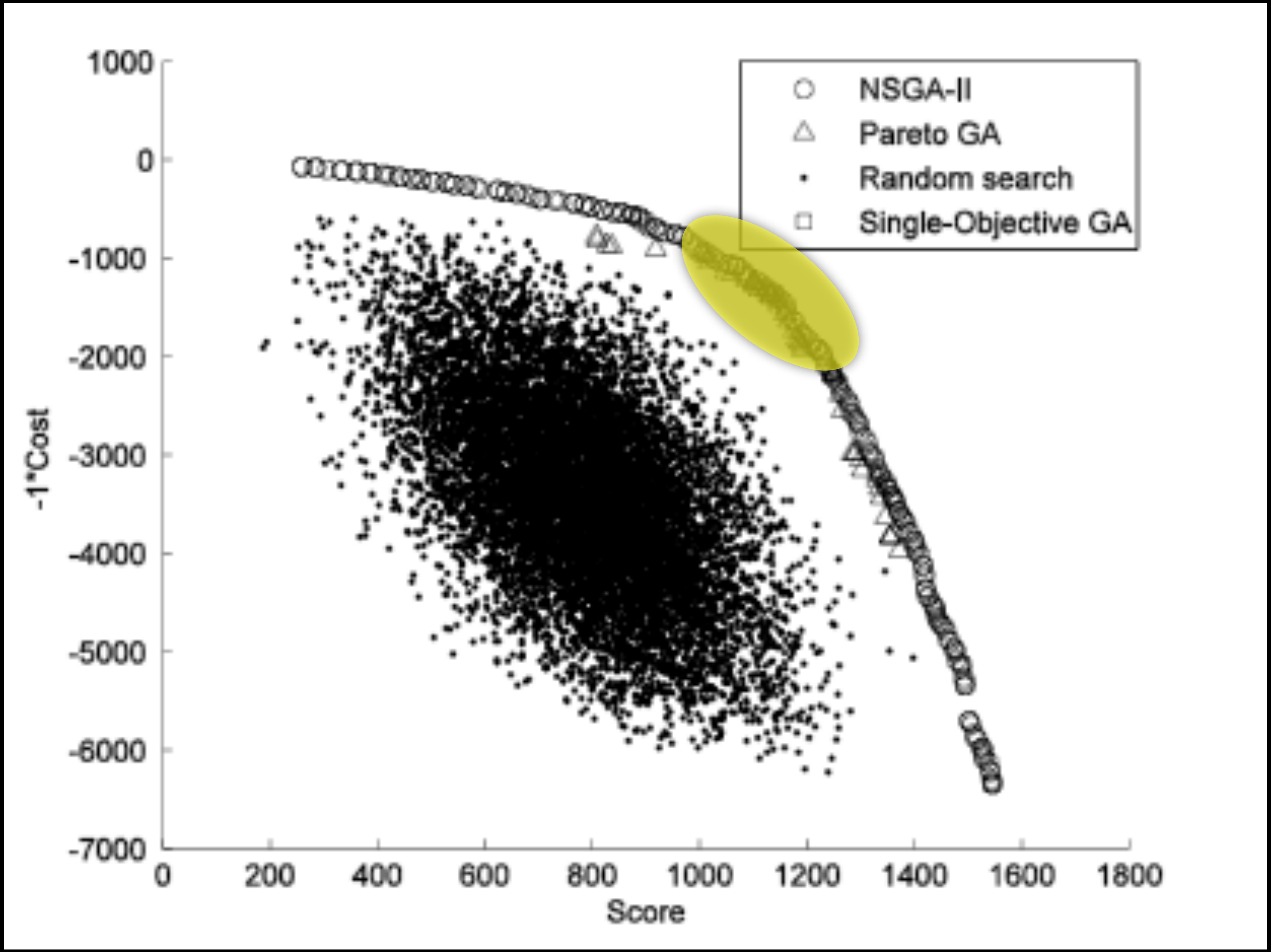
Motorola Cell Phone Requirements



Motorola Cell Phone Requirements



Motorola Cell Phone Requirements



SBSE for Regression Testing

Growth Trends

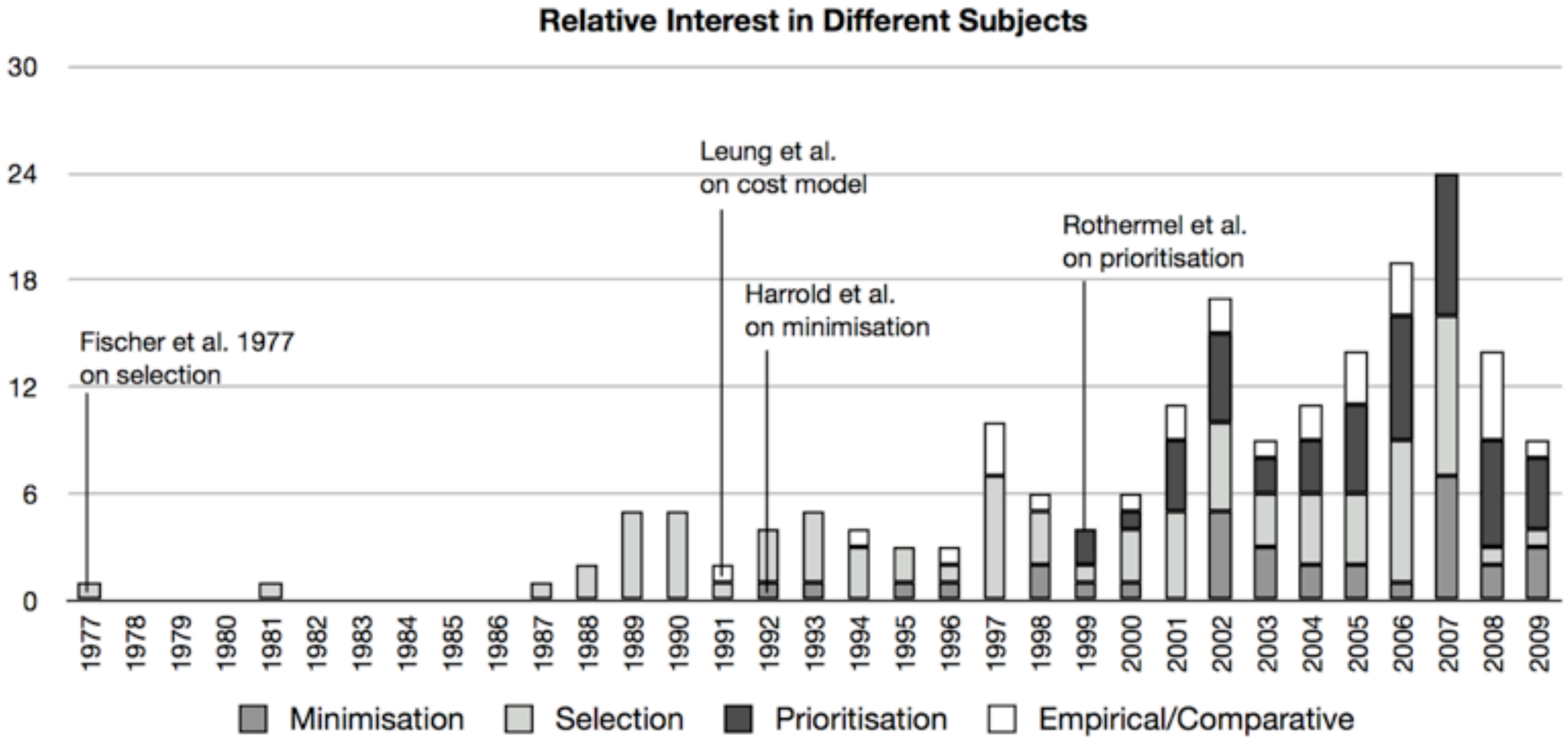


Figure 3. Relative research interest in each subject. Papers that consider more than one subject were counted multiple times.

Taken from forthcoming STVR survey
by Shin Yoo and Mark Harman

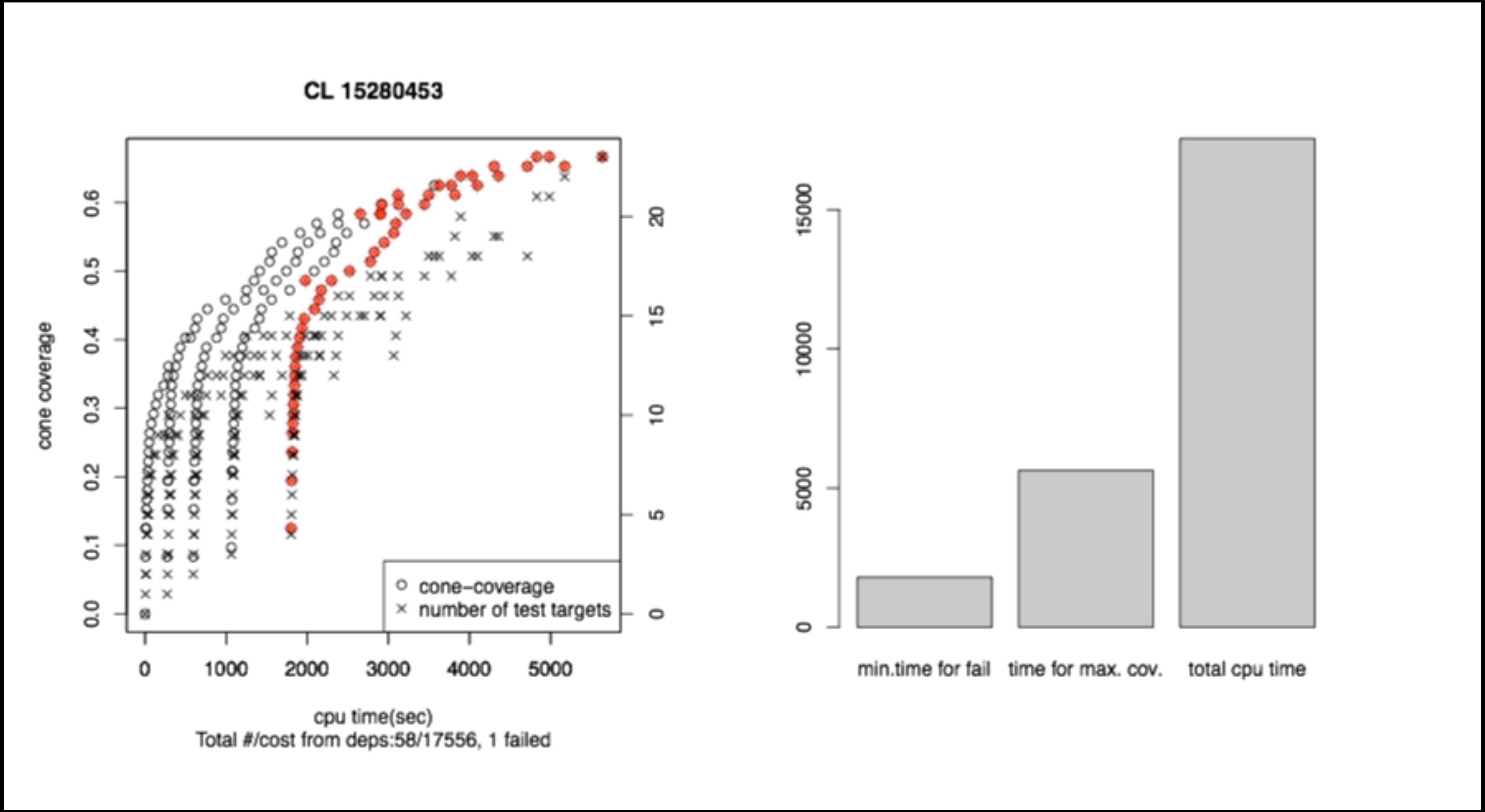
Recently at FSE ...

industry track

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)



Realism

Multi Objective SBSE:

Mark Harman

The Current State and Future of Search Based Software Engineering.

ICSE Future of Software Engineering: 342-357, 2007

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

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

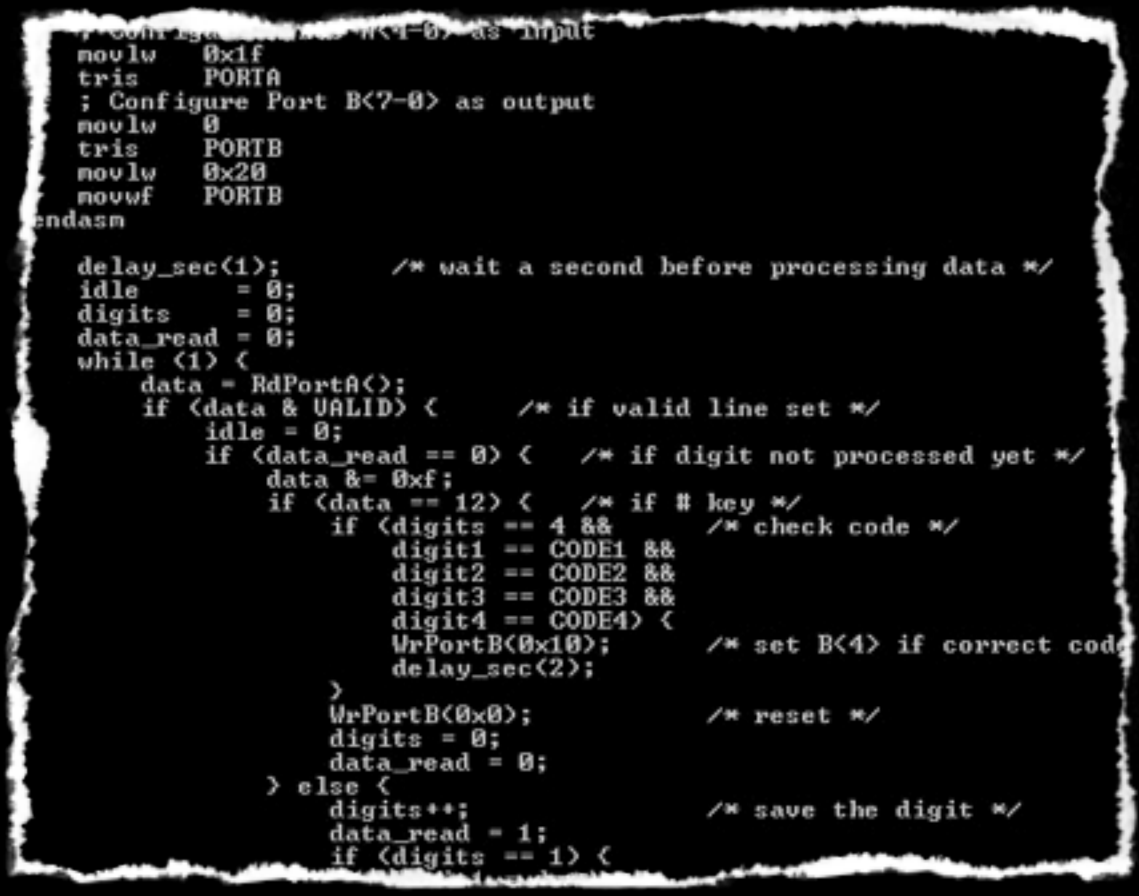
MORTO agenda

Cost

Value

Constraints

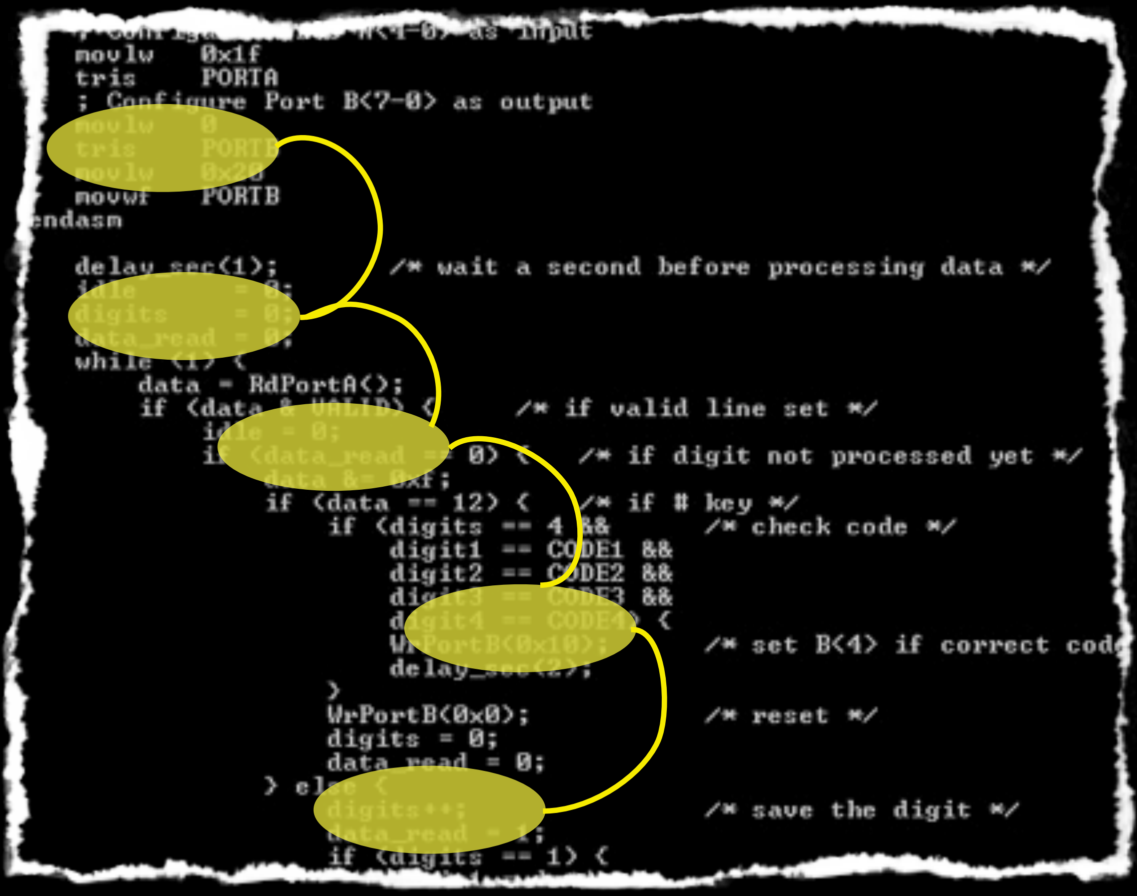
Value Objectives



```
  ; Configure Port A<4-0> as input
  movlw   0x1f
  tris    PORTA
  ; Configure Port B<7-0> as output
  movlw   0
  tris    PORTB
  movlw   0x20
  movwf   PORTB
endasm

  delay_sec(1);        /* wait a second before processing data */
  idle      = 0;
  digits    = 0;
  data_read = 0;
  while (1) {
      data = RdPortA();
      if (data & VALID) {      /* if valid line set */
          idle = 0;
          if (data_read == 0) {   /* if digit not processed yet */
              data &= 0xf;
              if (data == 12) {   /* if # key */
                  if (digits == 4 &&       /* check code */
                      digit1 == CODE1 &&
                      digit2 == CODE2 &&
                      digit3 == CODE3 &&
                      digit4 == CODE4) {
                      WrPortB(0x10);       /* set B<4> if correct code
                      delay_sec(2);
                  }
                  WrPortB(0x0);            /* reset */
                  digits = 0;
                  data_read = 0;
              } else {
                  digits++;                /* save the digit */
                  data_read = 1;
                  if (digits == 1) {
```

Value Objectives

Code Coverage

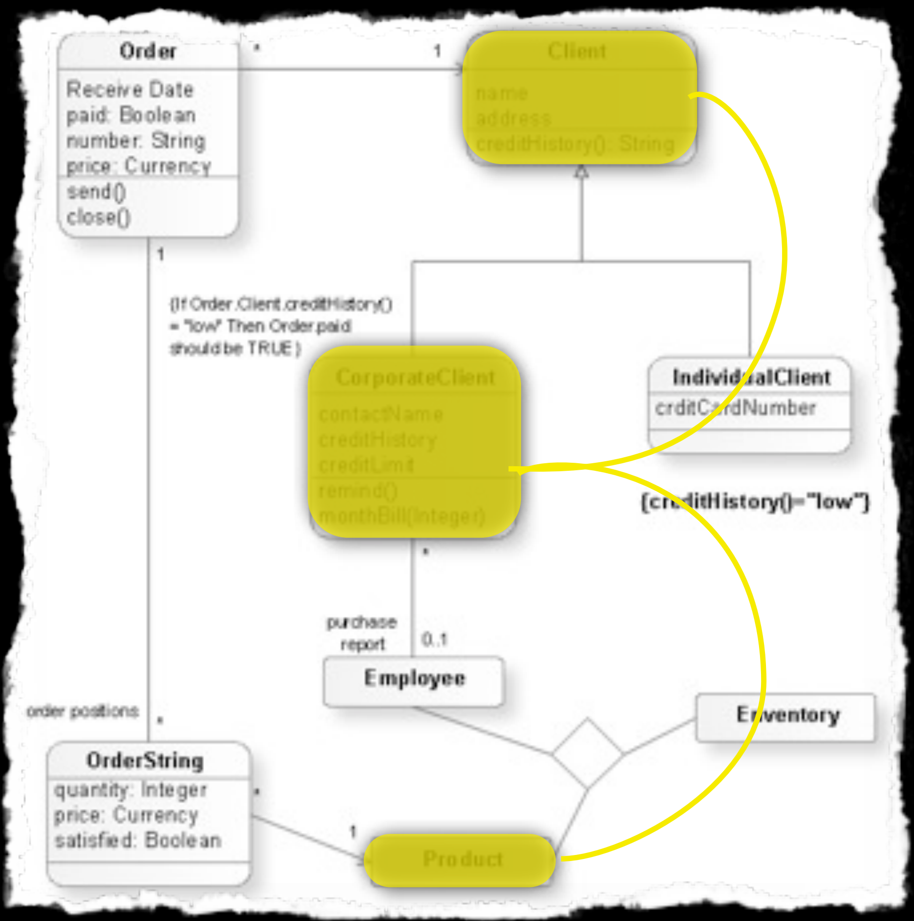


```
; Configure Port A<4-0> as input
movlw   0x1f
tris    PORTA
; Configure Port B<7-0> as output
movlw   0
tris    PORTB
movlw   0x20
movwf   PORTB
endasm

delay_sec(1);           /* wait a second before processing data */
idle      = 0;
digits    = 0;
data_read = 0;
while (1) {
    data = RdPortA();
    if (data & VALID) {         /* if valid line set */
        idle = 0;
        if (data_read == 0) {       /* if digit not processed yet */
            data &= 0xf;
            if (data == 12) {       /* if # key */
                if (digits == 4 &&          /* check code */
                    digit1 == CODE1 &&
                    digit2 == CODE2 &&
                    digit3 == CODE3 &&
                    digit4 == CODE4) {
                    WrPortB(0x10);          /* set B<4> if correct cod
                    delay_sec(2);
                }
                WrPortB(0x0);               /* reset */
                digits = 0;
                data_read = 0;
            } else {
                digits++;                   /* save the digit */
                data_read = 1;
                if (digits == 1) {
```

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

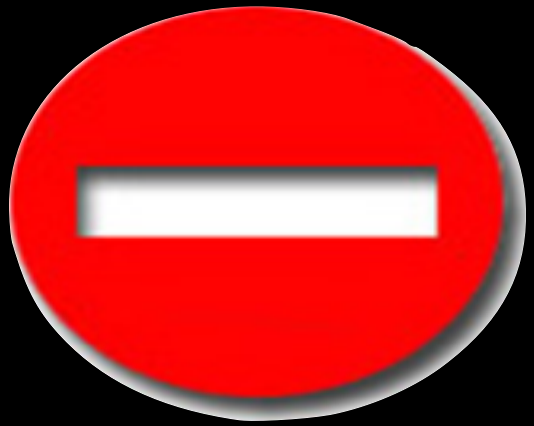
Constraints

Precedence

Conjunction

Exclusion

Dependence



BLOCKED

... 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

SBSE is so generic

Survey:

Mark Harman, Afshin Mansouri and Yuanyuan Zhang,

Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications

Technical Report TR-09-03, King's College London, 2009

SBSE is so generic

Survey:

Mark Harman,Afshin Mansouri and Yuanyuan Zhang,

Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications

Technical Report TR-09-03, King's College London, 2009

SBSE is so generic

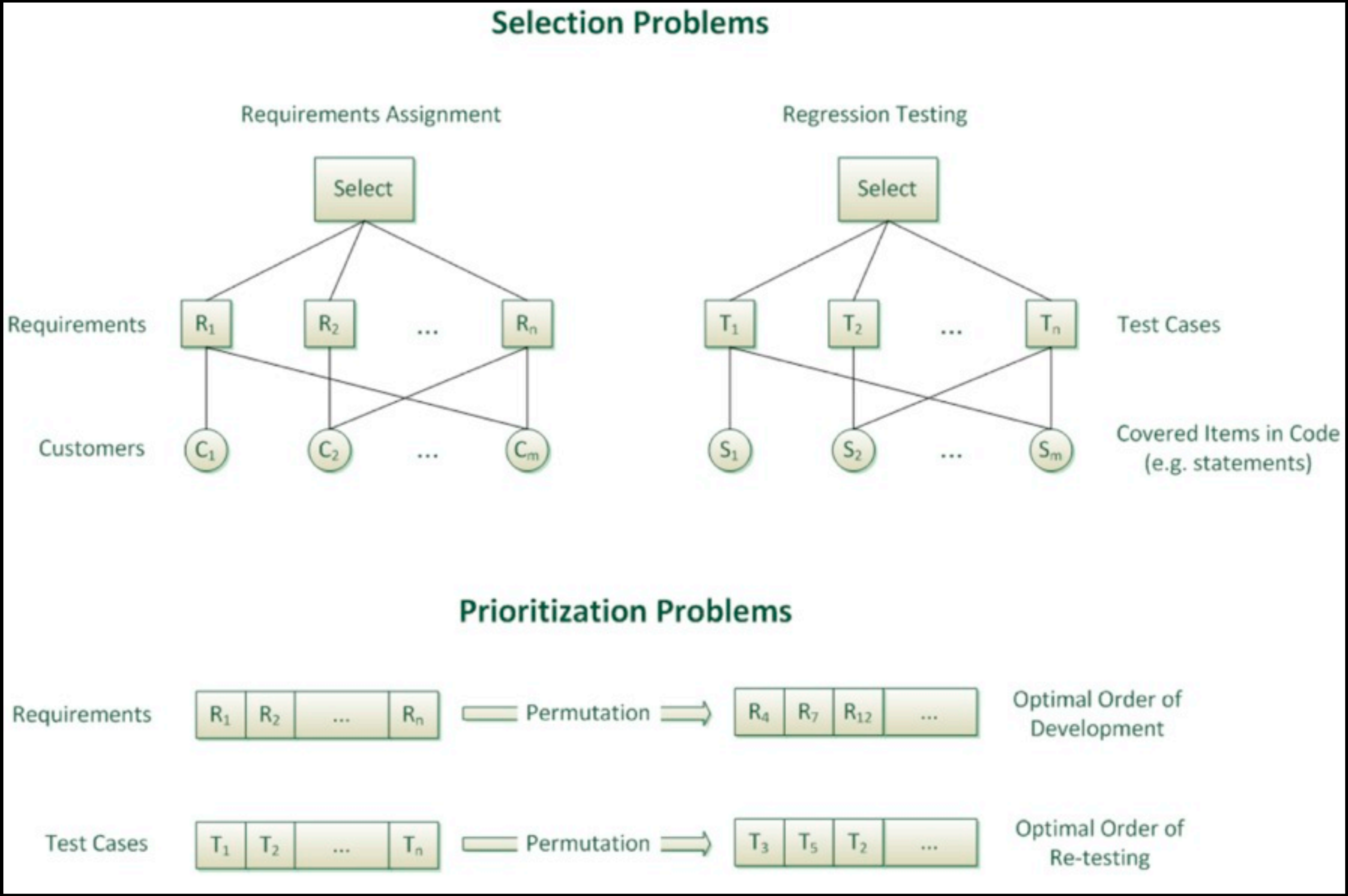
Survey:

Mark Harman,Afshin Mansouri and Yuanyuan Zhang,

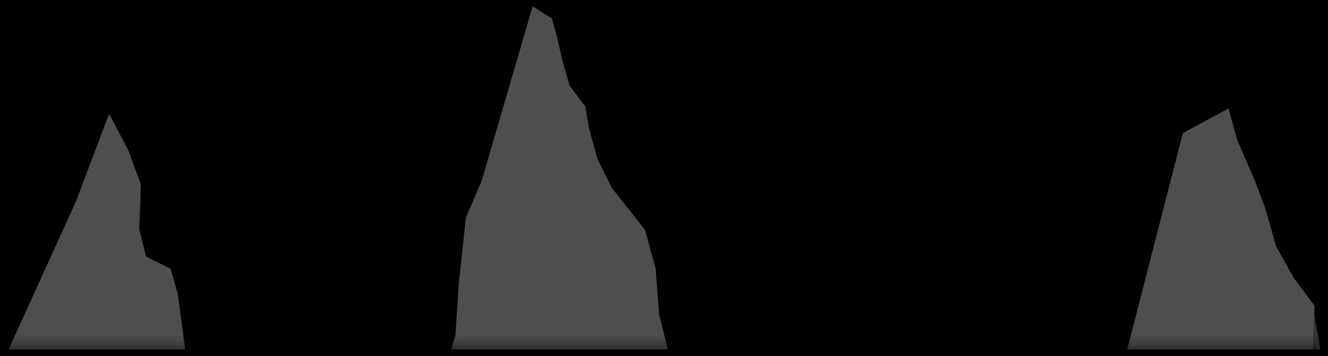
Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications

Technical Report TR-09-03, King's College London, 2009

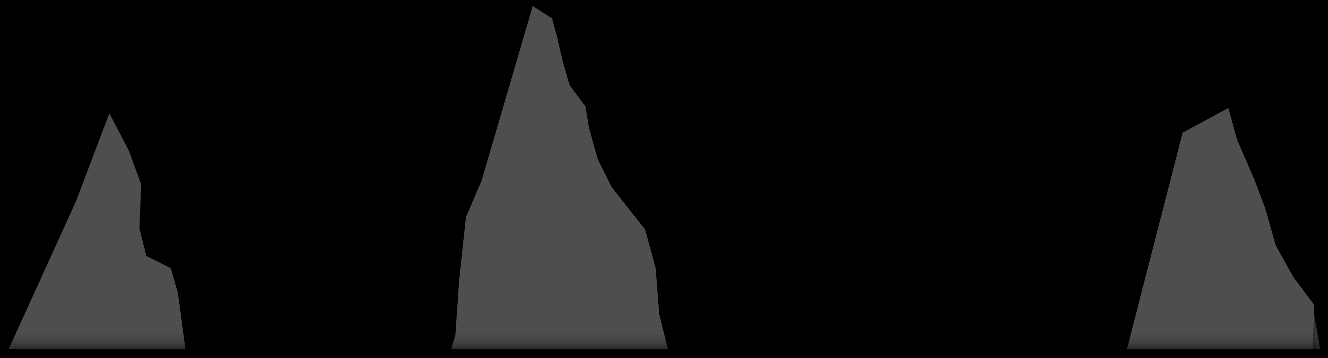
Now to appear in ACM Surveys.


Selection Problems
Requirements Assignment
Regression Testing
Select
Select
Requirements
R_1
R_2
...
R_n
T_1
T_2
...
T_n
Test Cases
Customers
C_1
C_2
...
C_m
S_1
S_2
...
S_m
Covered Items in Code
(e.g. statements)
Prioritization Problems
Requirements
R_1
R_2
...
R_n
Permutation
R_4
R_7
R_{12}
...
Optimal Order of
Development
Test Cases
T_1
T_2
...
T_n
Permutation
T_3
T_5
T_2
...
Optimal Order of
Re-testing

Requirements and regression testing:



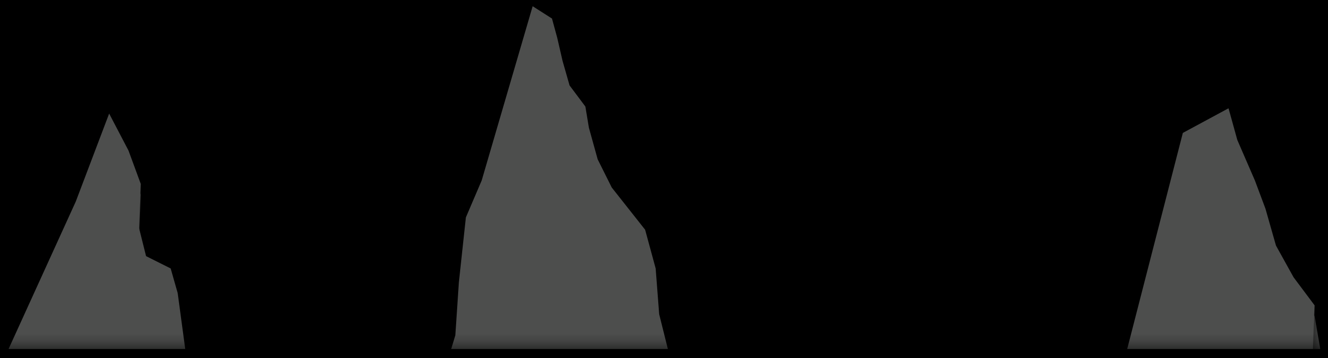
Requirements and regression testing:



Alone

Requirements and regression testing:

Requirements

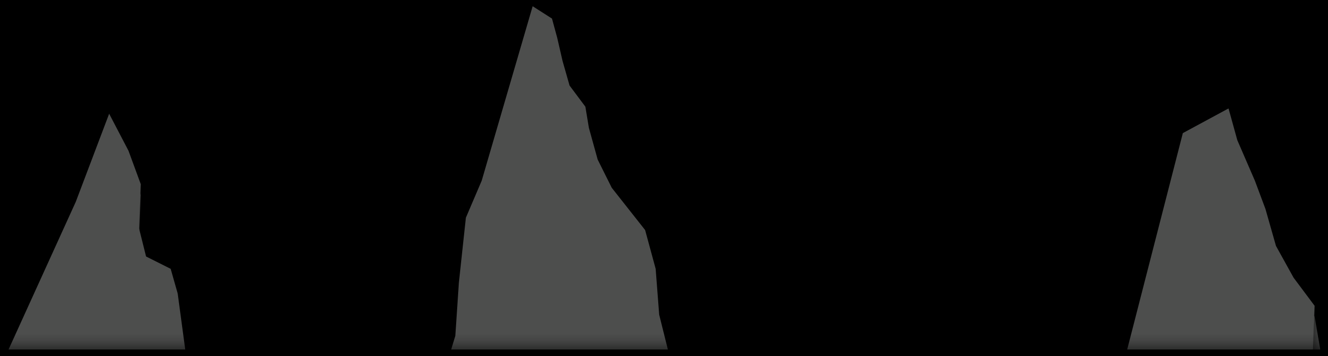


Alone

Requirements and regression testing:

Requirements

early phase



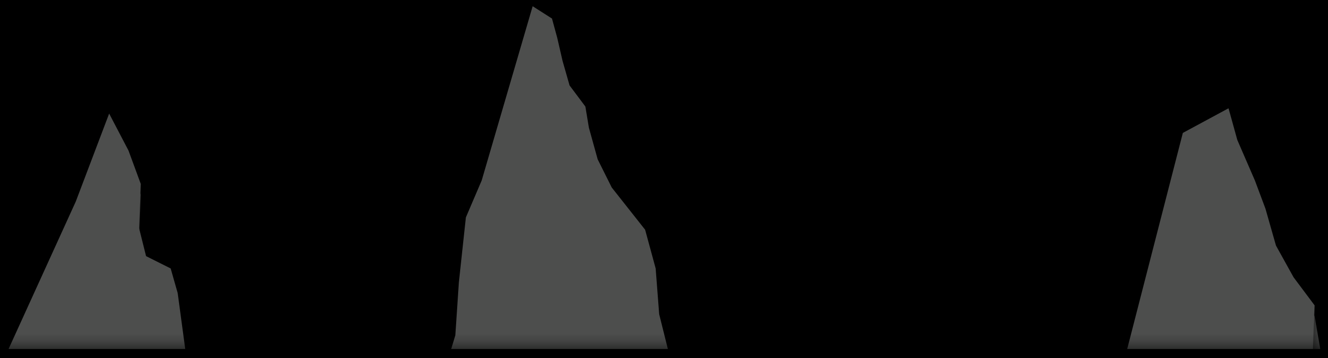
Alone

Requirements and regression testing:

Requirements

early phase

RE



Alone

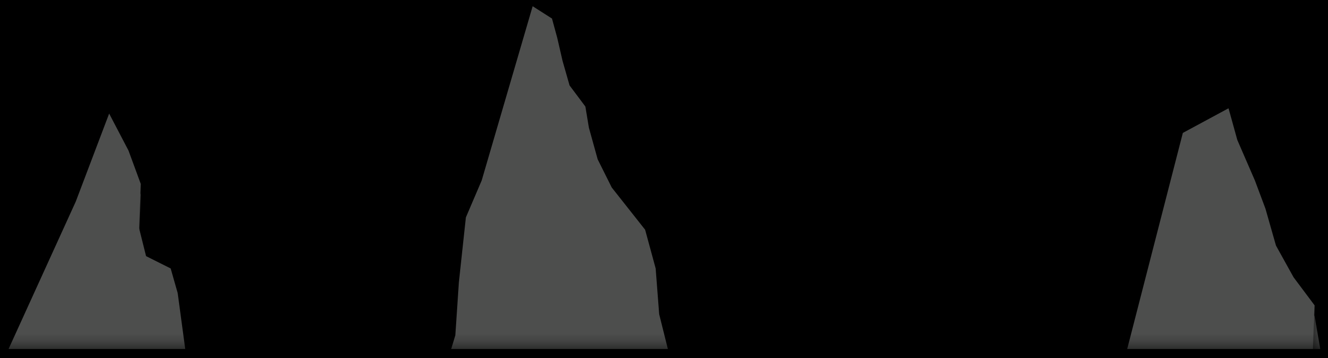
Requirements and regression testing:

Requirements

early phase

RE

REFSQ



Alone

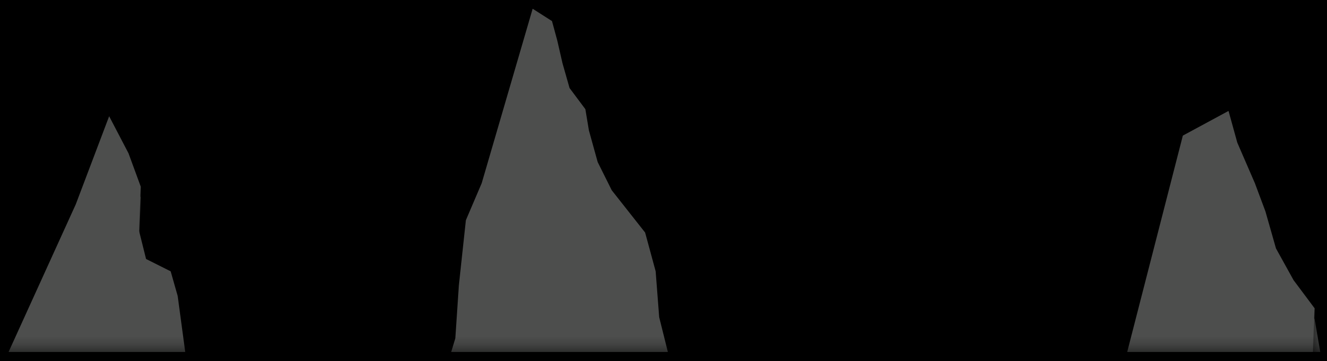
Requirements and regression testing:

Requirements

early phase

RE

REFSQ



Regression

Alone

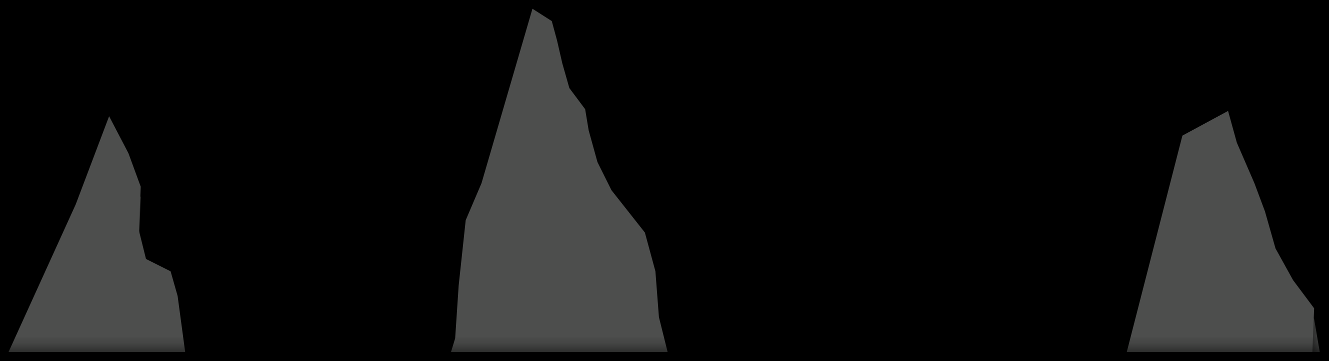
Requirements and regression testing:

Requirements

early phase

RE

REFSQ



Regression

late phase

Alone

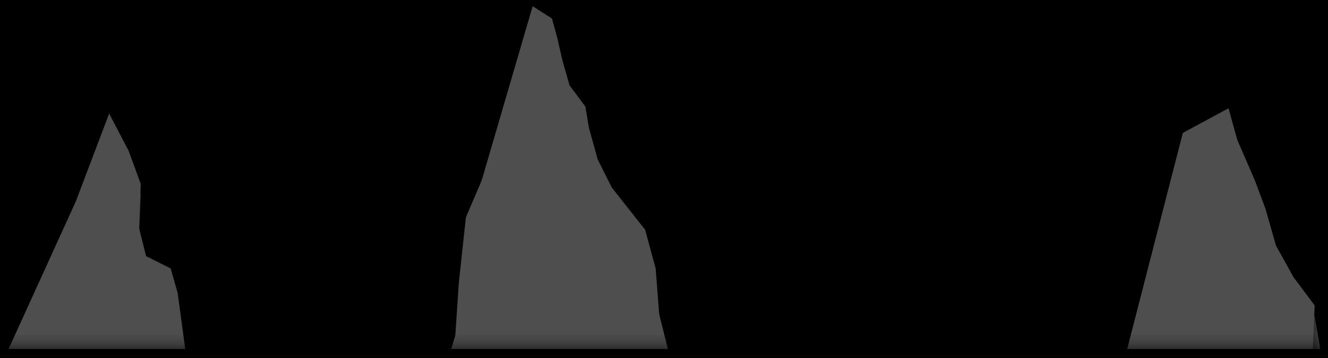
Requirements and regression testing:

Requirements

early phase

RE

REFSQ



Regression

late phase

ICST

Alone

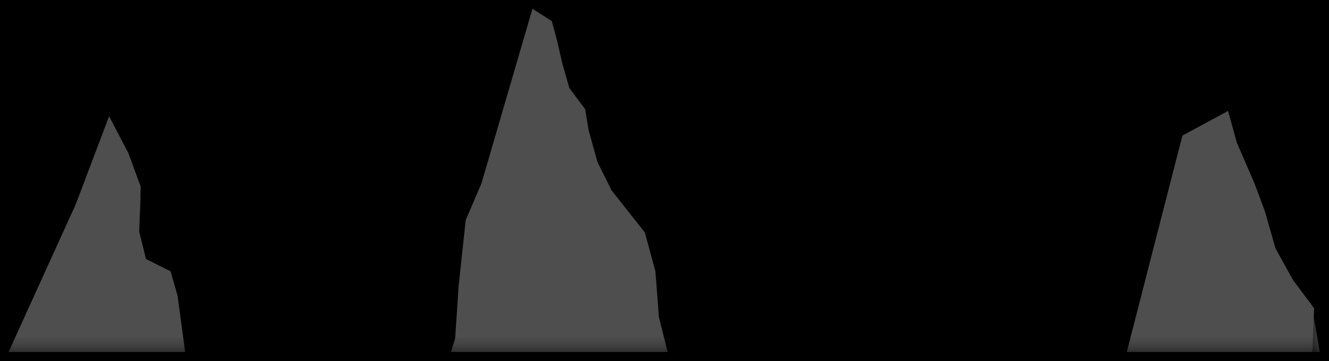
Requirements and regression testing:

Requirements

early phase

RE

REFSQ



Regression

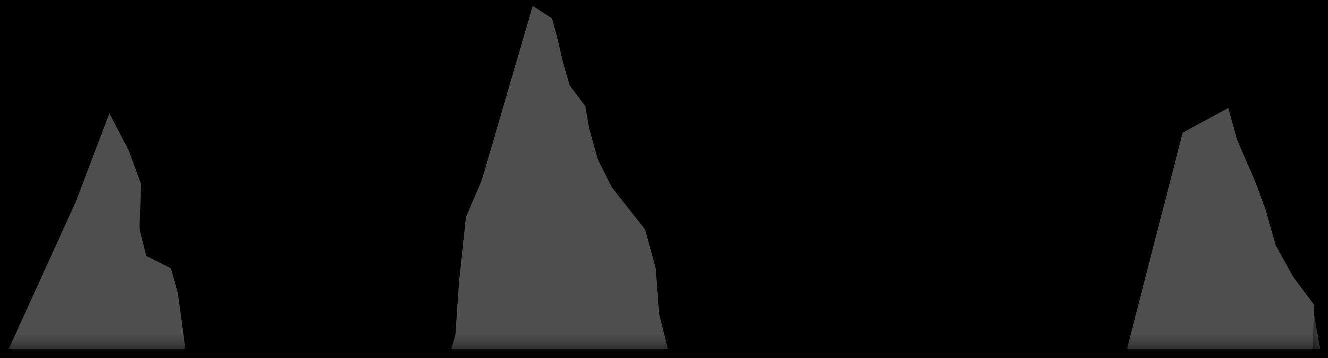
late phase

ICST

ISSTA

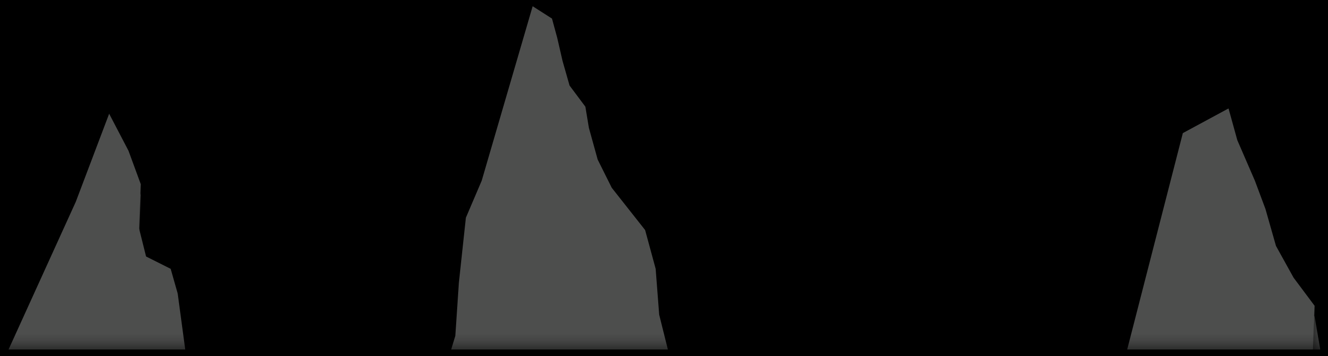
Alone

Requirements and regression testing: really different ?



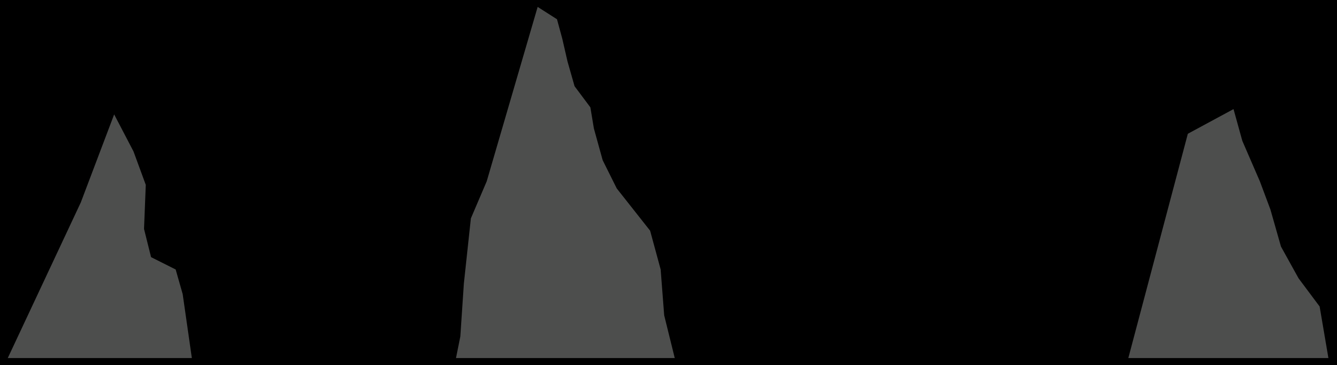
Alone

Requirements and regression testing: really different ?



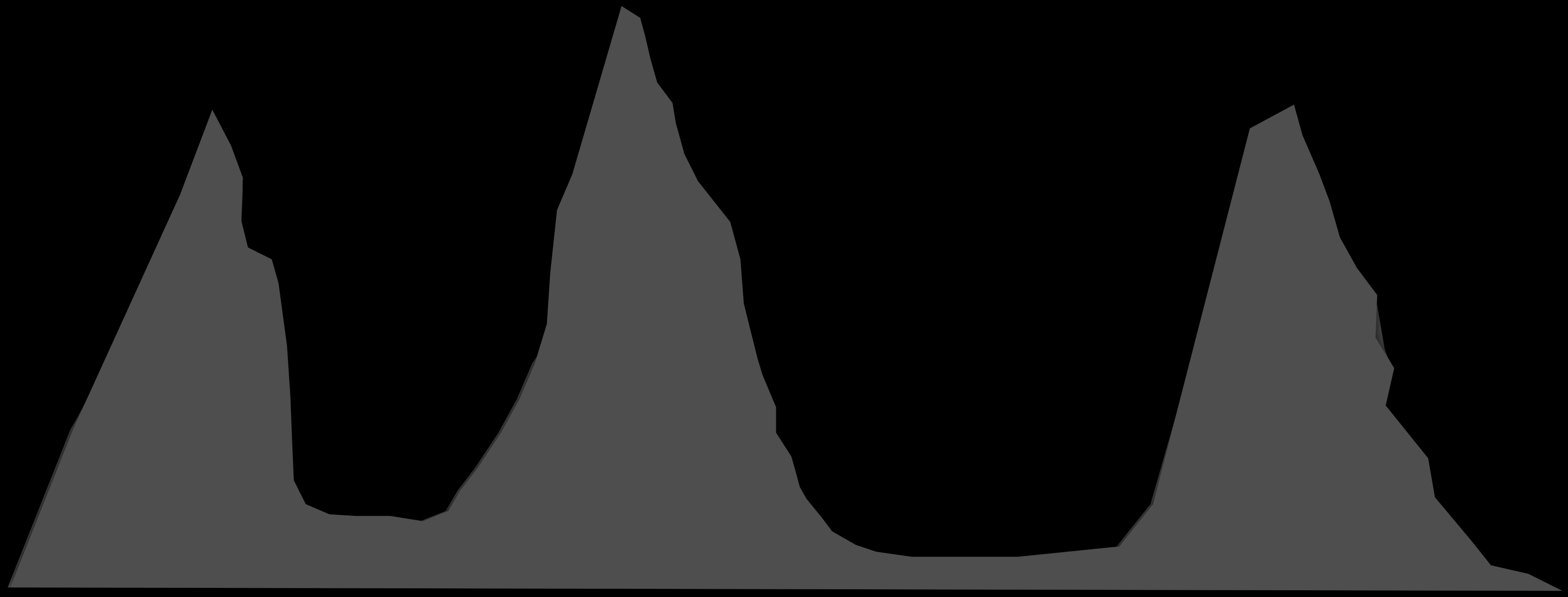
Alone

Requirements and regression testing: really different ?



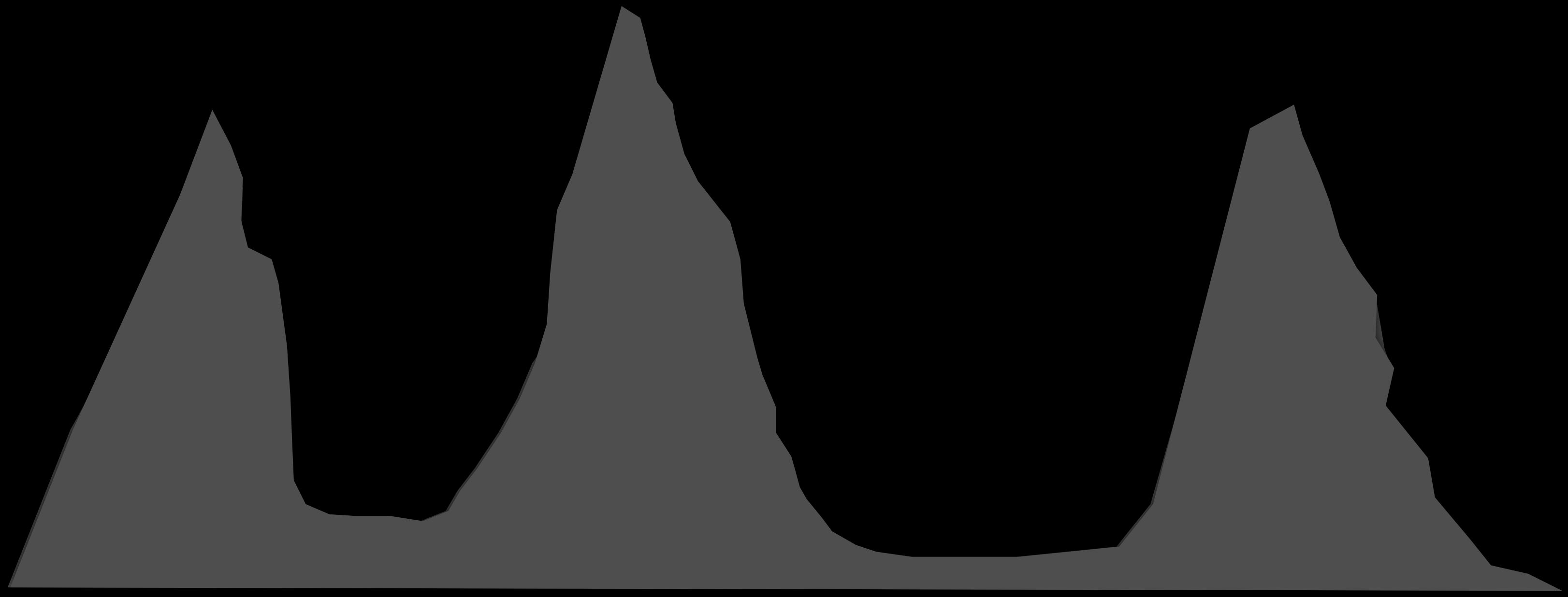
Alone

Requirements and regression testing: really different ?



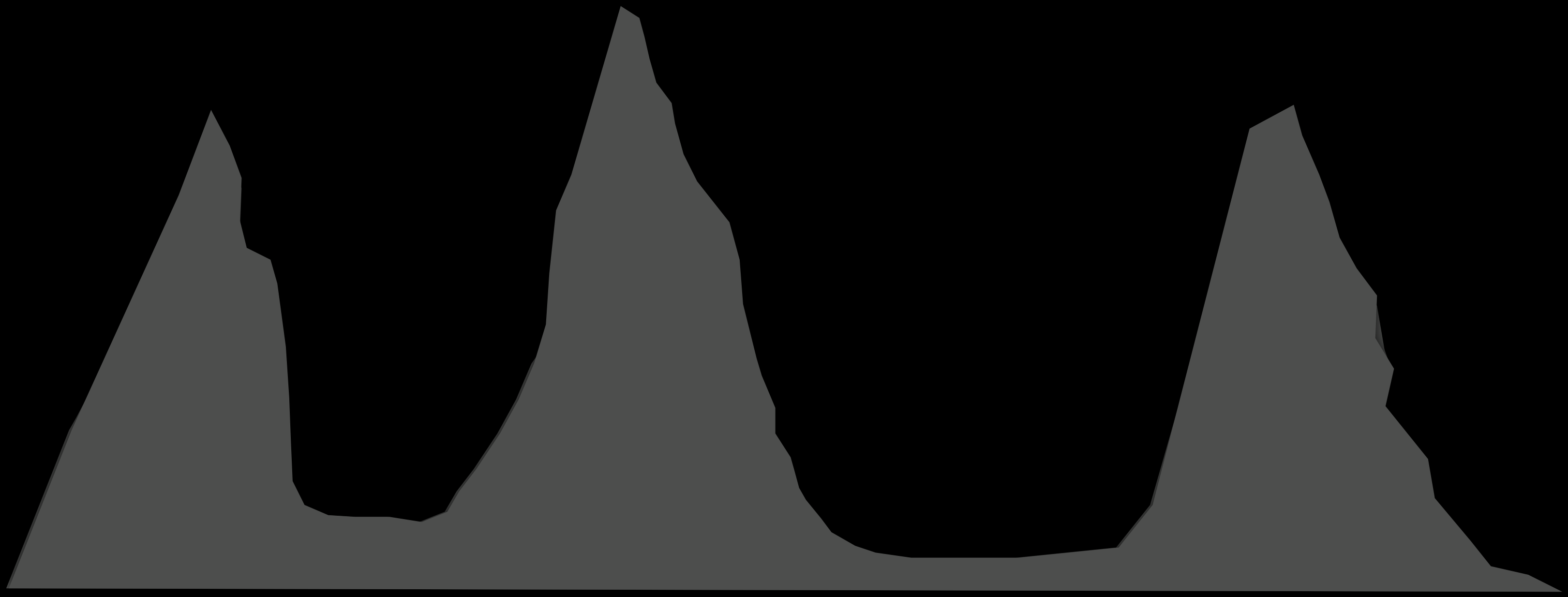
Alone

Requirements and regression testing: really different ?



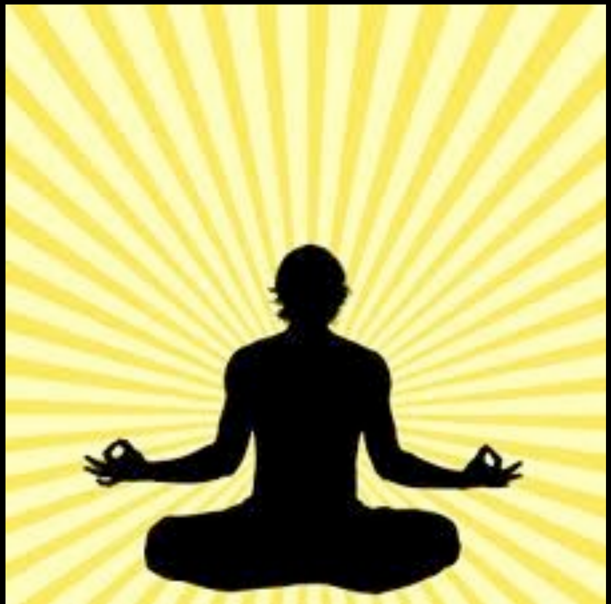
Alone

Requirements and regression testing: really different ?



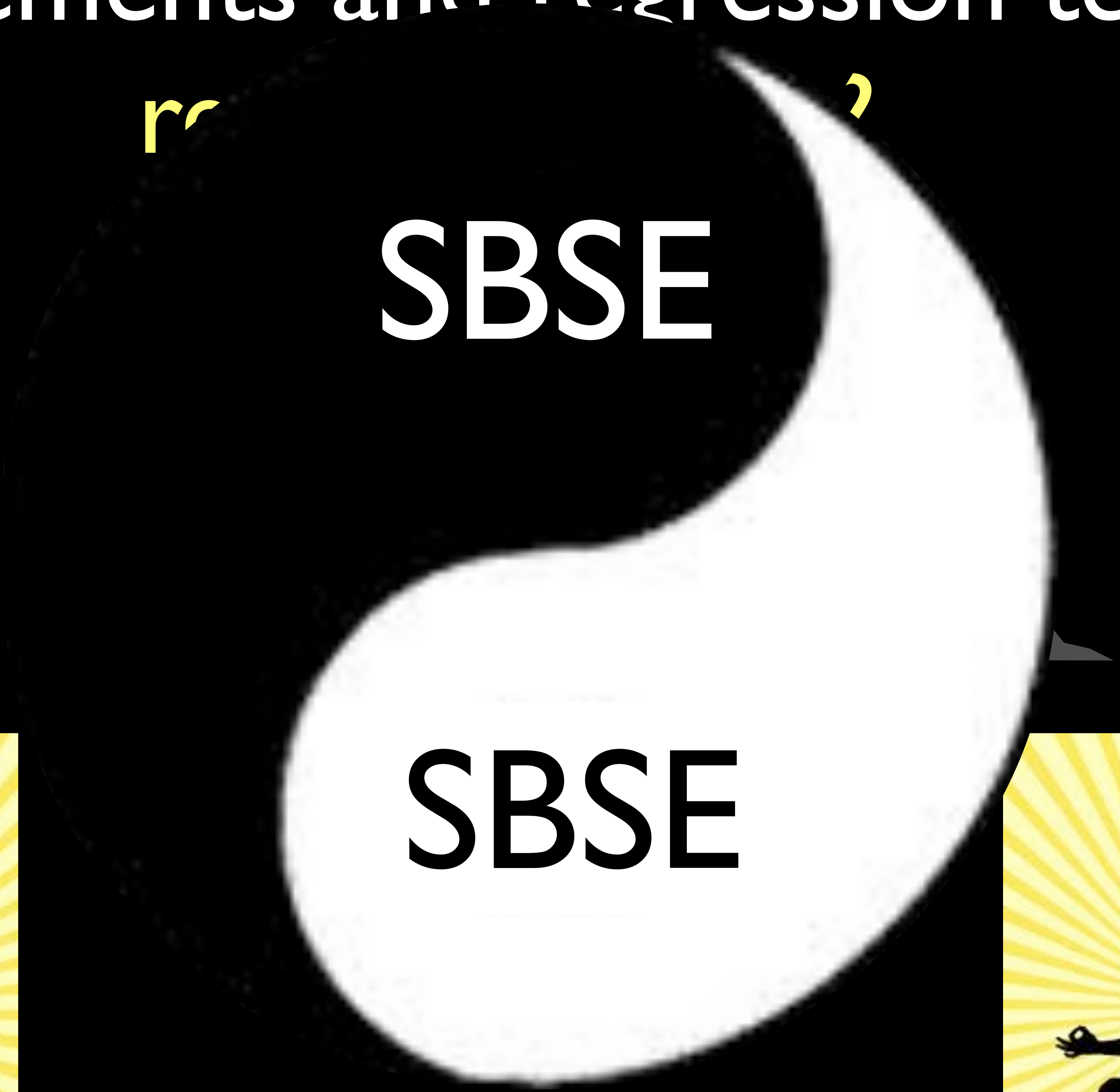
Alone

Requirements and regression testing:



Requirements and regression testing:

r



… but …
why is
Software Engineering different?

Search Based Optimization

Search Based Optimization

Mechanical Engineering

Electronic Engineering

Civil Engineering

Aerospace Engineering

What makes Software Engineering so special ?

Search Based Optimization

Mechanical Engineering

Electronic Engineering

Civil Engineering

Aerospace Engineering

What makes Software Engineering so special ?

FASE 2010 Keynote

Mark Harman
Why the Virtual Nature of Software Makes it Ideal for Search Based Optimization
FASE 2010.

European Joint Conferences on Theory And Practice of Software 2010

FASE 2010 Keynote

Mark Harman
Why the Virtual Nature of Software Makes it Ideal for Search Based Optimization
FASE 2010.

in situ fitness test

in situ fitness test

Physical Engineering

in situ fitness test

Physical Engineering

in situ fitness test

Physical Engineering

cost: $20,000.00

in situ fitness test

Physical Engineering

Virtual Engineering

cost: $20,000.00

in situ fitness test

Physical Engineering

cost: $20,000.00

Virtual Engineering

in situ fitness test

Physical Engineering

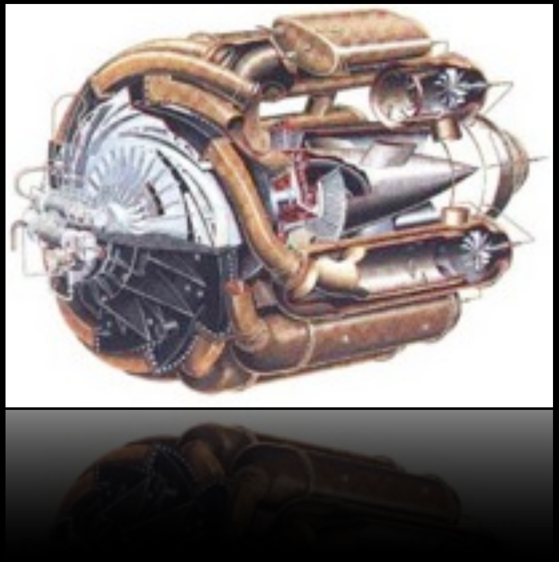
cost: $20,000.00

Virtual Engineering

cost: $0.00.0000000002

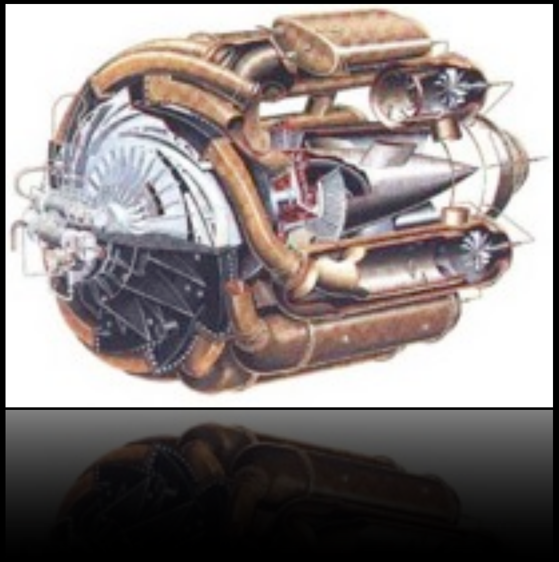
Traditional Engineering Artifact

Traditional Engineering Artifact



Traditional Engineering Artifact

Optimization goal



Traditional Engineering Artifact

Optimization goal

Maximize compression

Traditional
Engineering Artifact

Optimization
goal

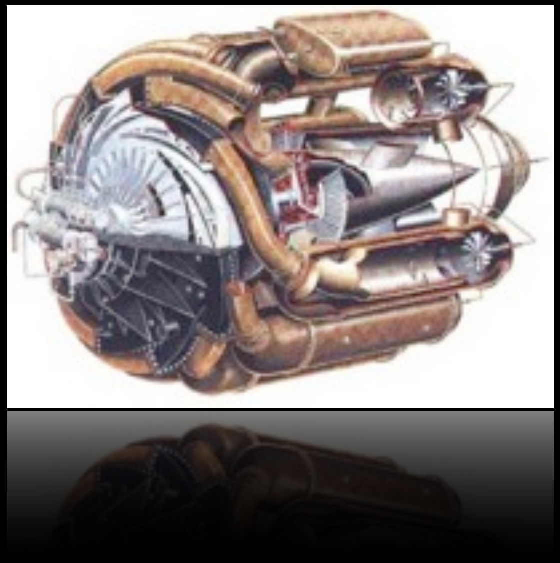
Maximize compression

Minimize fuel consumption

Traditional Engineering Artifact

Optimization goal

Fitness computed on a representation



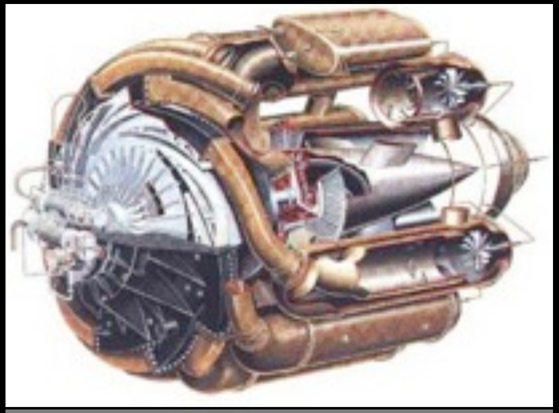
Maximize compression

Minimize fuel consumption

Traditional Engineering Artifact

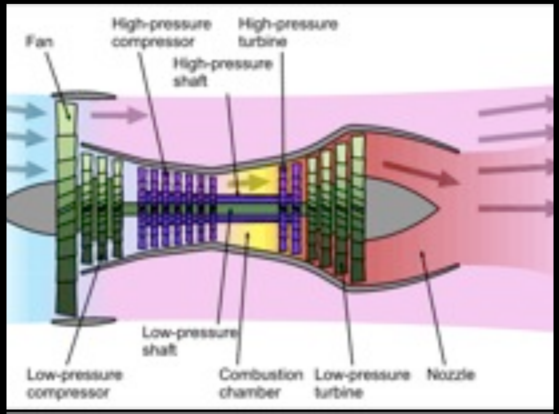
Optimization goal

Fitness computed on a representation



Maximize compression

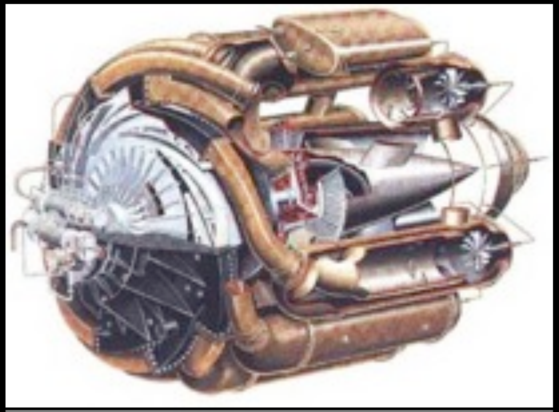
Minimize fuel consumption



Traditional Engineering Artifact

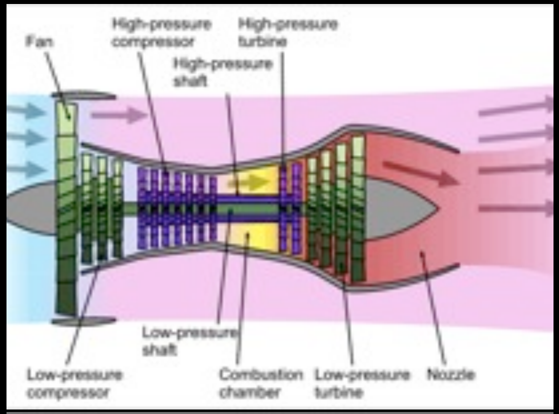
Optimization goal

Fitness computed on a representation



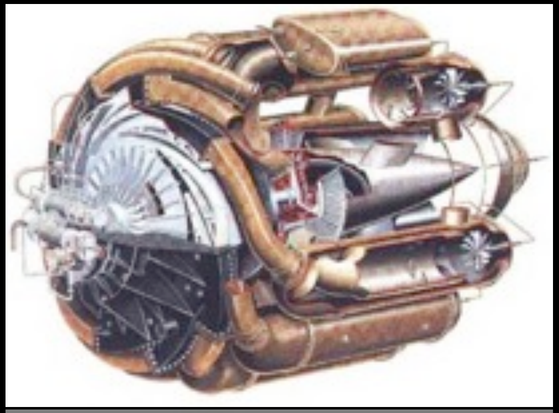
Maximize compression

Minimize fuel consumption



Software Engineering Artifact

Traditional Engineering Artifact

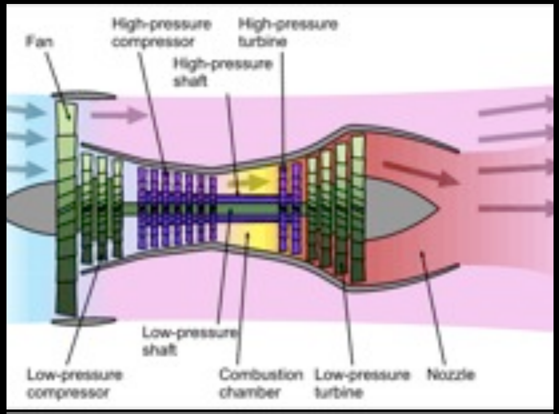


Optimization goal

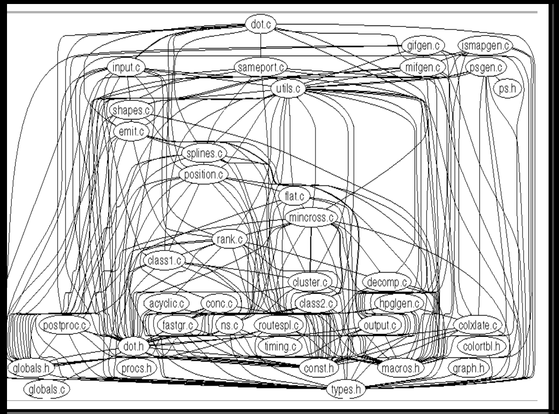
Maximize compression

Minimize fuel consumption

Fitness computed on a representation



Software Engineering Artifact



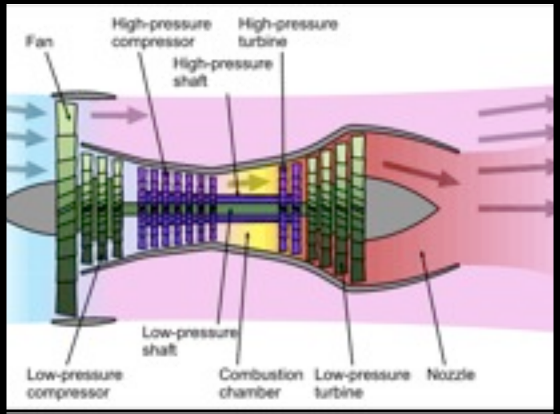
Traditional Engineering Artifact

Optimization goal

Fitness computed on a representation

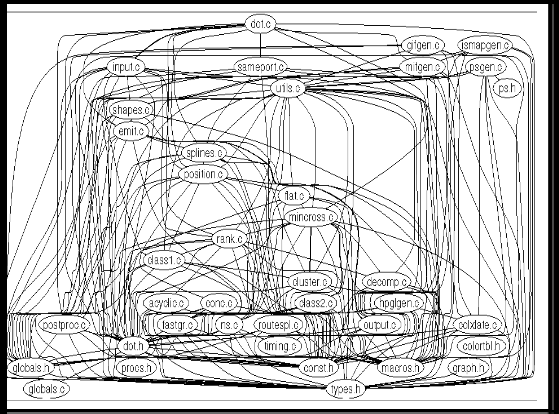
Maximize compression

Minimize fuel consumption



Software Engineering Artifact

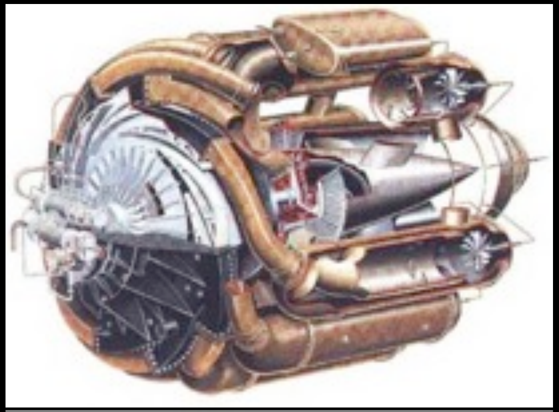
Optimization goal



Traditional
Engineering Artifact

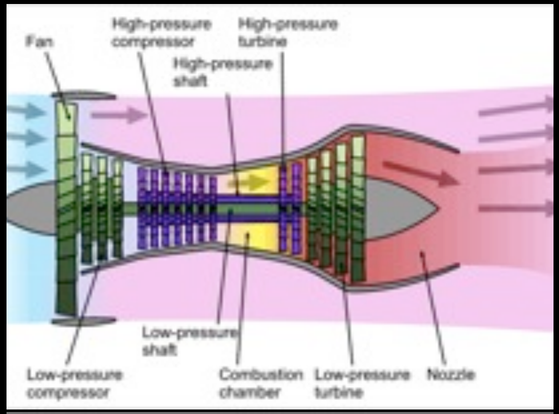
Optimization
goal

Fitness computed
on a representation



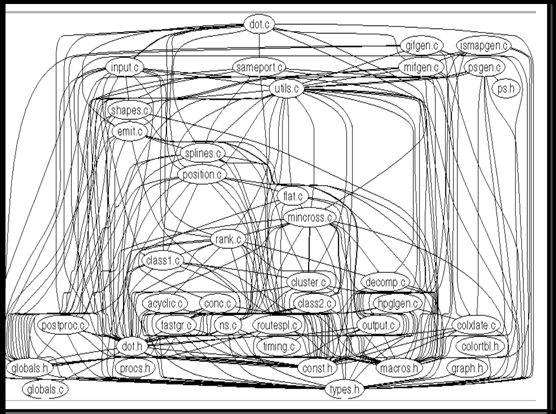
Maximize compression

Minimize fuel consumption



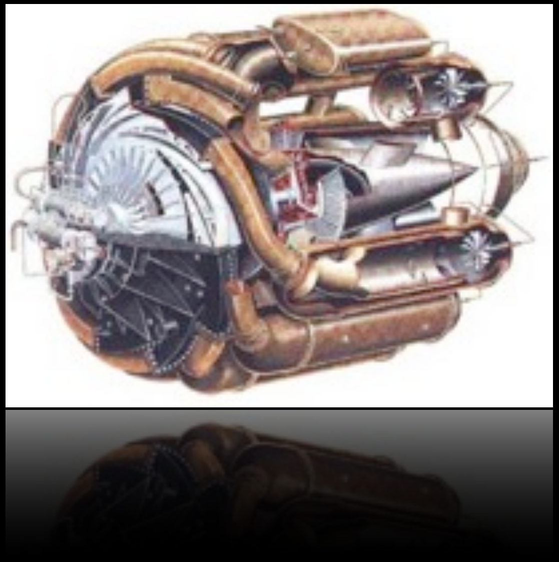
Software
Engineering Artifact

Optimization
goal



Maximize cohesion

Traditional Engineering Artifact

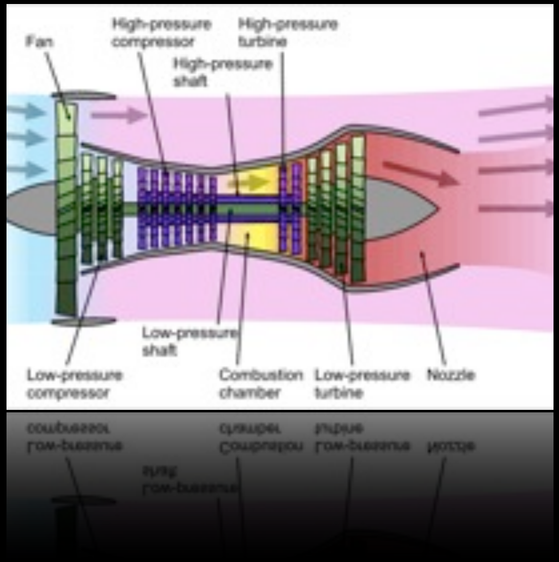


Optimization goal

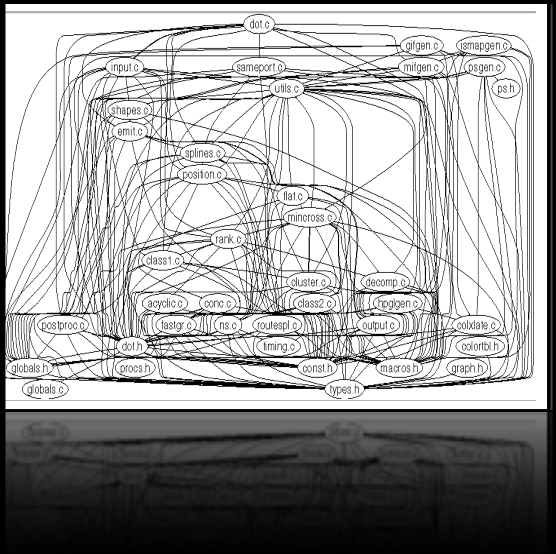
Maximize compression

Minimize fuel consumption

Fitness computed on a representation



Software Engineering Artifact



Optimization goal

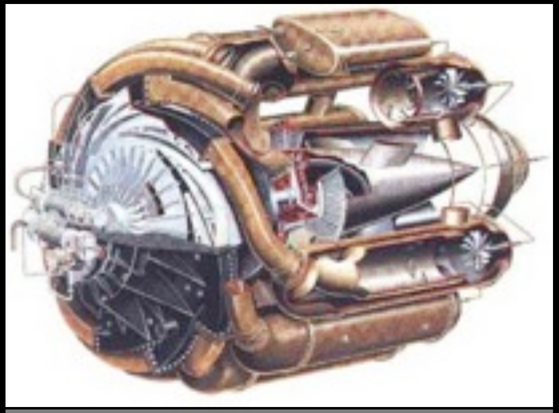
Maximize cohesion

Minimize coupling

Traditional
Engineering Artifact

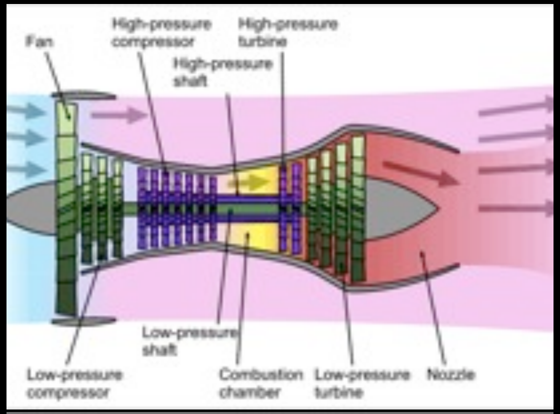
Optimization
goal

Fitness computed
on a representation



Maximize compression

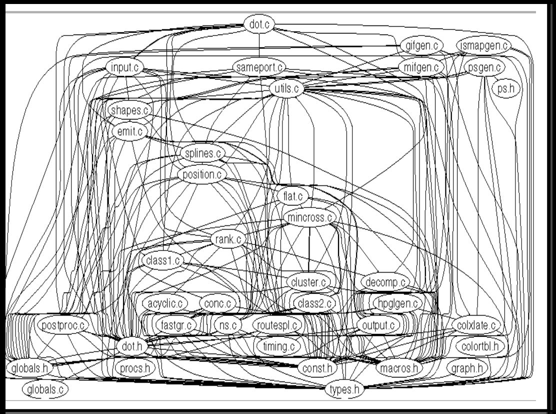
Minimize fuel consumption



Software
Engineering Artifact

Optimization
goal

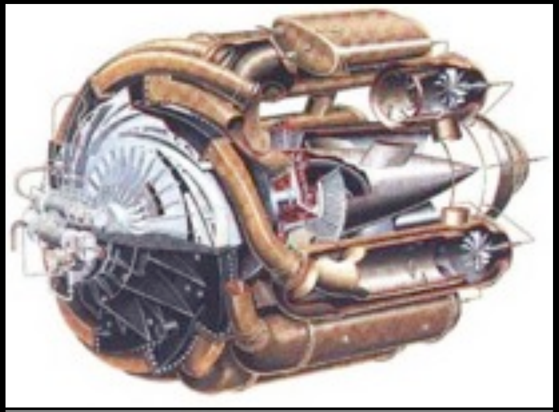
Fitness computed
Directly



Maximize cohesion

Minimize coupling

Traditional
Engineering Artifact

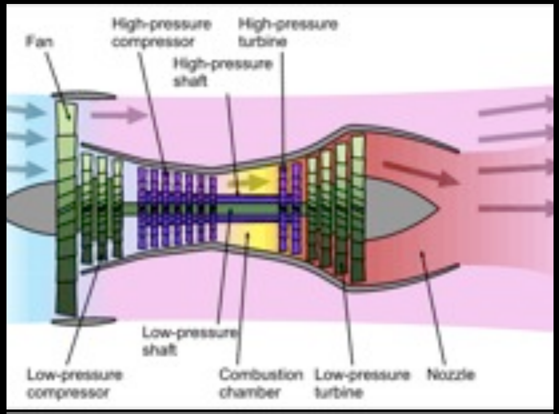


Optimization
goal

Maximize compression

Minimize fuel consumption

Fitness computed
on a representation



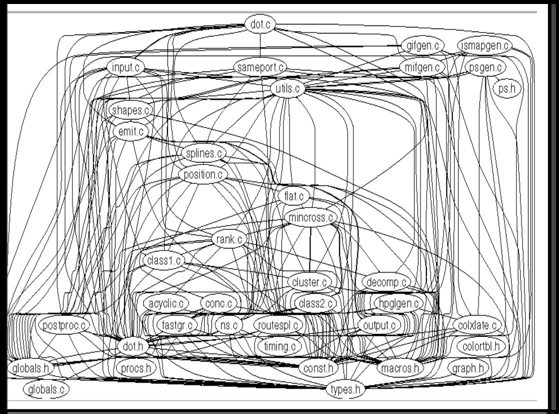
Software
Engineering Artifact

Optimization
goal

Maximize cohesion

Minimize coupling

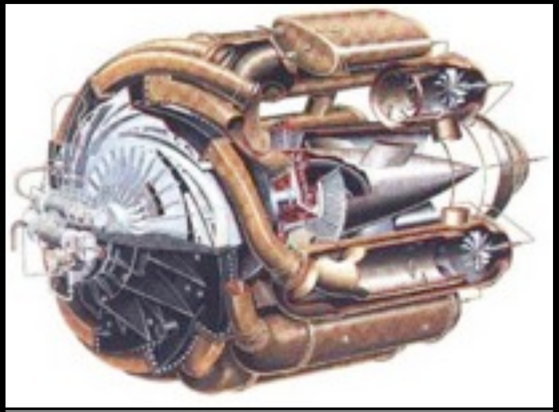
Fitness computed
Directly



Traditional Engineering Artifact

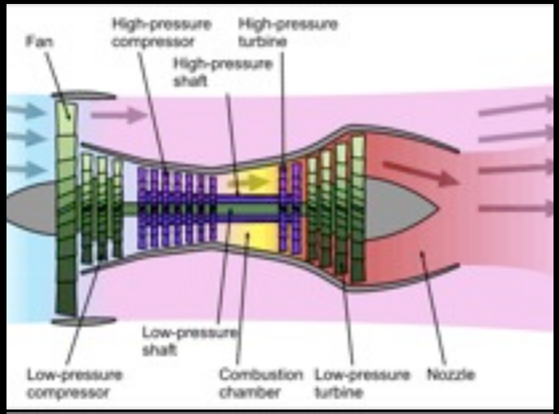
Optimization goal

Fitness computed on a representation



Maximize compression

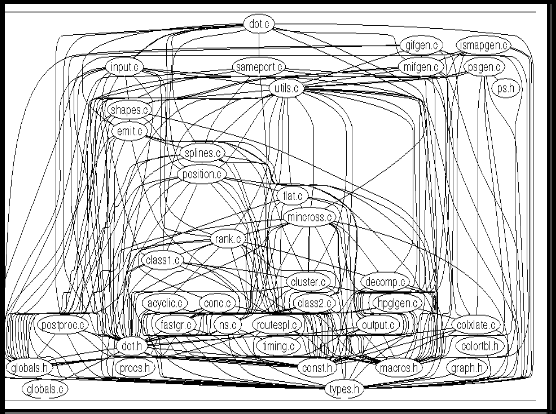
Minimize fuel consumption



Software Engineering Artifact

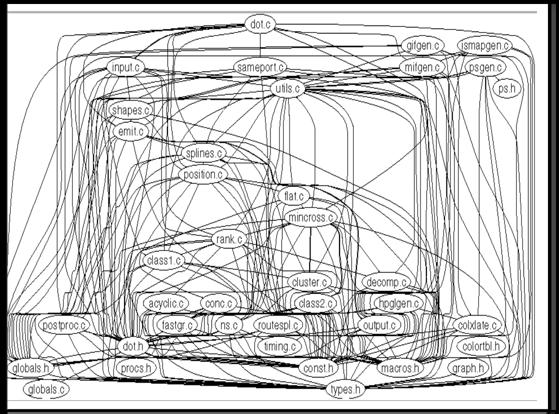
Optimization goal

Fitness computed Directly



Maximize cohesion

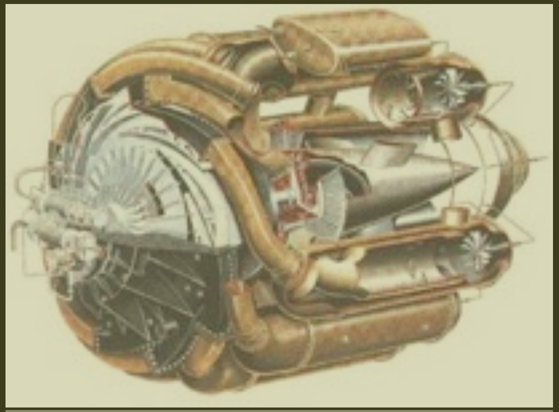
Minimize coupling



Traditional Engineering Artifact

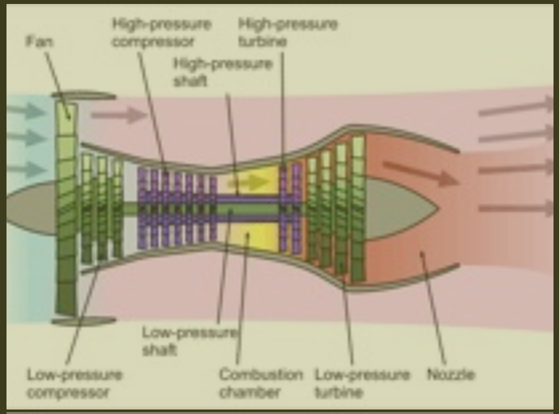
Optimization goal

Fitness computed on a representation



Maximize compression

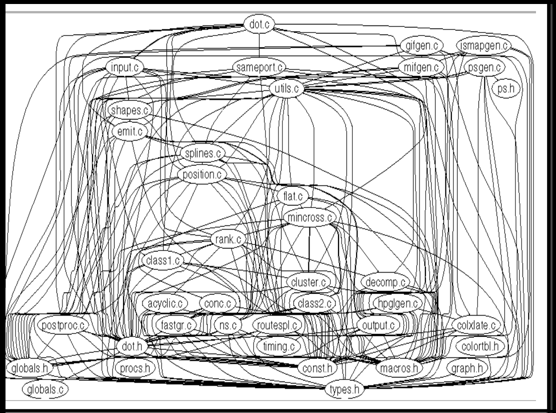
Minimize fuel consumption



Software Engineering Artifact

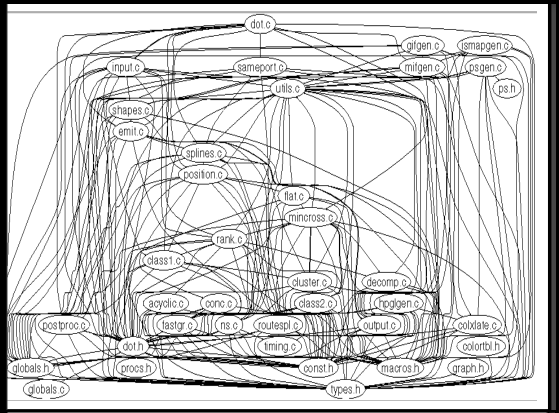
Optimization goal

Fitness computed Directly



Maximize cohesion

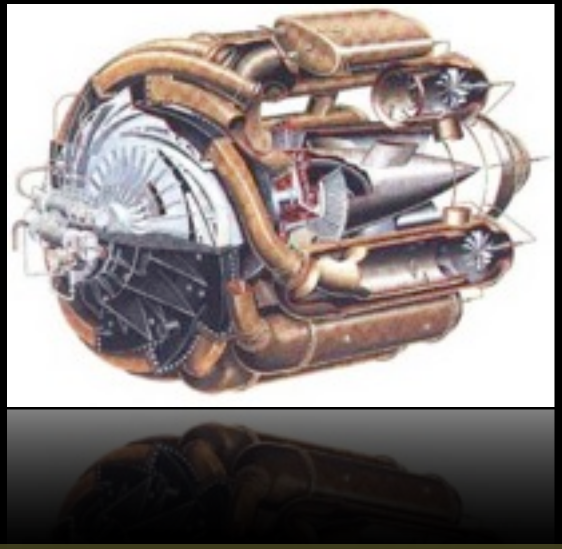
Minimize coupling



Traditional Engineering Artifact

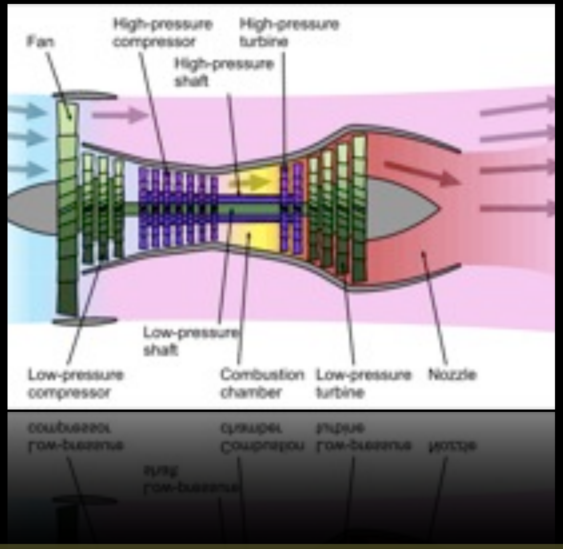
Optimization goal

Fitness computed on a representation



Maximize compression

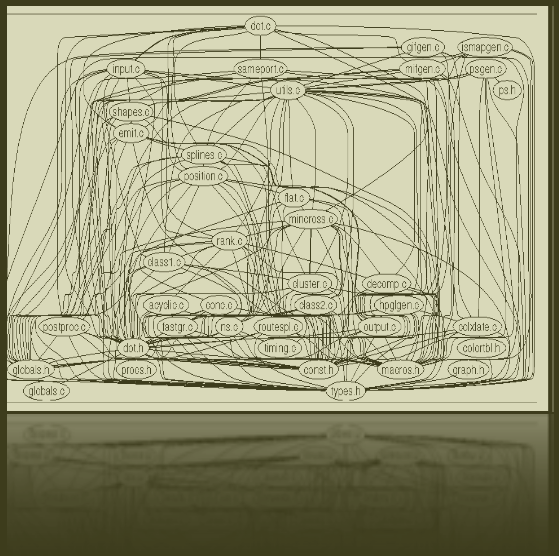
Minimize fuel consumption



Software Engineering Artifact

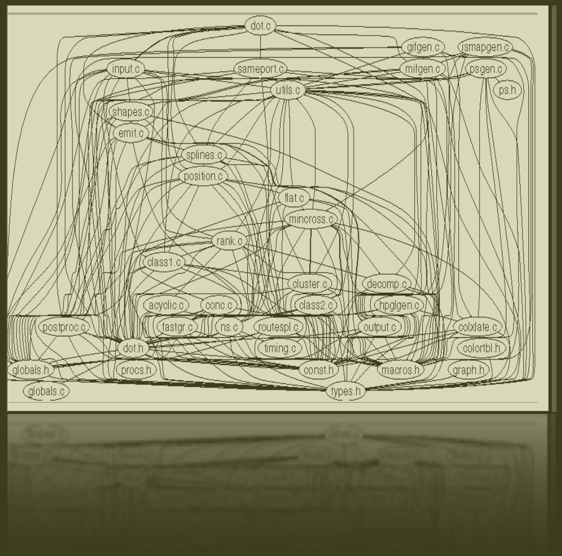
Optimization goal

Fitness computed Directly



Maximize cohesion

Minimize coupling



ssbse

symposium on search based software engineering

SSBSE: build and support the growing SBSE community:

1. share ideas about problems and solutions
2. develop research agenda
3. critical mass of interaction

ssbse

symposium on search based software engineering

SSBSE has a democratic charter

Governed by a Steering Committee

9 members of the Committee

Listed on the website and in the proceedings

SC decides on strategy and policies

Elected by the community

ssbse

symposium on search based software engineering

SSBSE has a democratic charter

3 drop off the SC each year

2 terms maximum without a break

Procedure:

```
LOOP
    Nominate; second; accept/decline.
UNTIL Close of nominations;
Election Addresses
Votes: up to 3 votes per attendee. You can vote for a candidate at most once.
```

ssbse

symposium on search based software engineering

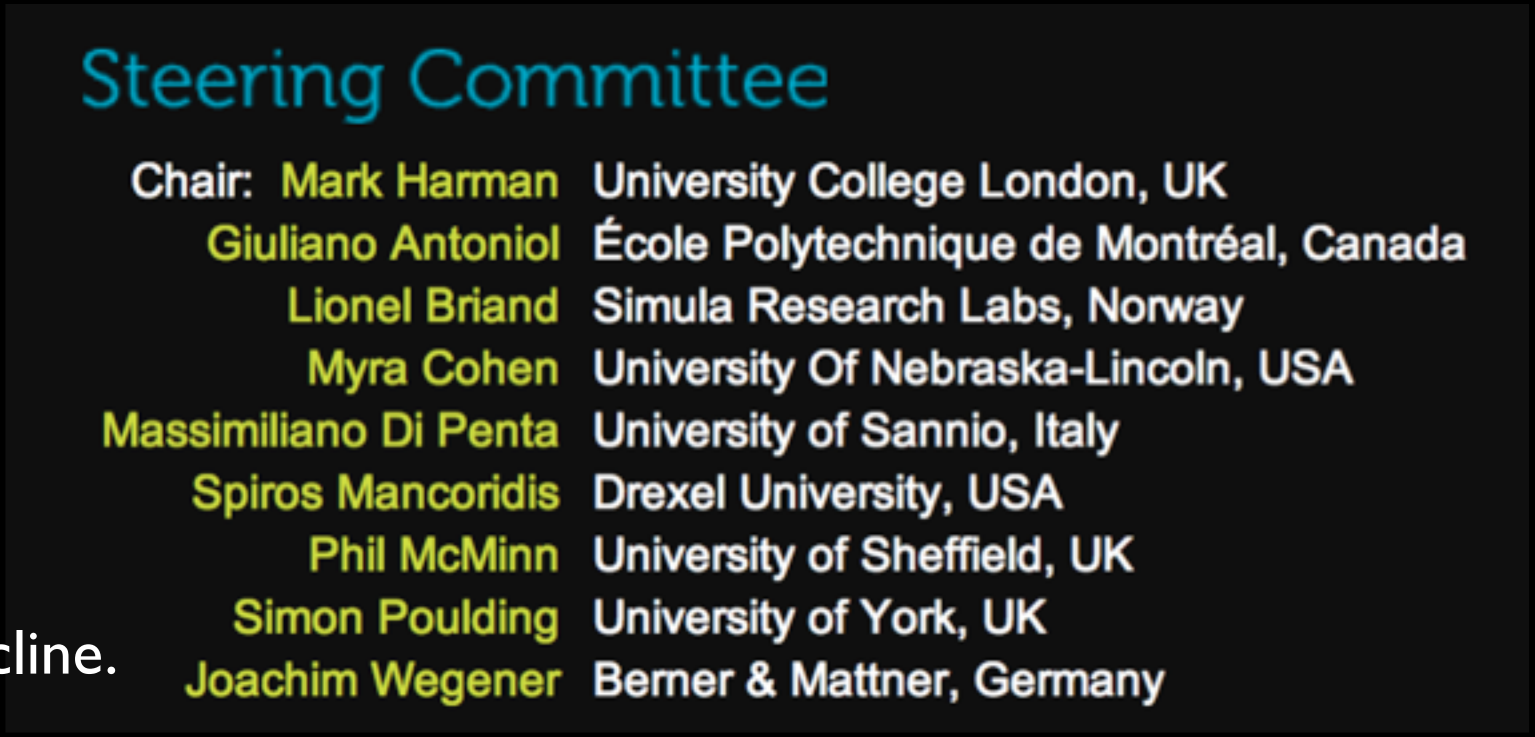
SSBSE has a democratic charter

3 drop off the SC each year

2 terms maximum without a break

Procedure:

- LOOP
  - Nominate; second; accept/decline.
- UNTIL Close of nominations;
- Election Addresses
- Votes: up to 3 votes per attendee. You can vote for a candidate at most once.



Steering Committee

Chair: Mark Harman	University College London, UK
Giuliano Antoniol	École Polytechnique de Montréal, Canada
Lionel Briand	Simula Research Labs, Norway
Myra Cohen	University Of Nebraska-Lincoln, USA
Massimiliano Di Penta	University of Sannio, Italy
Spiros Mancoridis	Drexel University, USA
Phil McMinn	University of Sheffield, UK
Simon Poulding	University of York, UK
Joachim Wegener	Berner & Mattner, Germany

ssbse

symposium on search based software engineering

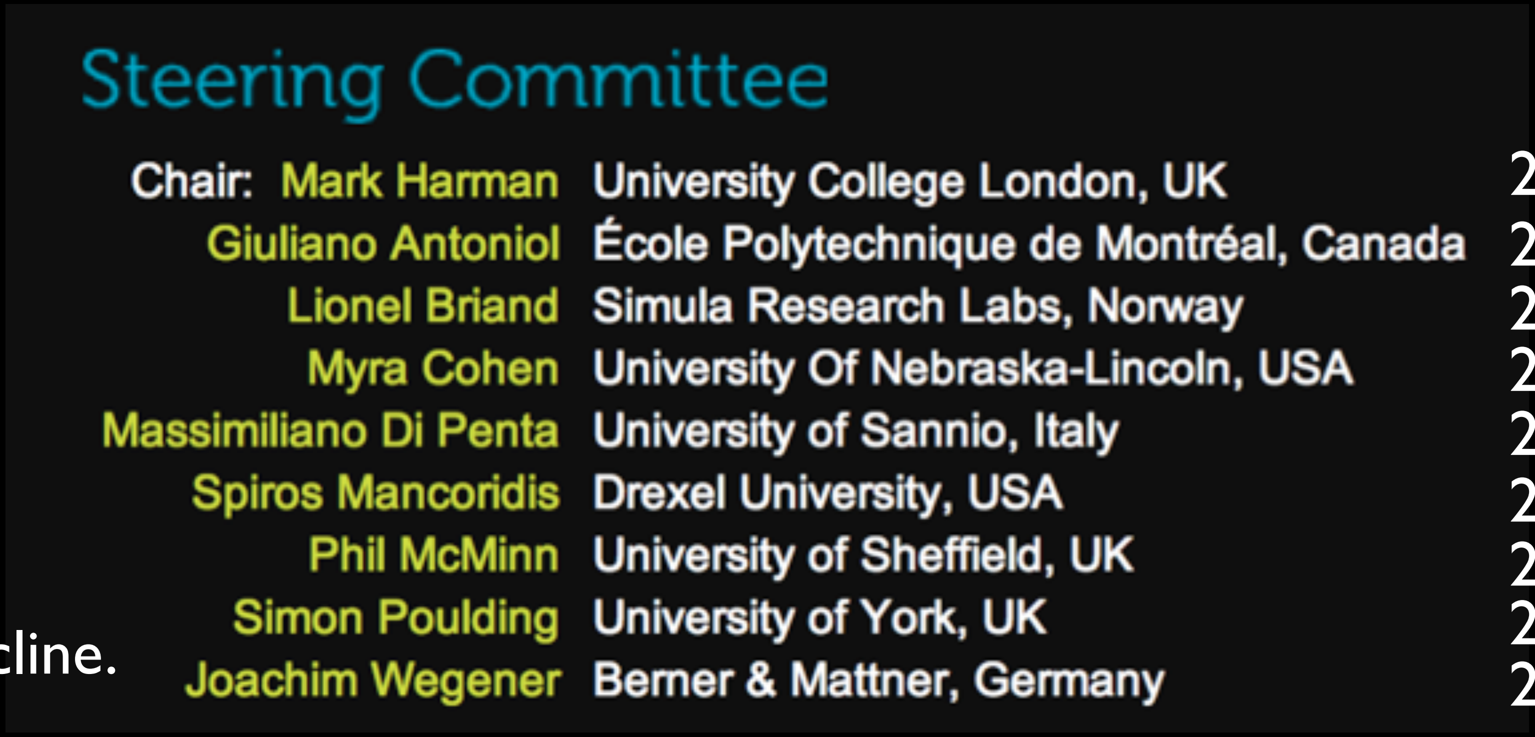
SSBSE has a democratic charter

3 drop off the SC each year

2 terms maximum without a break

Procedure:

```
    LOOP
        Nominate; second; accept/decline.
    UNTIL Close of nominations;
    Election Addresses
    Votes: up to 3 votes per attendee. You can vote for a candidate at most once.
```



Steering Committee

Chair:	Mark Harman	University College London, UK	2013
	Giuliano Antoniol	École Polytechnique de Montréal, Canada	2012
	Lionel Briand	Simula Research Labs, Norway	2012
	Myra Cohen	University Of Nebraska-Lincoln, USA	2013
	Massimiliano Di Penta	University of Sannio, Italy	2012
	Spiros Mancoridis	Drexel University, USA	2011
	Phil McMinn	University of Sheffield, UK	2013
	Simon Poulding	University of York, UK	2011
	Joachim Wegener	Berner & Mattner, Germany	2011

Extra slides

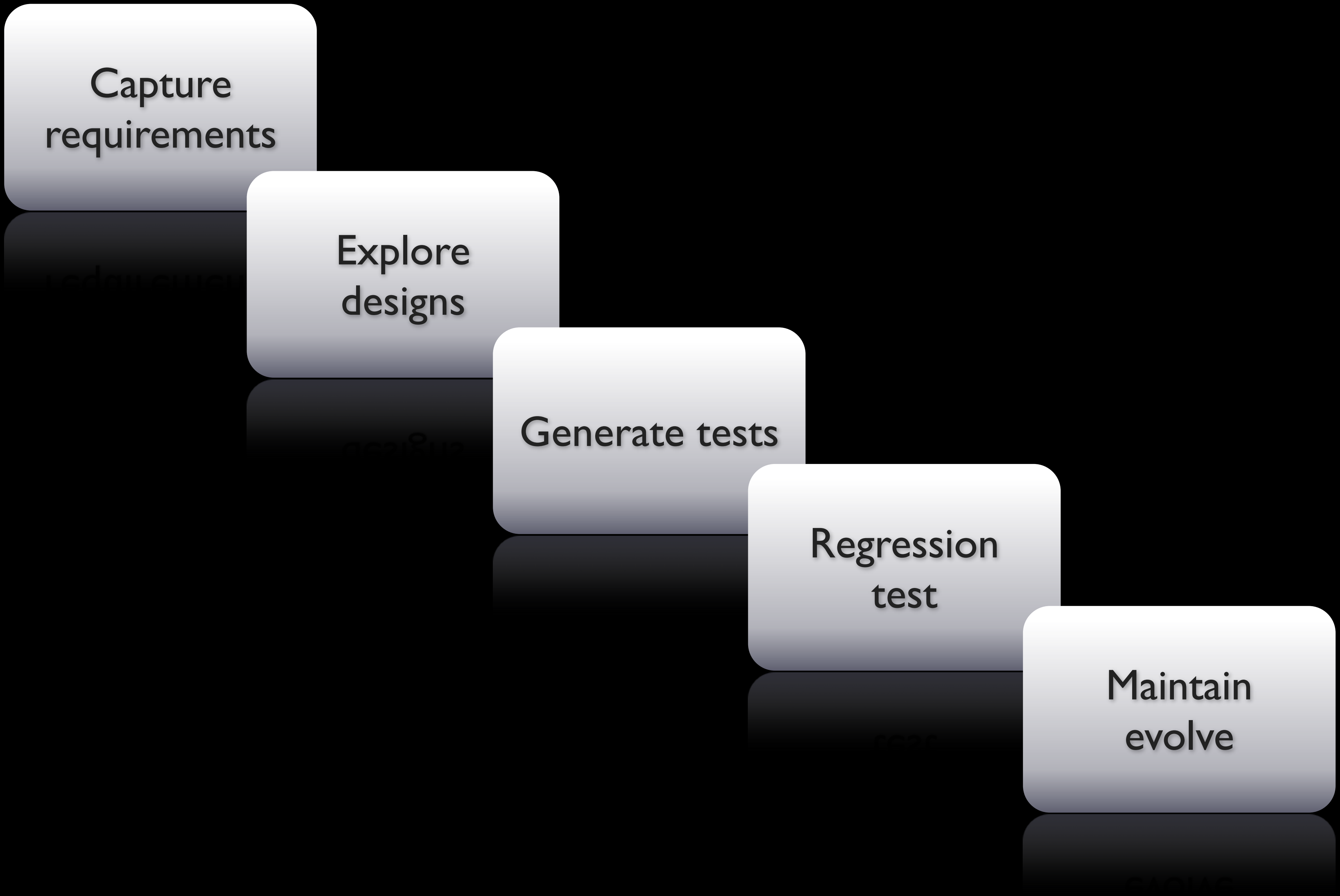
Extra slides

Extra slides

Extra slides

Software Engineering Problems

Software Engineering Problems



Capture requirements

Explore designs

Generate tests

Regression test

Maintain evolve

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Maximize

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Maximize

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Maximize

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

Minimize

Cost

Development time

Maximize

Satisfaction

Fairness

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Cost

Development time

Maximize

Satisfaction

Fairness

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Cost

Development time

Maximize

Satisfaction

Fairness

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Number of test

Development time

Maximize

Satisfaction

Fairness

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Number of test
Execution time

Maximize

Satisfaction
Fairness

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Number of test

Execution time

Maximize

Code coverage

Fairness

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

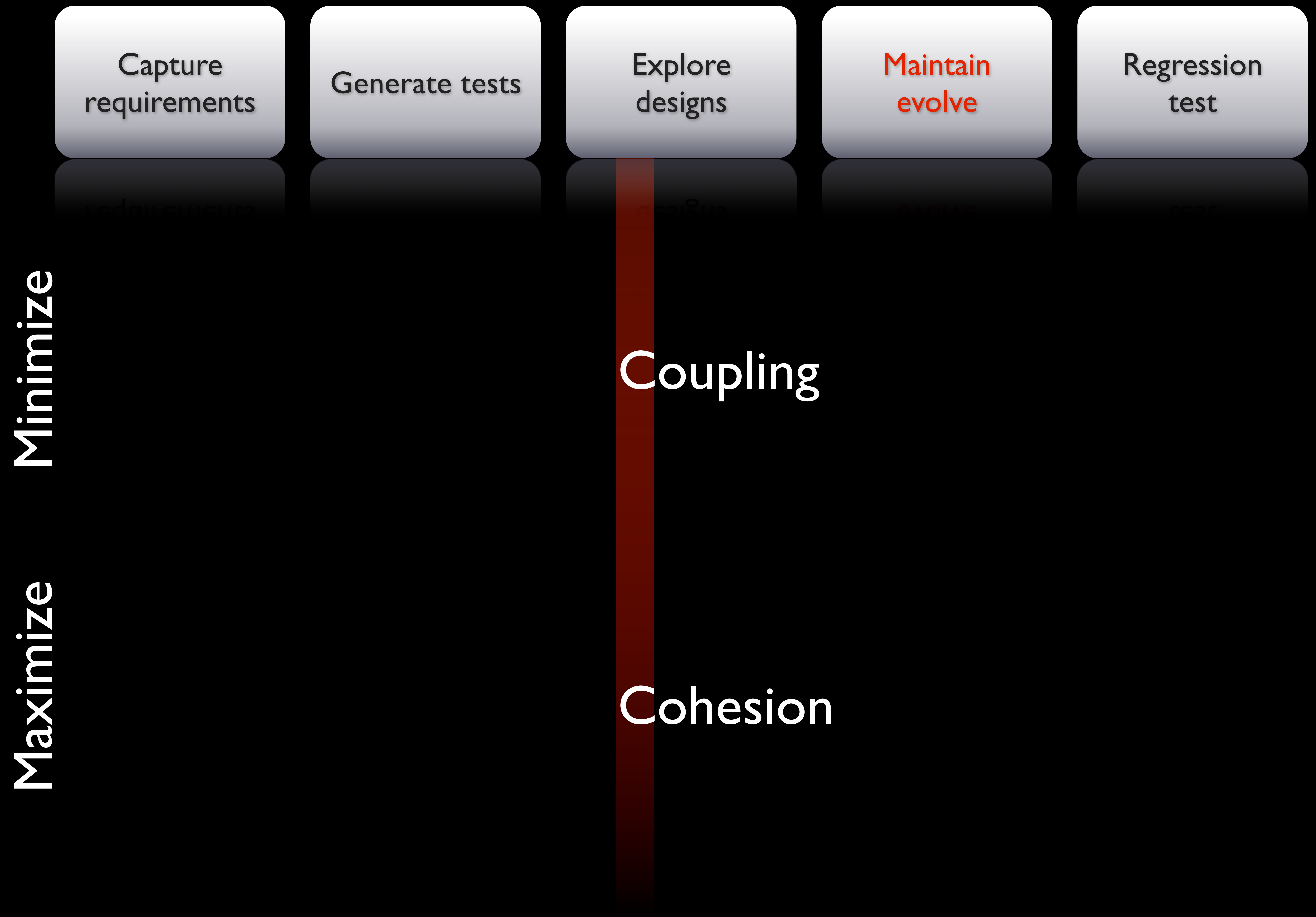
Regression test

Minimize

Coupling

Maximize

Cohesion



Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Coupling

Maximize

Cohesion

Capture requirements

Generate tests

Explore designs

Maintain evolve

Regression test

Minimize

Coupling

Maximize

Cohesion

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

Minimize

Number of test
Execution time

Maximize

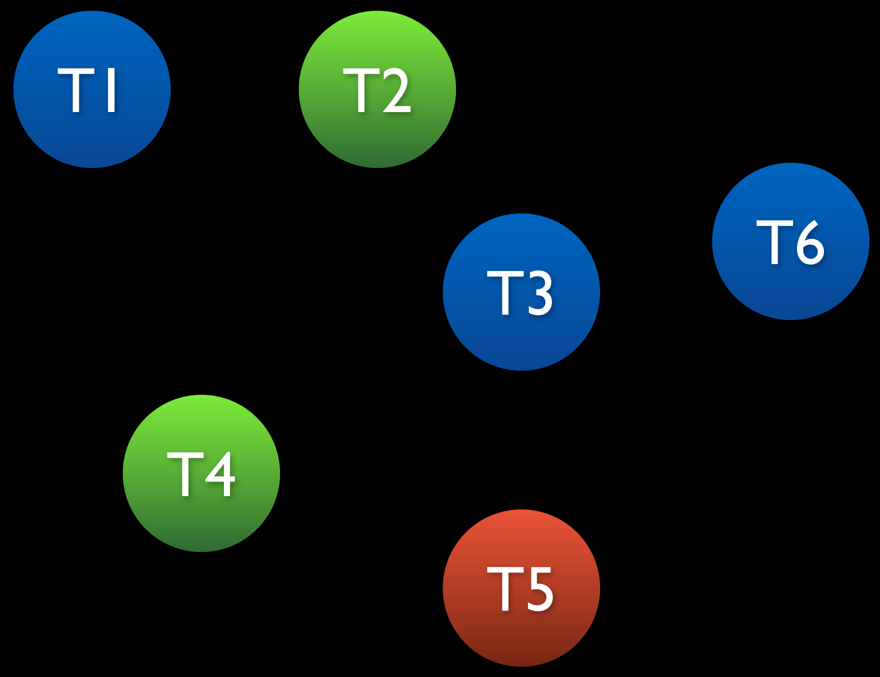
Code coverage
Fault coverage

Human in the Loop Regression Testing

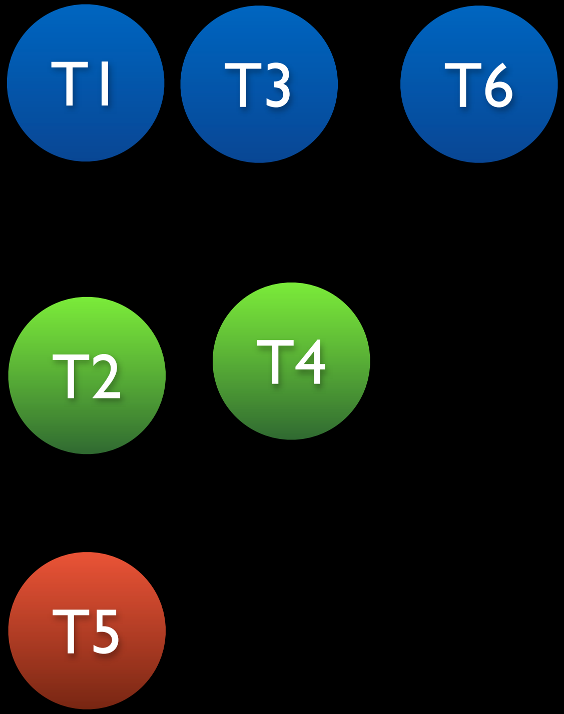
“*I don’t agree. This is the way it should be!*”

“*We need to prioritise business concerns*”

Interleaved Clusters Prioritisation

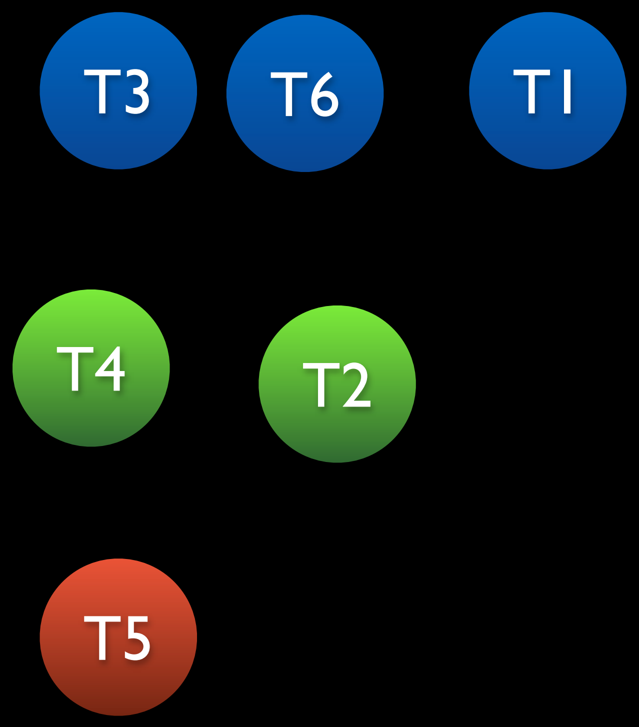


Interleaved Clusters Prioritisation



Cluster

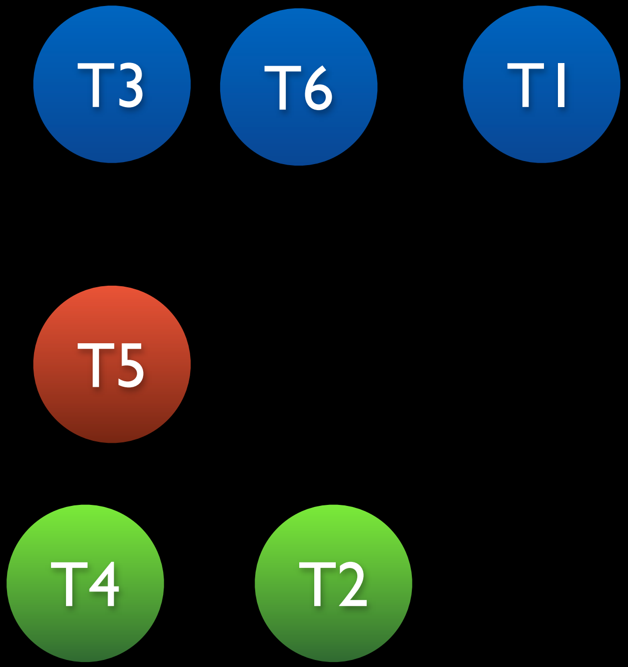
Interleaved Clusters Prioritisation



Cluster

Intra-cluster Prioritisation

Interleaved Clusters Prioritisation

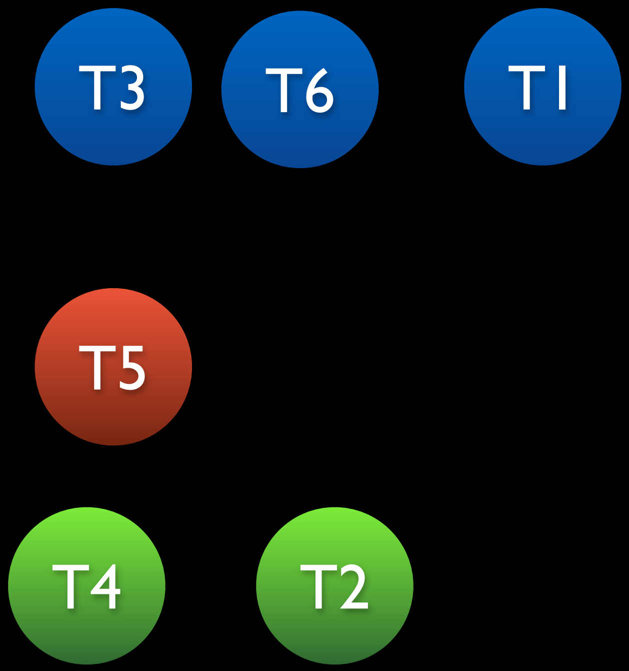


Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaved Clusters Prioritisation



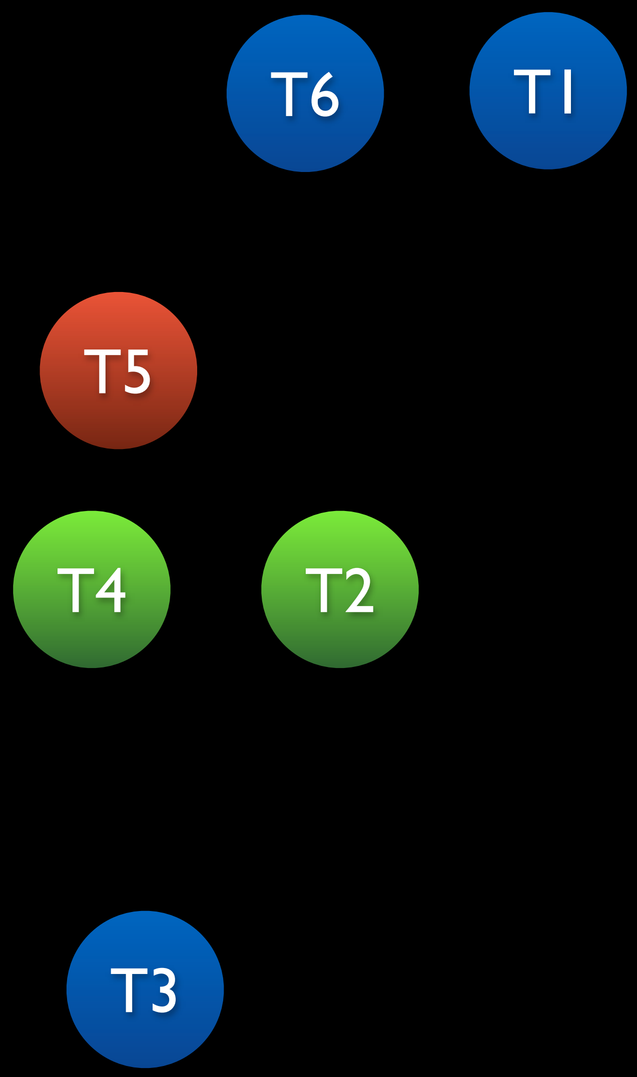
Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters

Interleaved Clusters Prioritisation



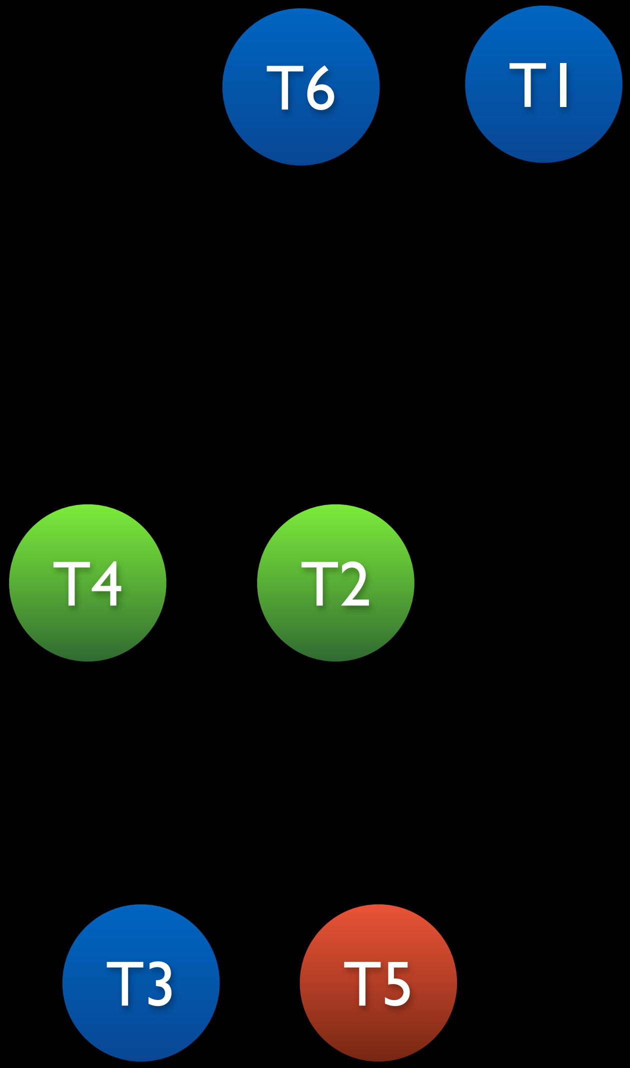
Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters

Interleaved Clusters Prioritisation



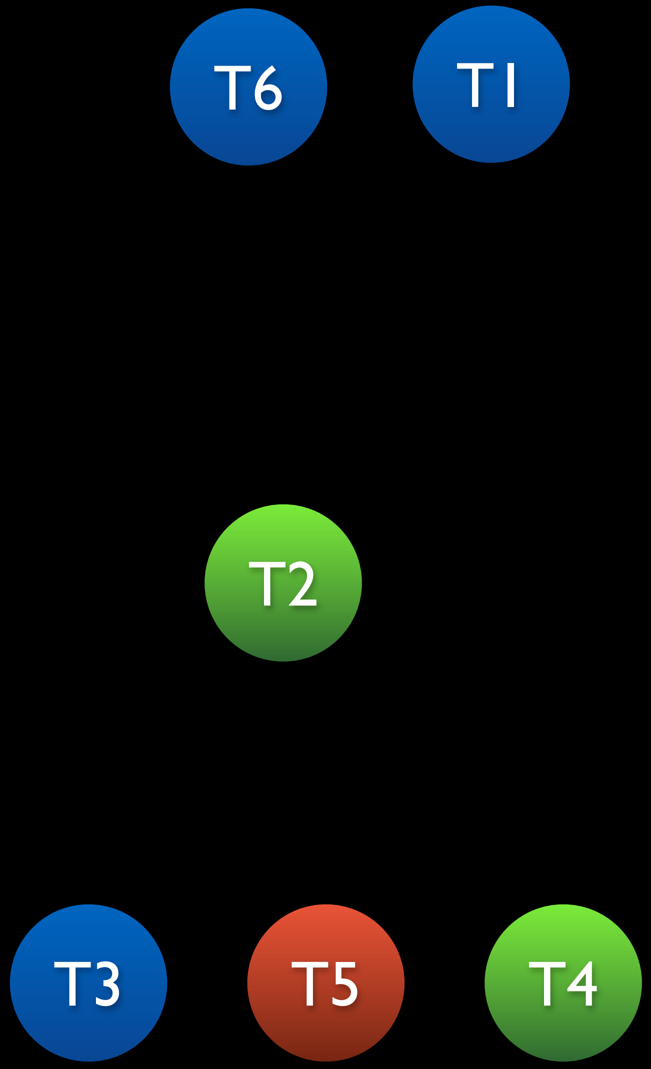
Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters

Interleaved Clusters Prioritisation



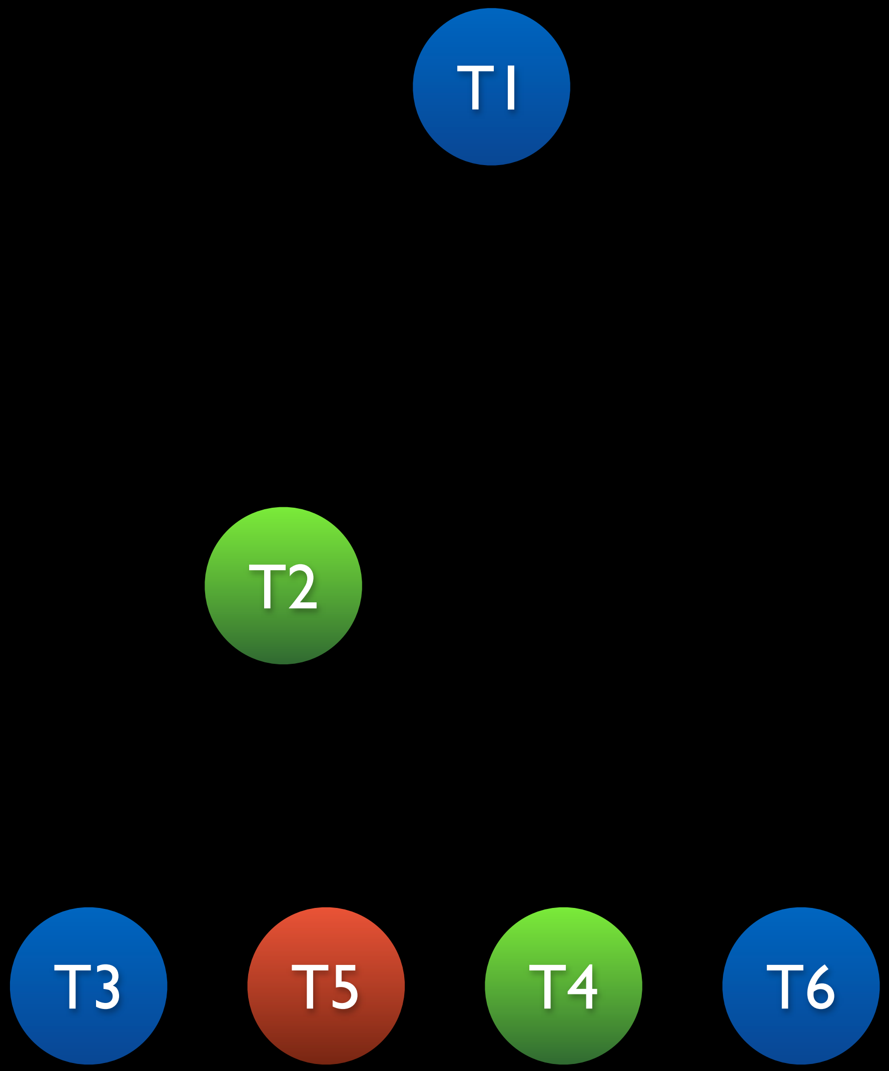
Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters

Interleaved Clusters Prioritisation



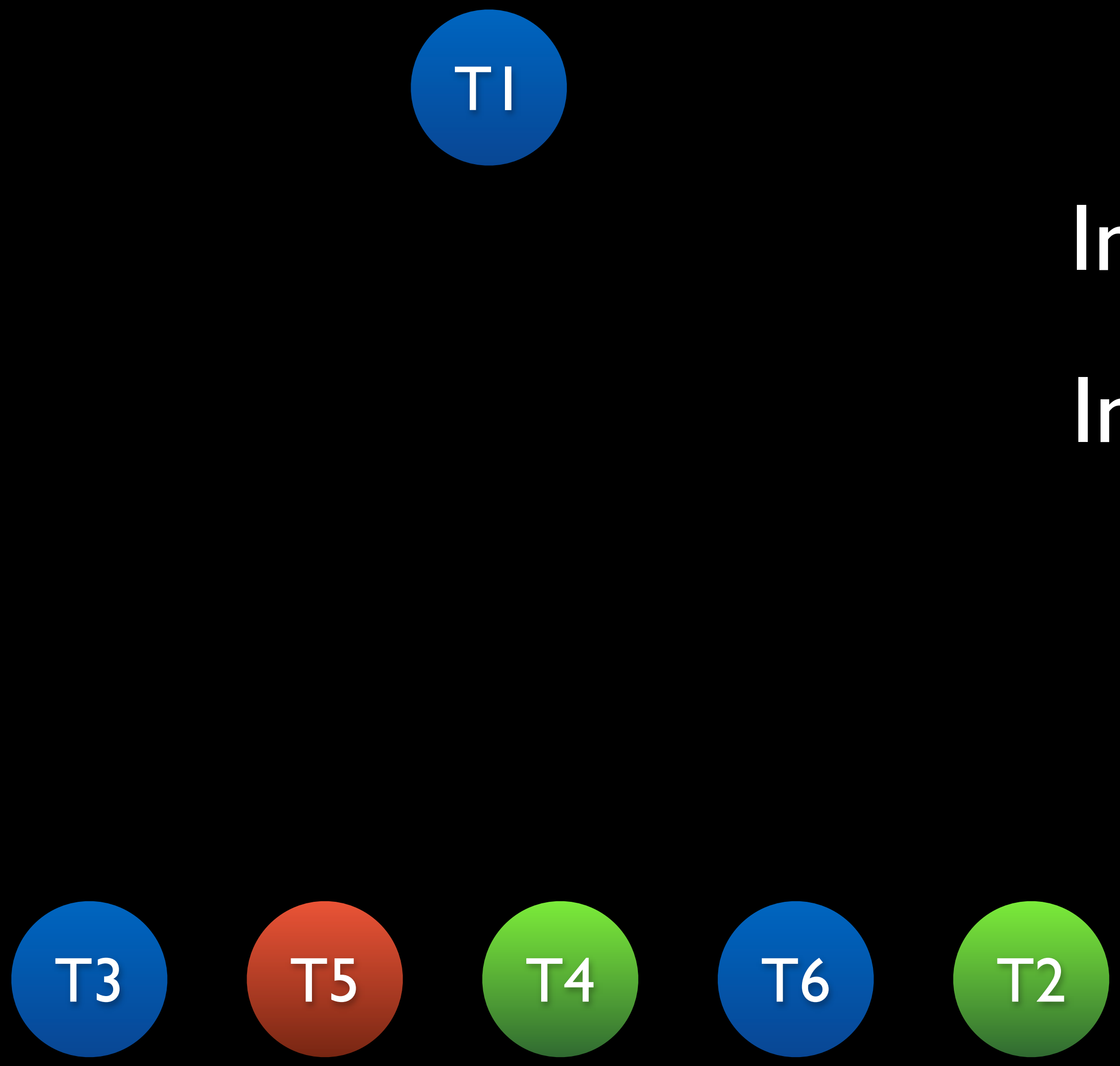
Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters

Interleaved Clusters Prioritisation



Cluster

Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters

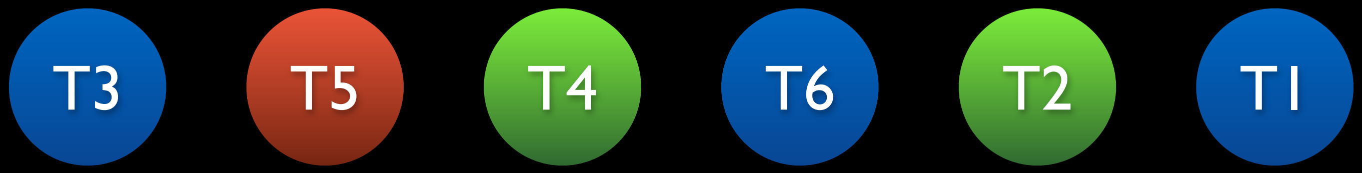
Interleaved Clusters Prioritisation

Cluster

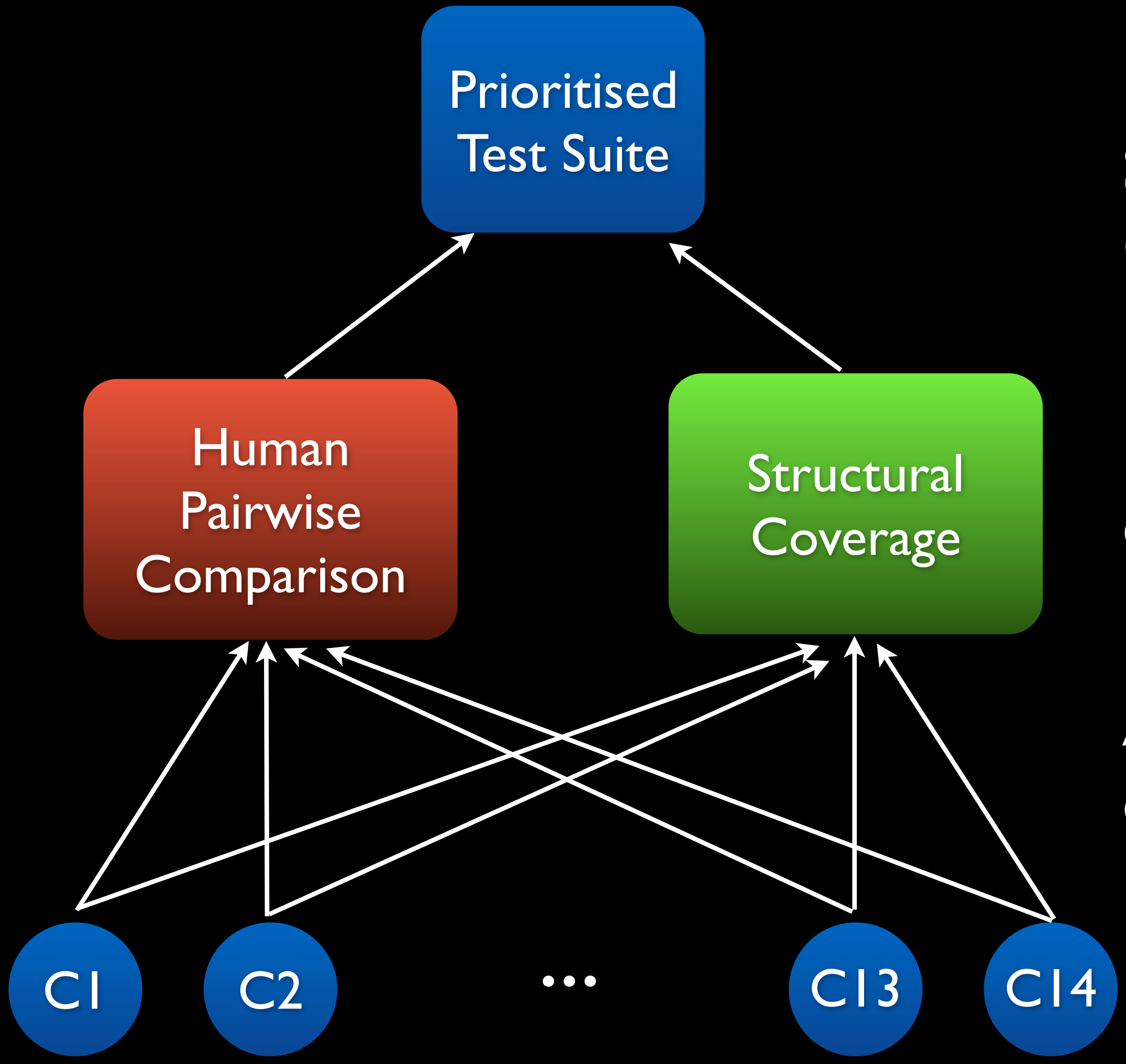
Intra-cluster Prioritisation

Inter-cluster Prioritisation

Interleaving Clusters



Experimental Setup

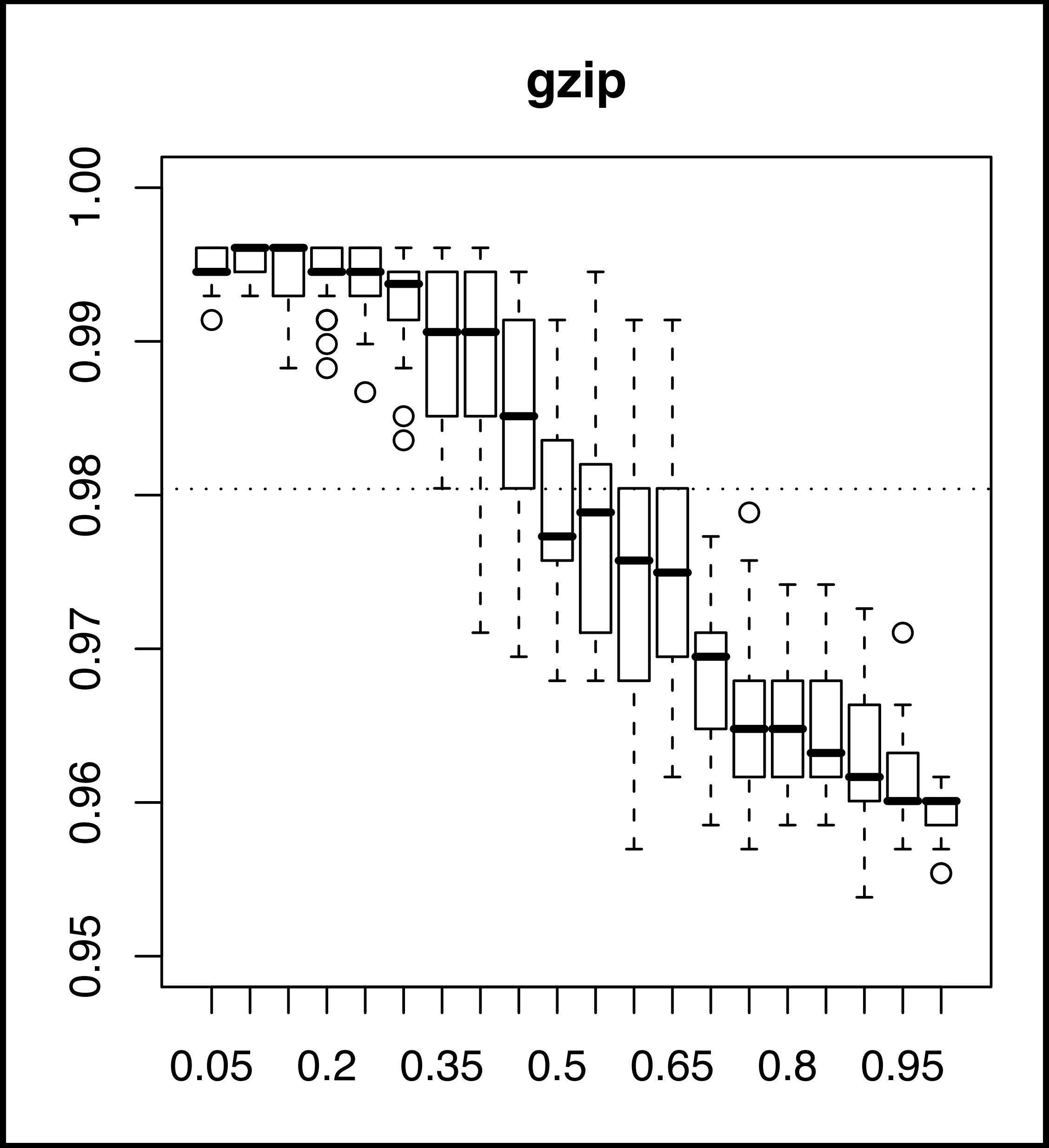


Simple Agglomerative Hierarchical Clustering (k=14)

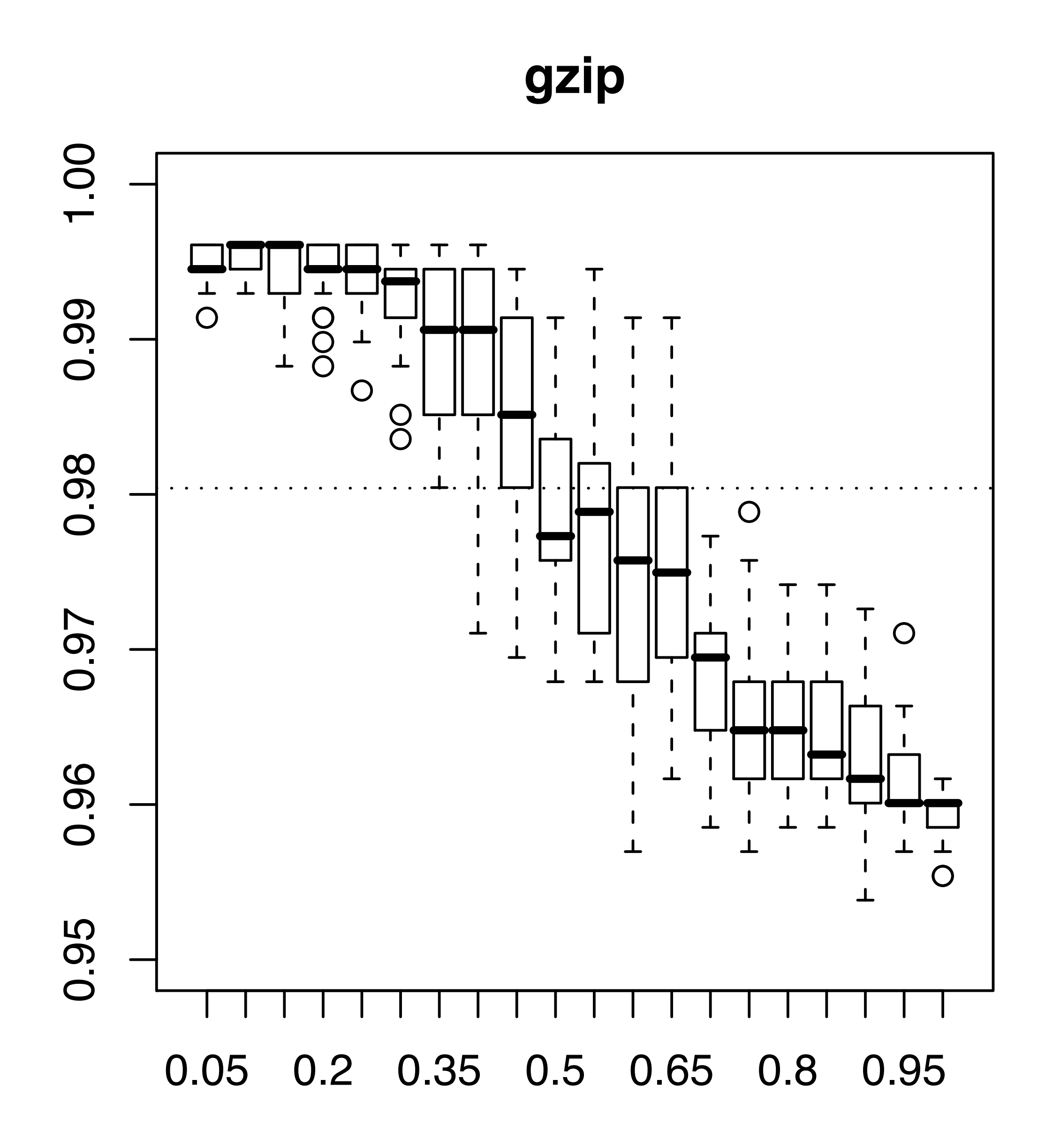
Hamming distance between stmt. coverage as dissimilarity metric

A human user model with controlled error rate

Tolerance

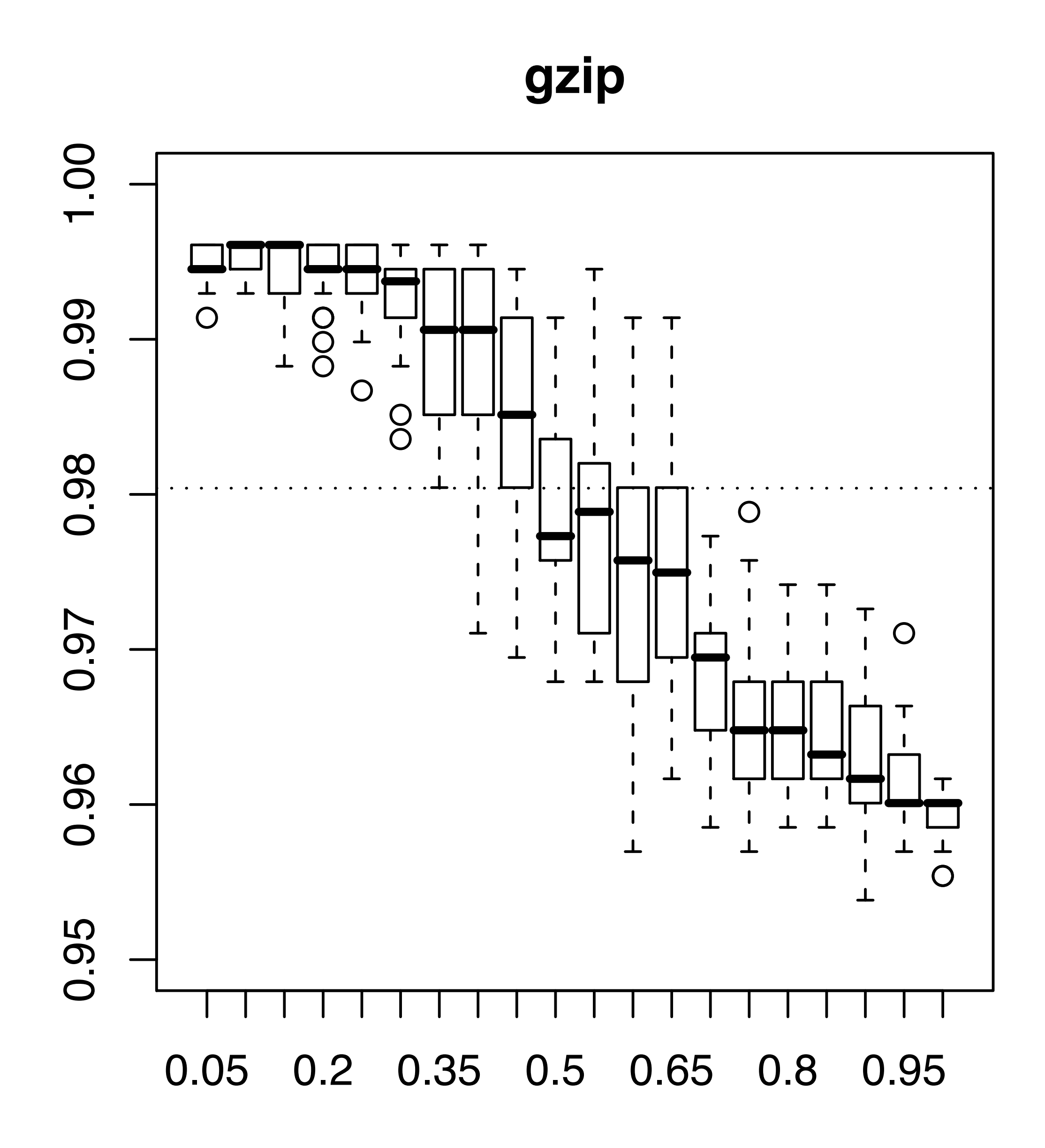


Tolerance



This is what we initially expected to see.

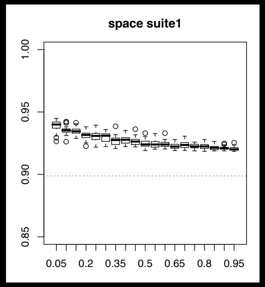
Tolerance

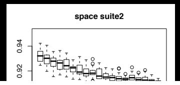


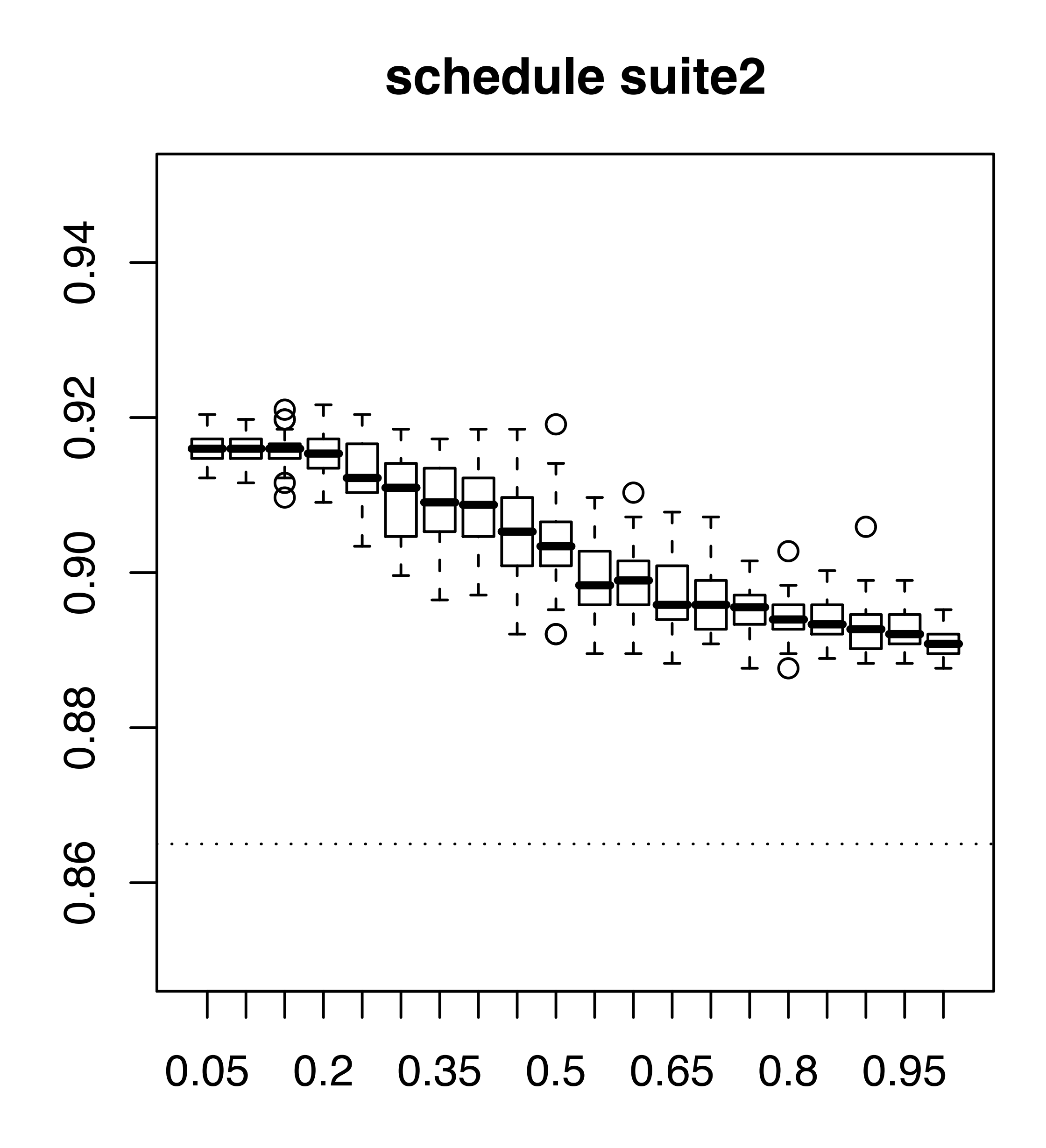
This is what we initially expected to see.

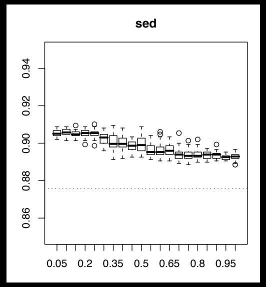
But...

Tolerance



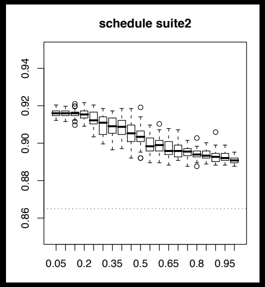


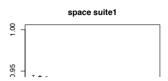


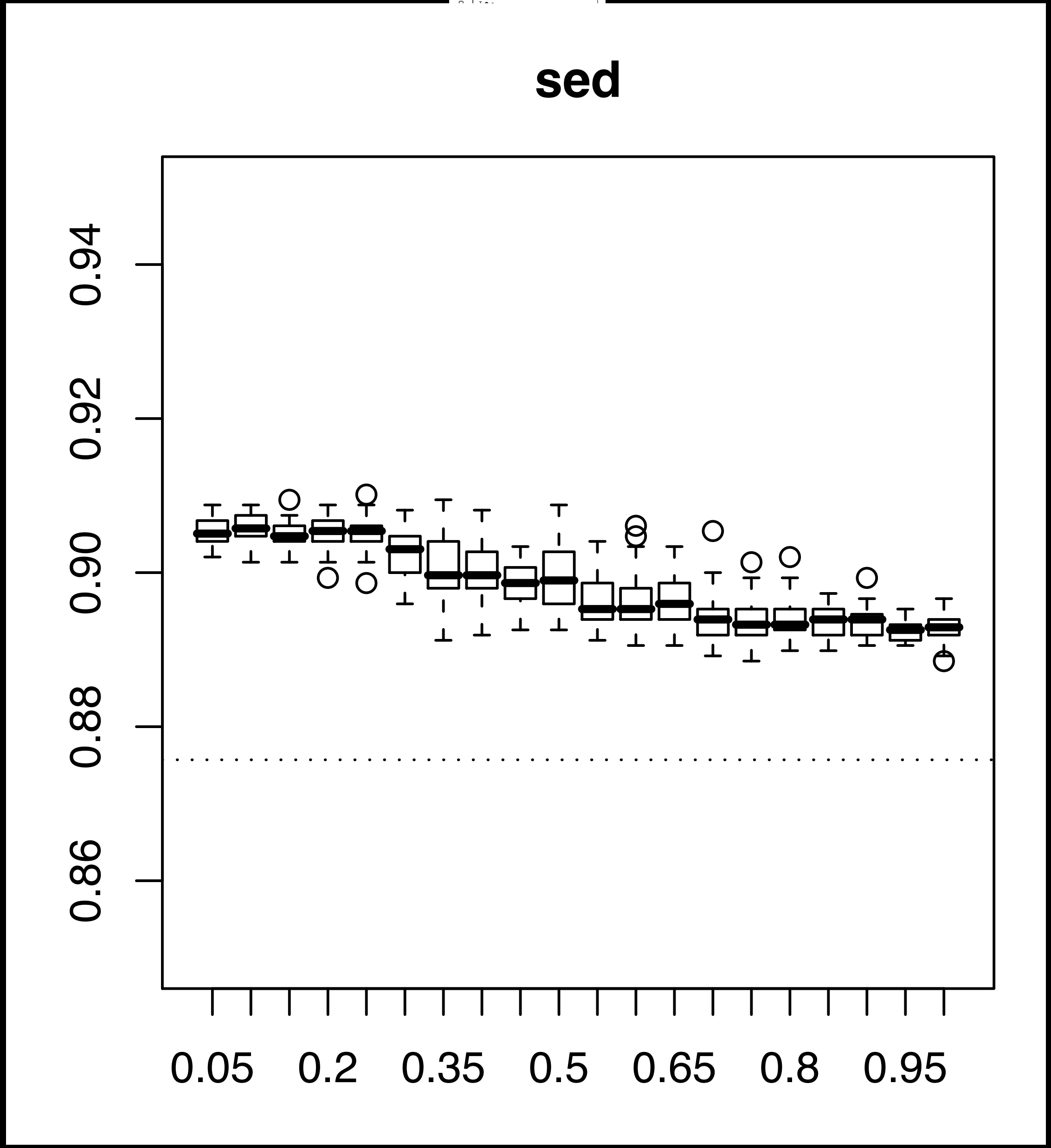


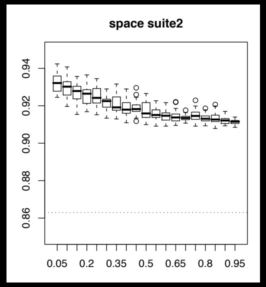
Some test suites are very resilient to errors

Tolerance



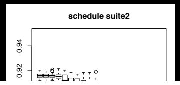


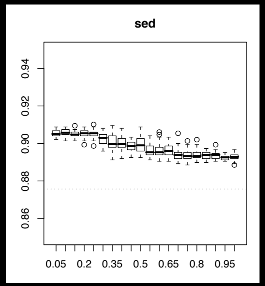


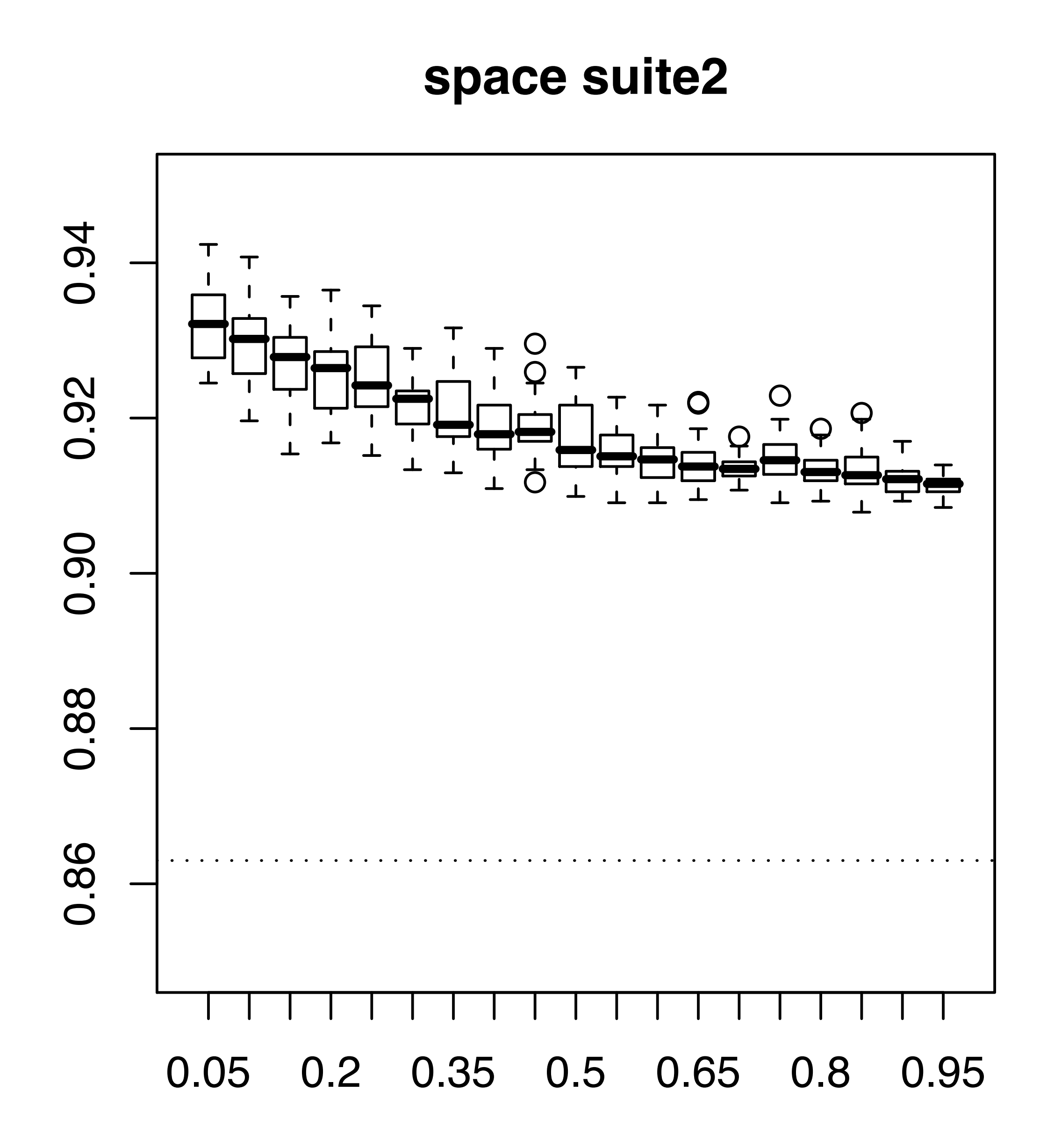


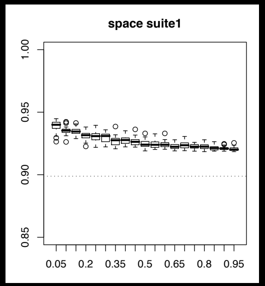
Some test suites are very resilient to errors

Tolerance



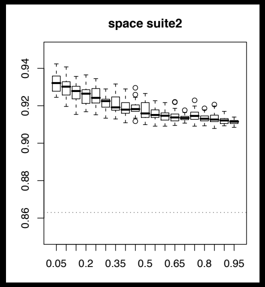


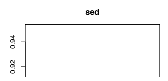


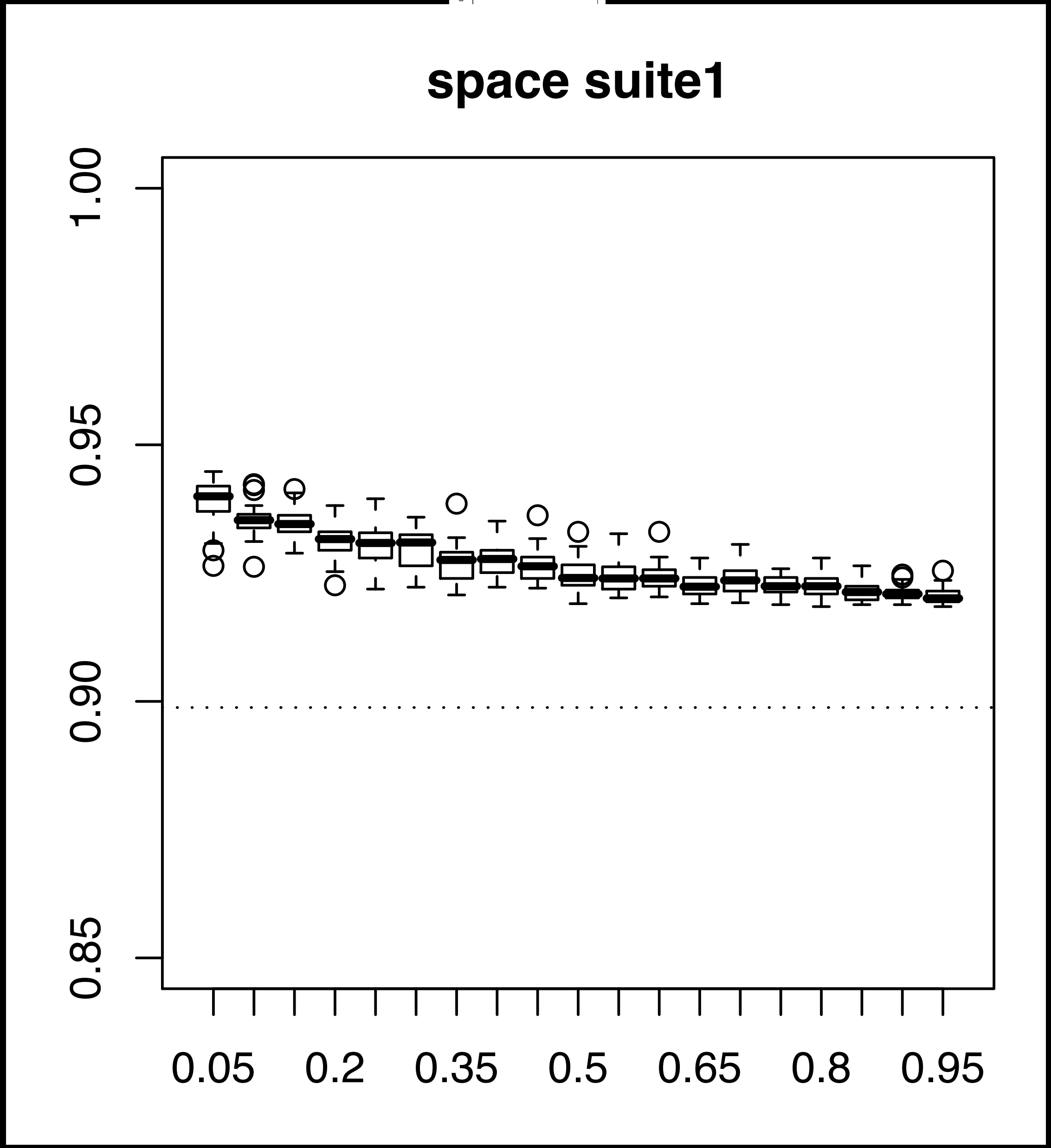


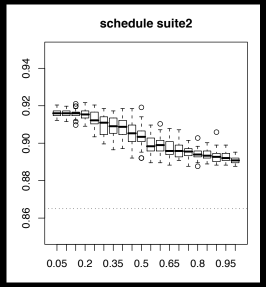
Some test suites are very resilient to errors

Tolerance





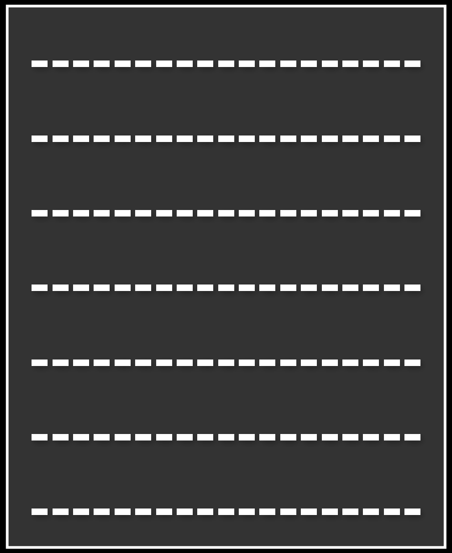




Some test suites are very resilient to errors

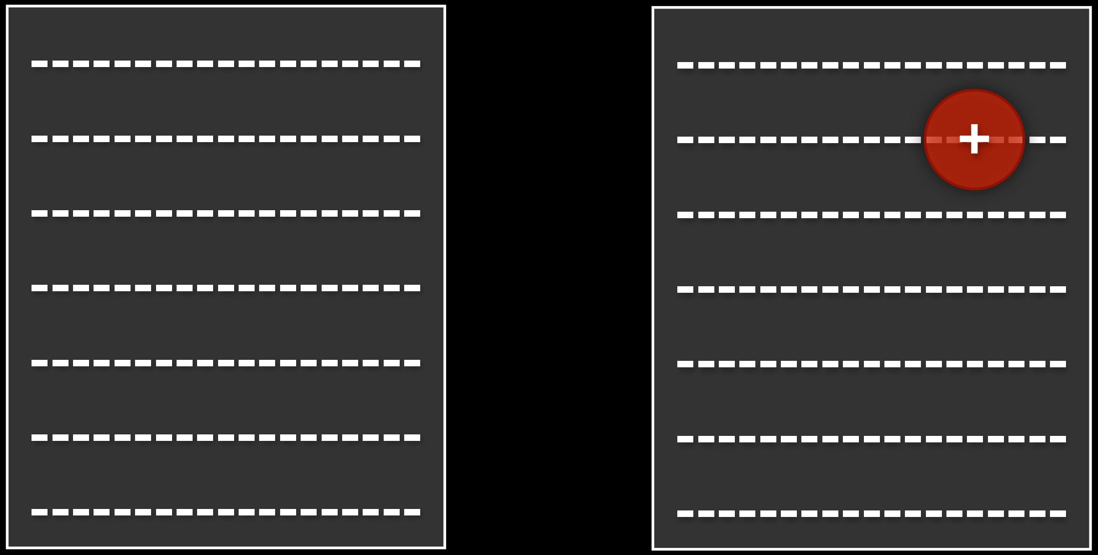
Optimising
Mutation Testing

FOMs and HOMs



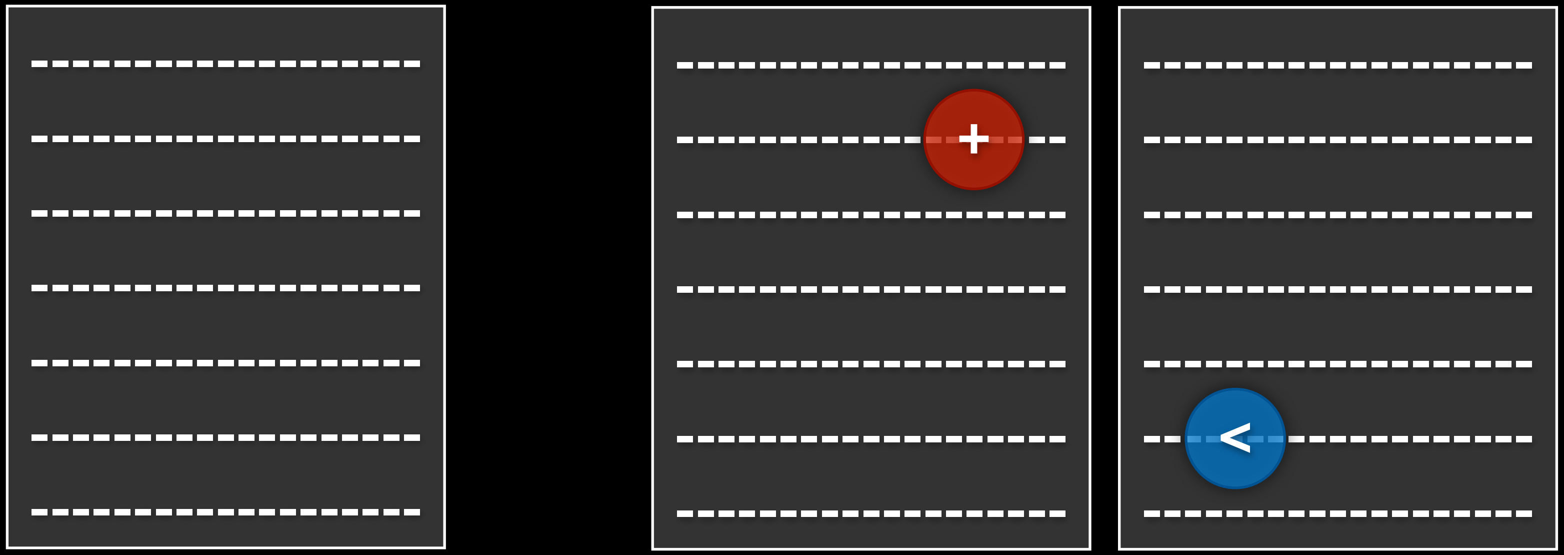
Original Program

FOMs and HOMs



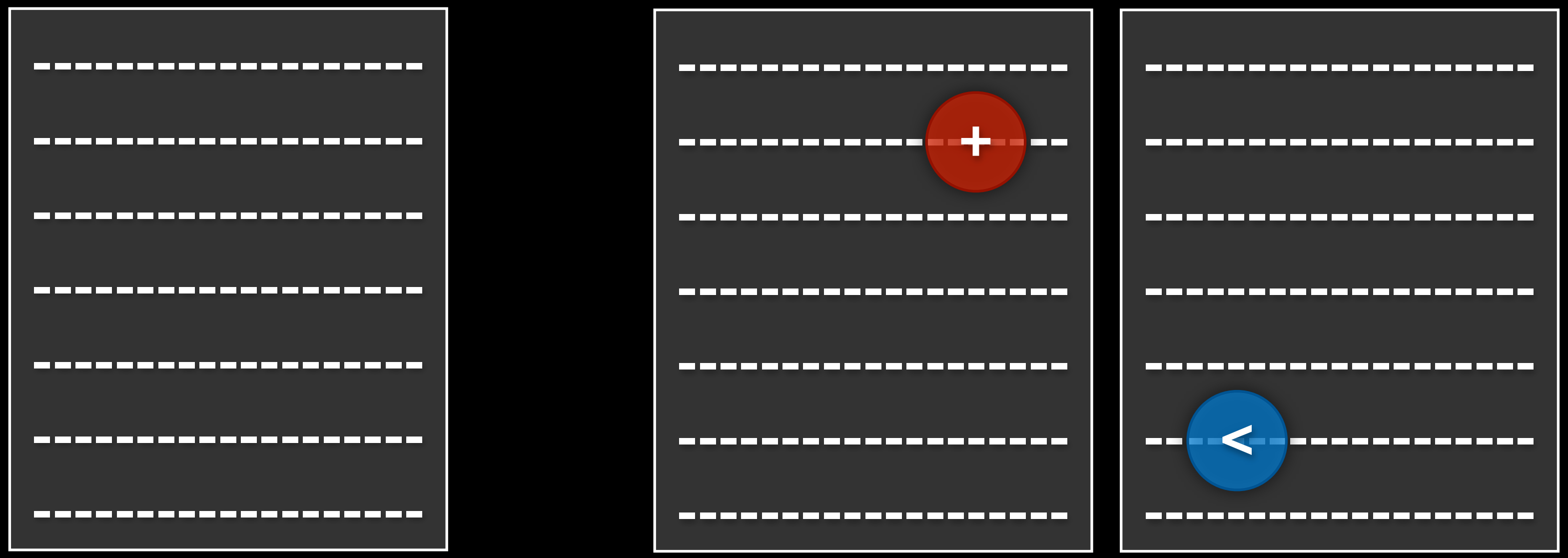
Original Program

FOMs and HOMs



Original Program

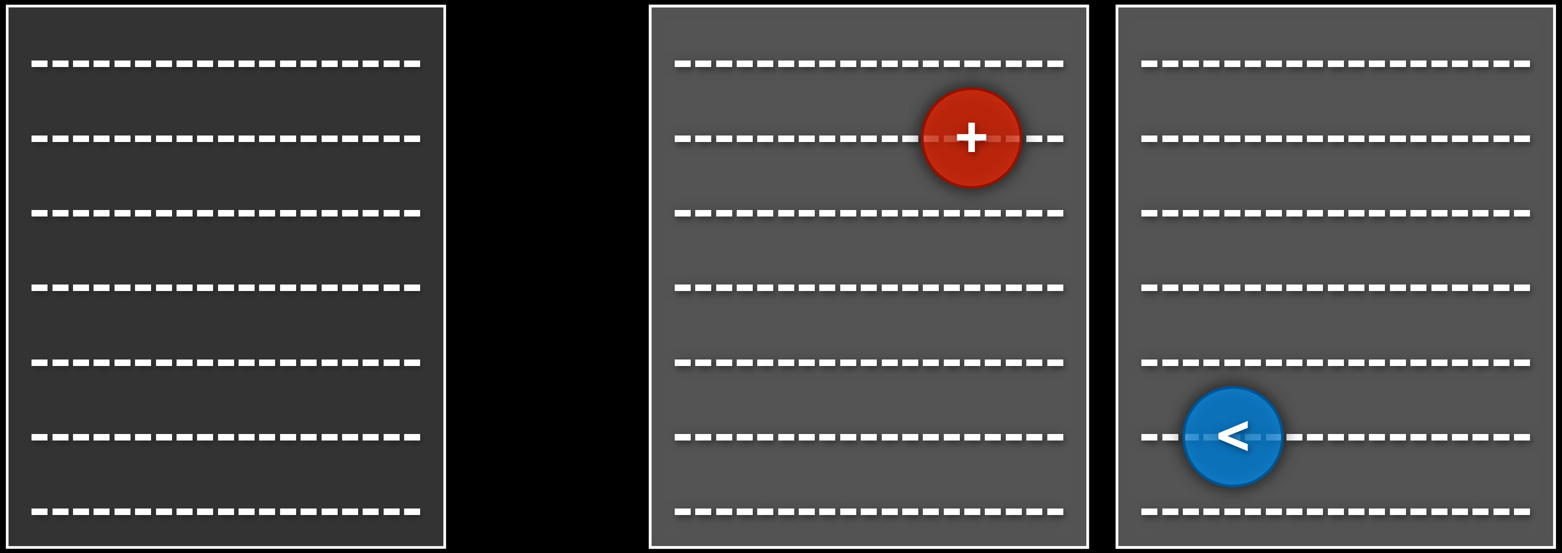
FOMs and HOMs



Original Program

First Order Mutants

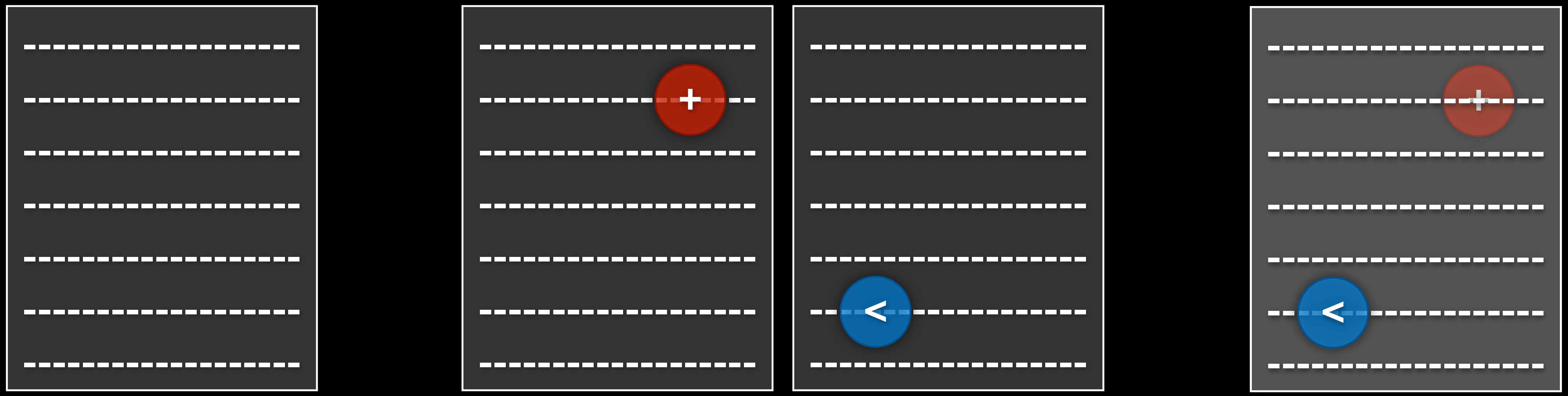
FOMs and HOMs



Original Program

First Order Mutants

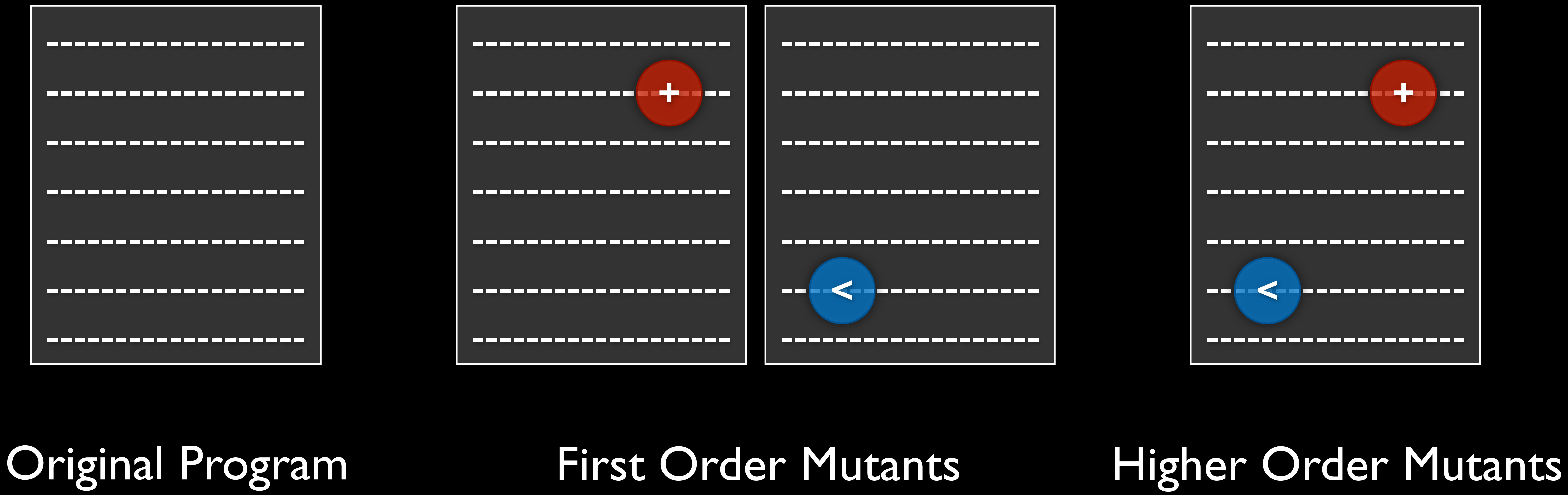
FOMs and HOMs



Original Program

First Order Mutants

FOMs and HOMs

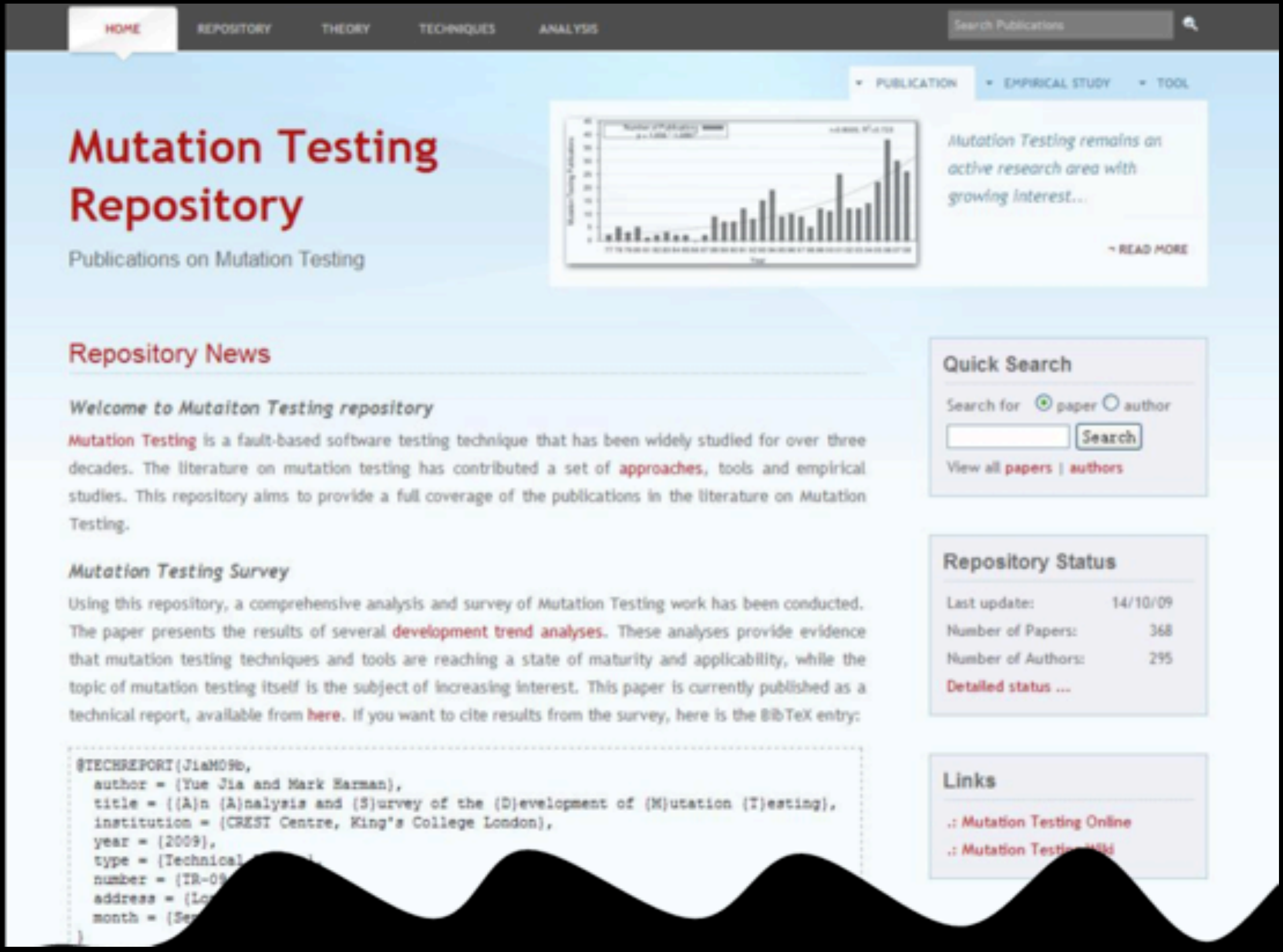


Original Program

First Order Mutants

Higher Order Mutants

Mutation Testing Repository



Mutation Testing Repository

Publications on Mutation Testing

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

Repository News

Welcome to Mutaiton 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{JiaM09b,
  author = {Yue Jia and Mark Harman},
  title = {{A}n {A}nalysis and {S}urvey of the {D}evelopment of {M}utation {T}esting},
  institution = {CREST Centre, King's College London},
  year = {2009},
  type = {Technical
  number = {TR-09
  address = {Lo
  month = {Se
}
```

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

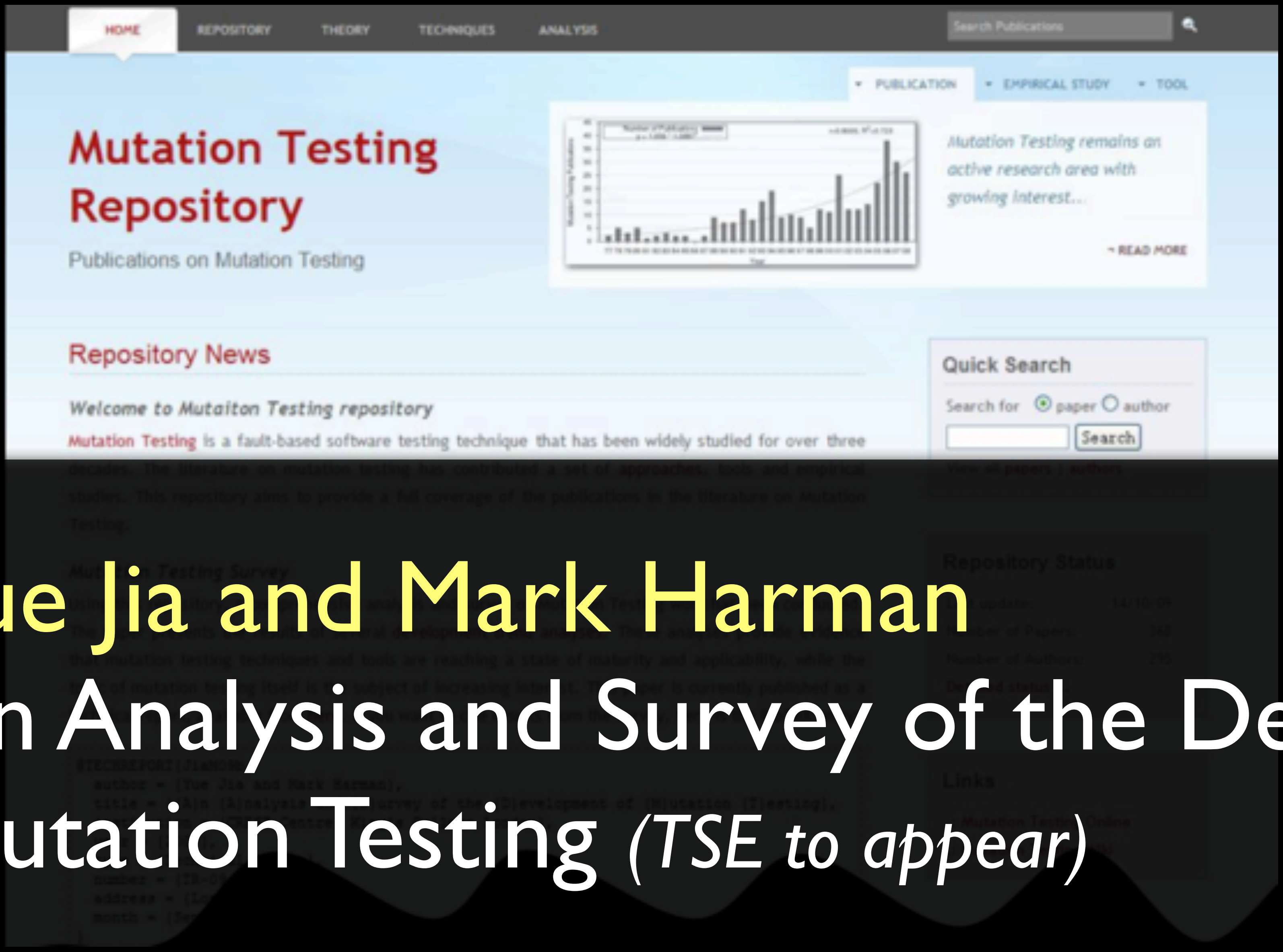
Detailed status ...

Links

.: Mutation Testing Online

.: Mutation Testi Wiki

Mutation Testing Repository



Mutation Testing Repository

Publications on Mutation Testing

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

Repository News

Welcome to Mutaiton 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.

Quick Search

Search for ● paper ○ author

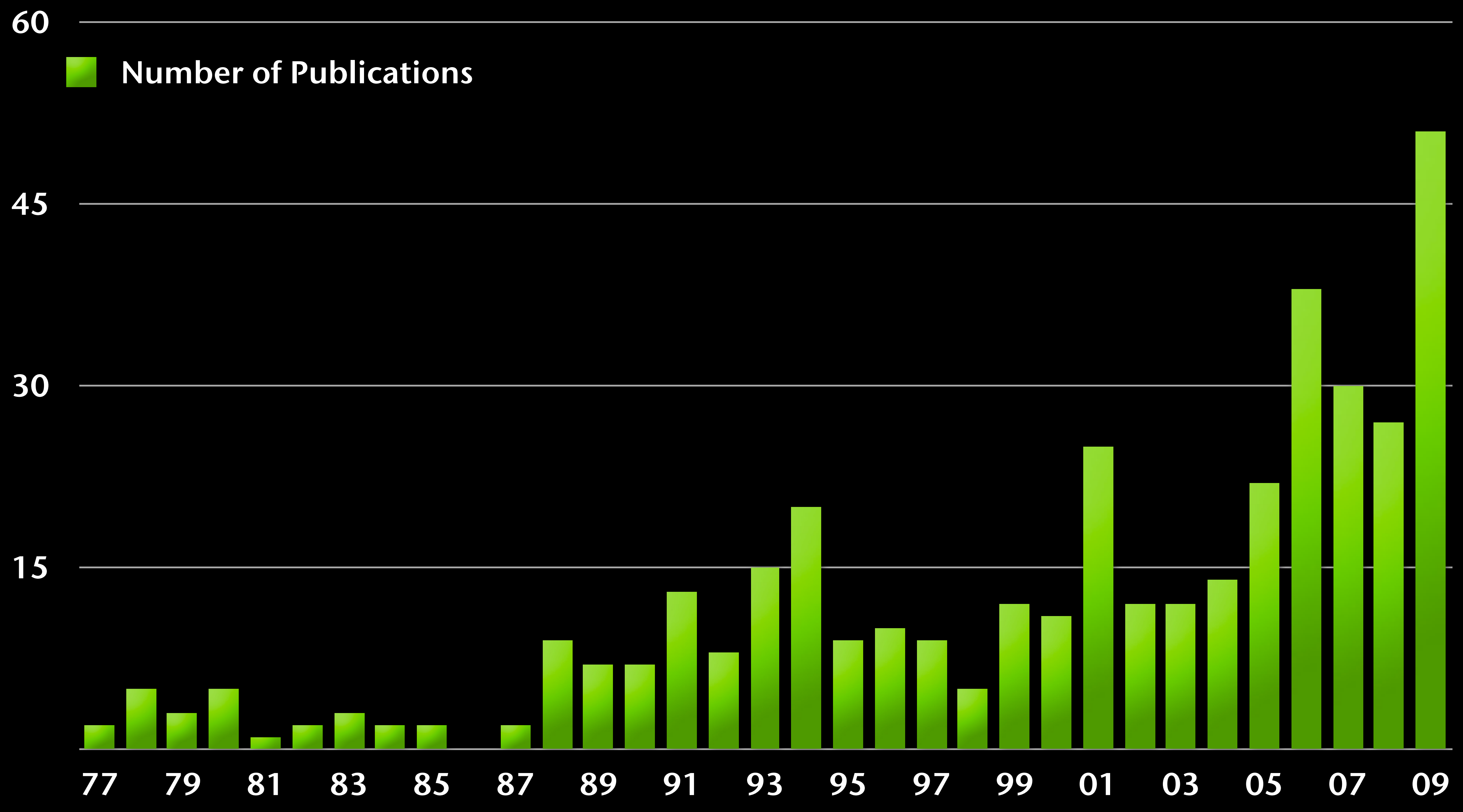
View all papers | authors

Repository Status

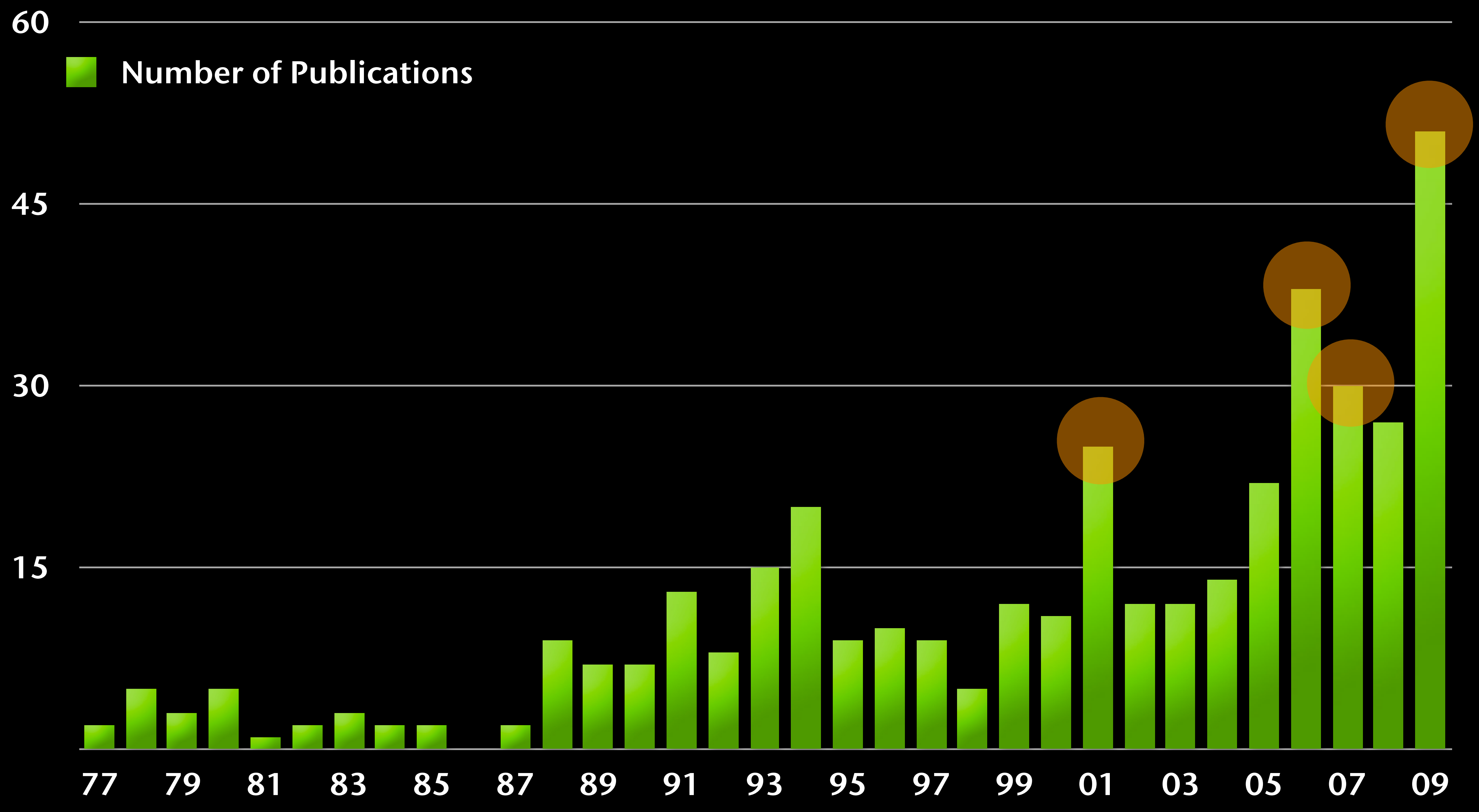
Yue Jia and Mark Harman

An Analysis and Survey of the Development of Mutation Testing *(TSE to appear)*

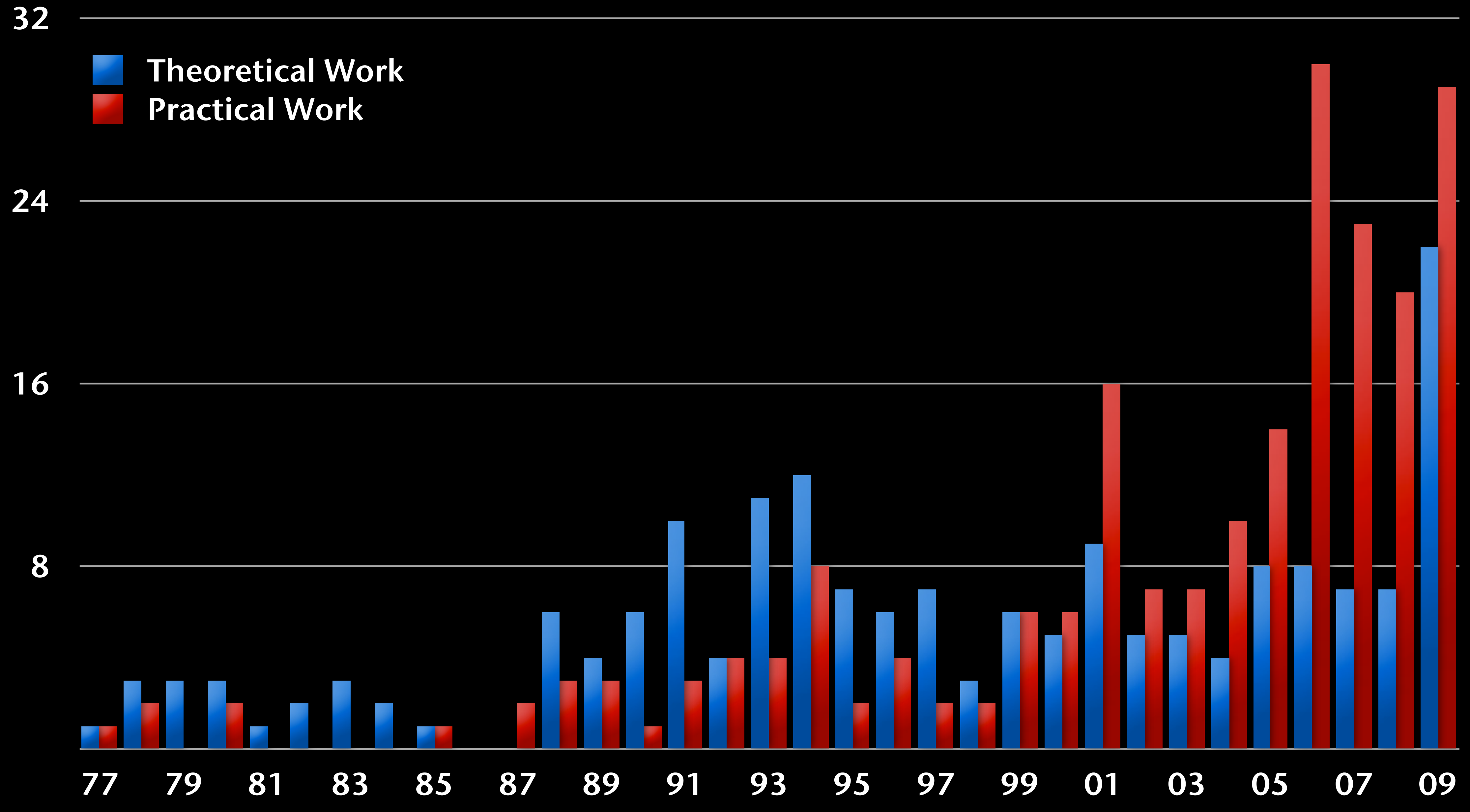
Research Publications



Research Publications



Research Publications



Applications

Applications

Ada

Algebraic Specification

C#

Network Protocol

Petri Nets

Security Policy

Fortran

Statecharts

C/C++

Spreadsheets

Java

Aspect Oriented Programming

Web Service

Lustre

SQL

Finite State Machine

PHP

Real Time System

Calculus Specification

Applications

C#

Ada

Algebraic Specification

Security Policy

Network Protocol

Petri Nets

Fortran

Statecharts

C/C++

Spreadsheets

Java

Aspect Oriented Programming

Web Service

Lustre

SQL

Finite State Machine

PHP

Real Time System

Calculus Specification

Applications

Ada

Algebraic Specification

C#

Network Protocol

Security Policy

Petri Nets

Fortran

Statecharts

C/C++

Spreadsheets

Java

Aspect Oriented Programming

Web Service

Lustre

SQL

Finite State Machine

PHP

Real Time System

Calculus Specification

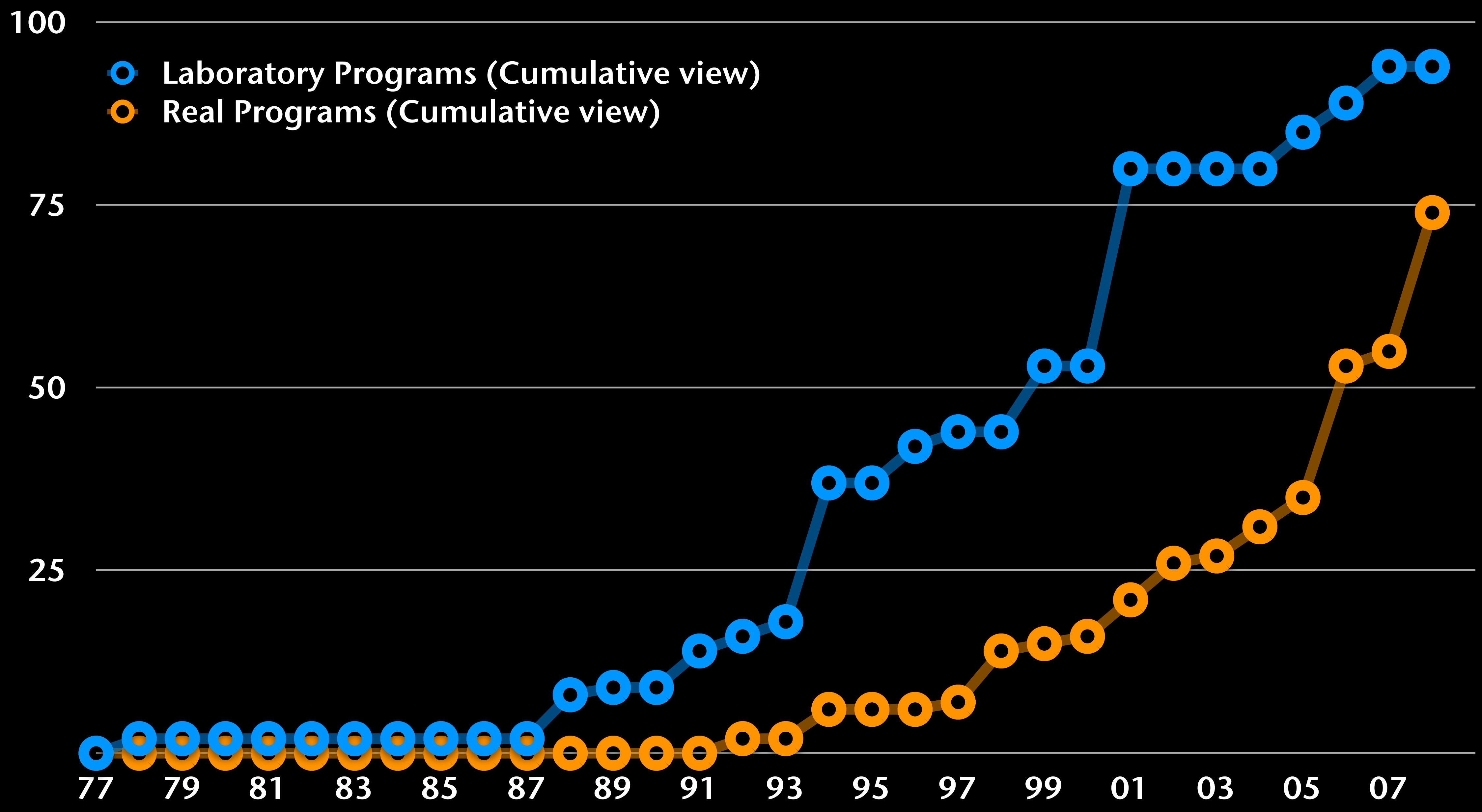
Empirical Studies

Newly applied subject programs

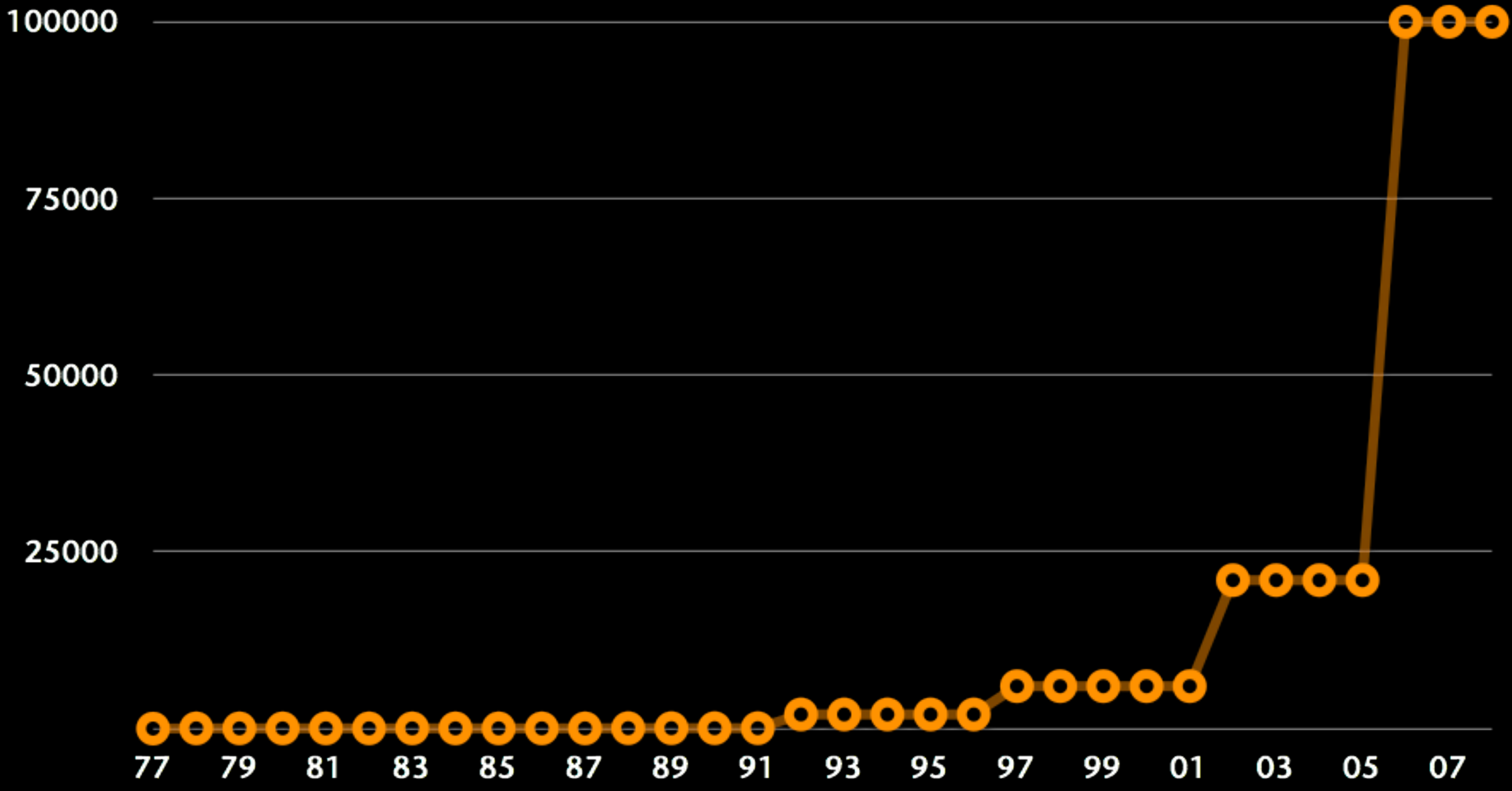
Laboratory programs vs real programs

Size of the largest program

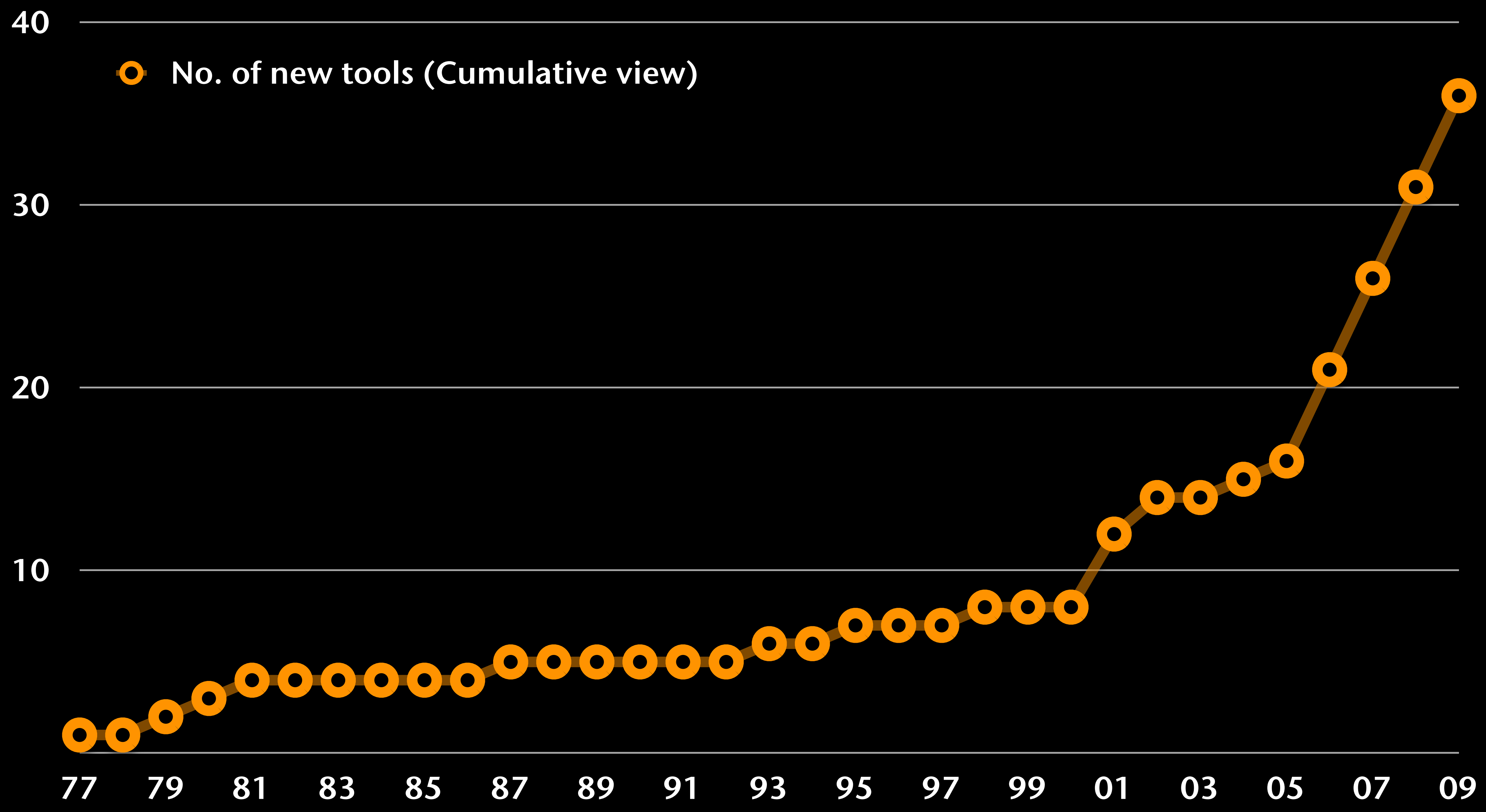
Laboratory vs Real



Largest Program



Tool development



Summary

mutation testing techniques and tools

are reaching a state of

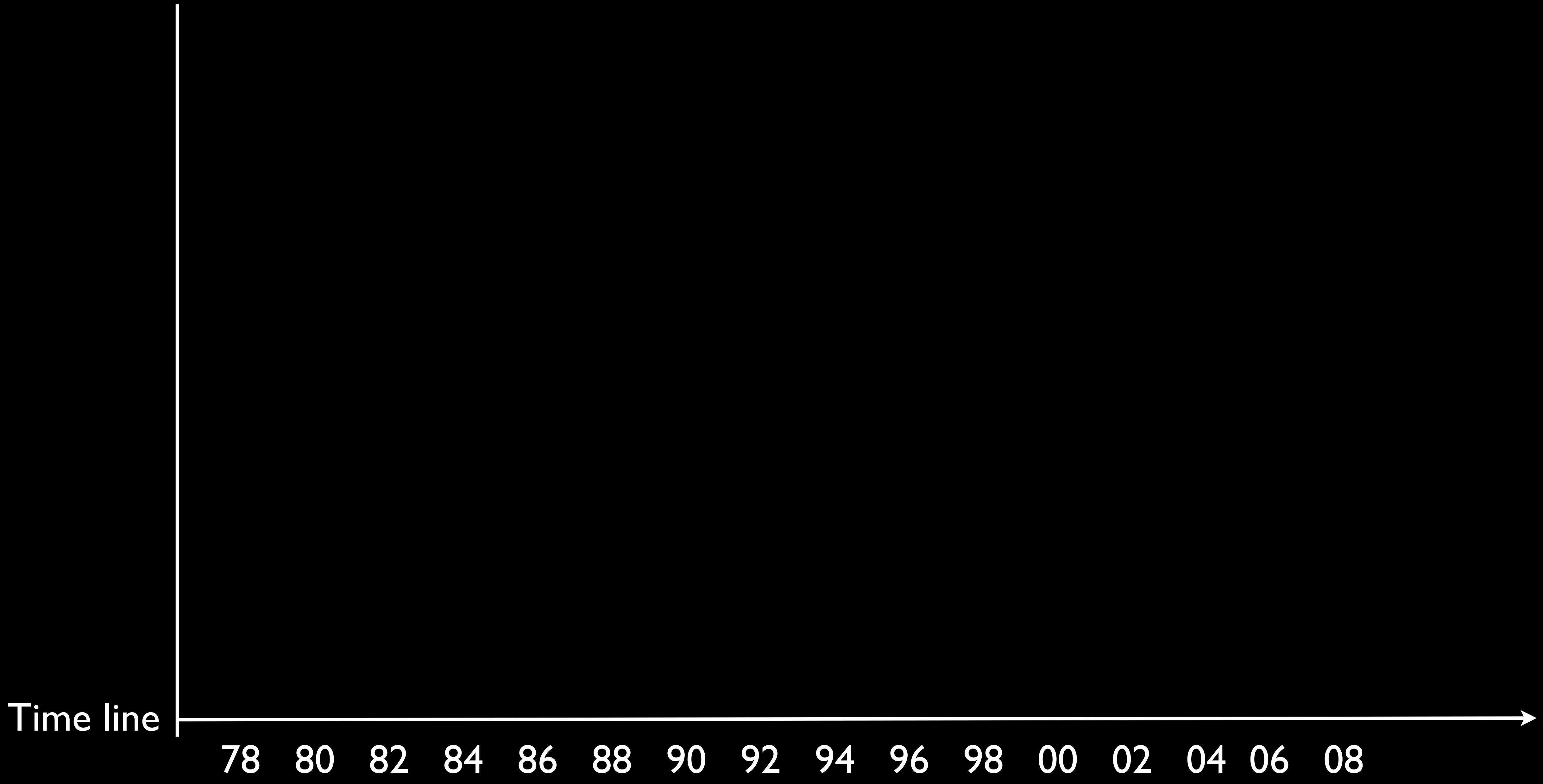
maturity and applicability

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

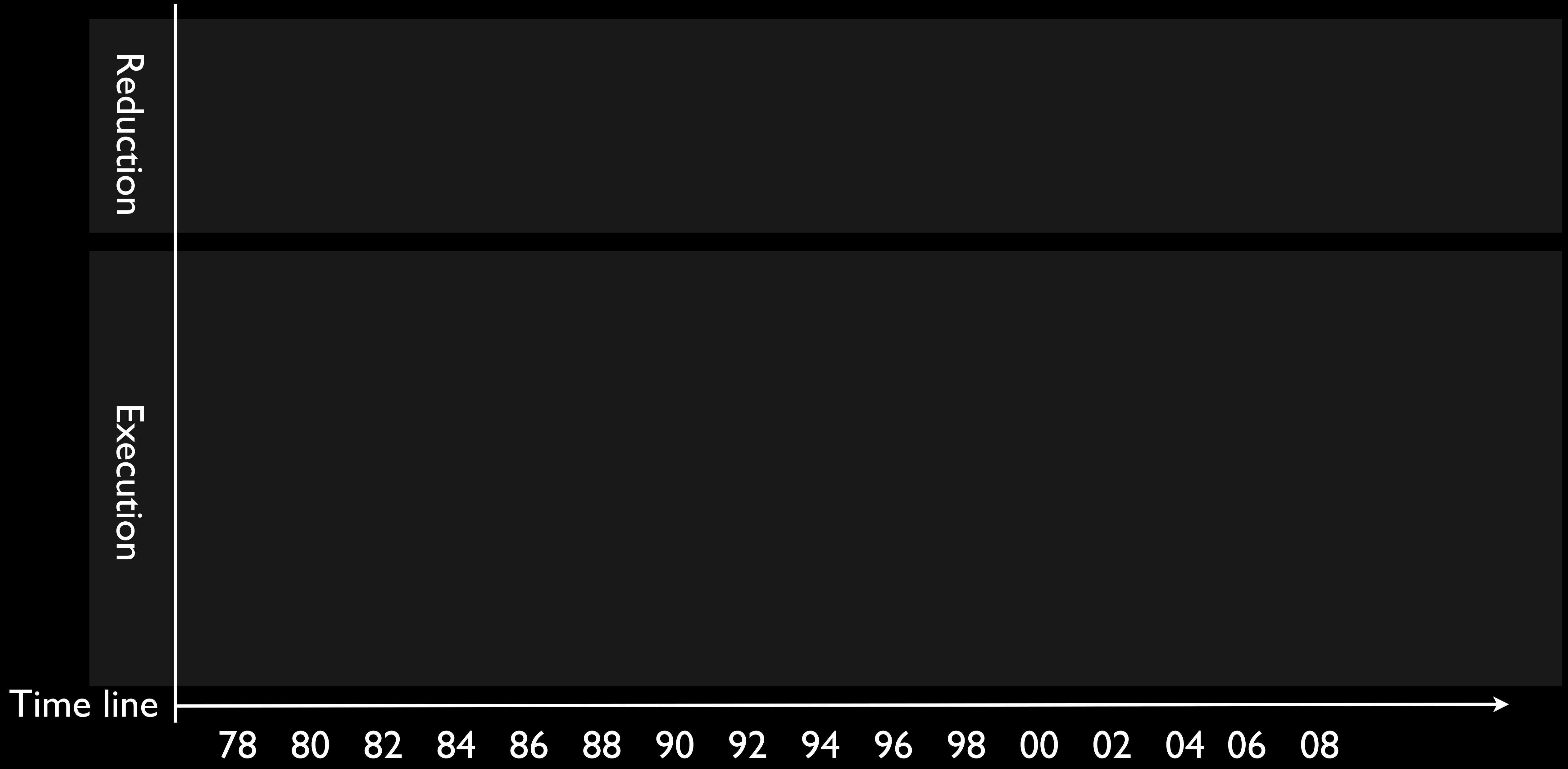
http://www.dcs.kcl.ac.uk/pg/jiayue/repository/

Mutant Reduction Techniques

Mutant Reduction Techniques



Mutant Reduction Techniques



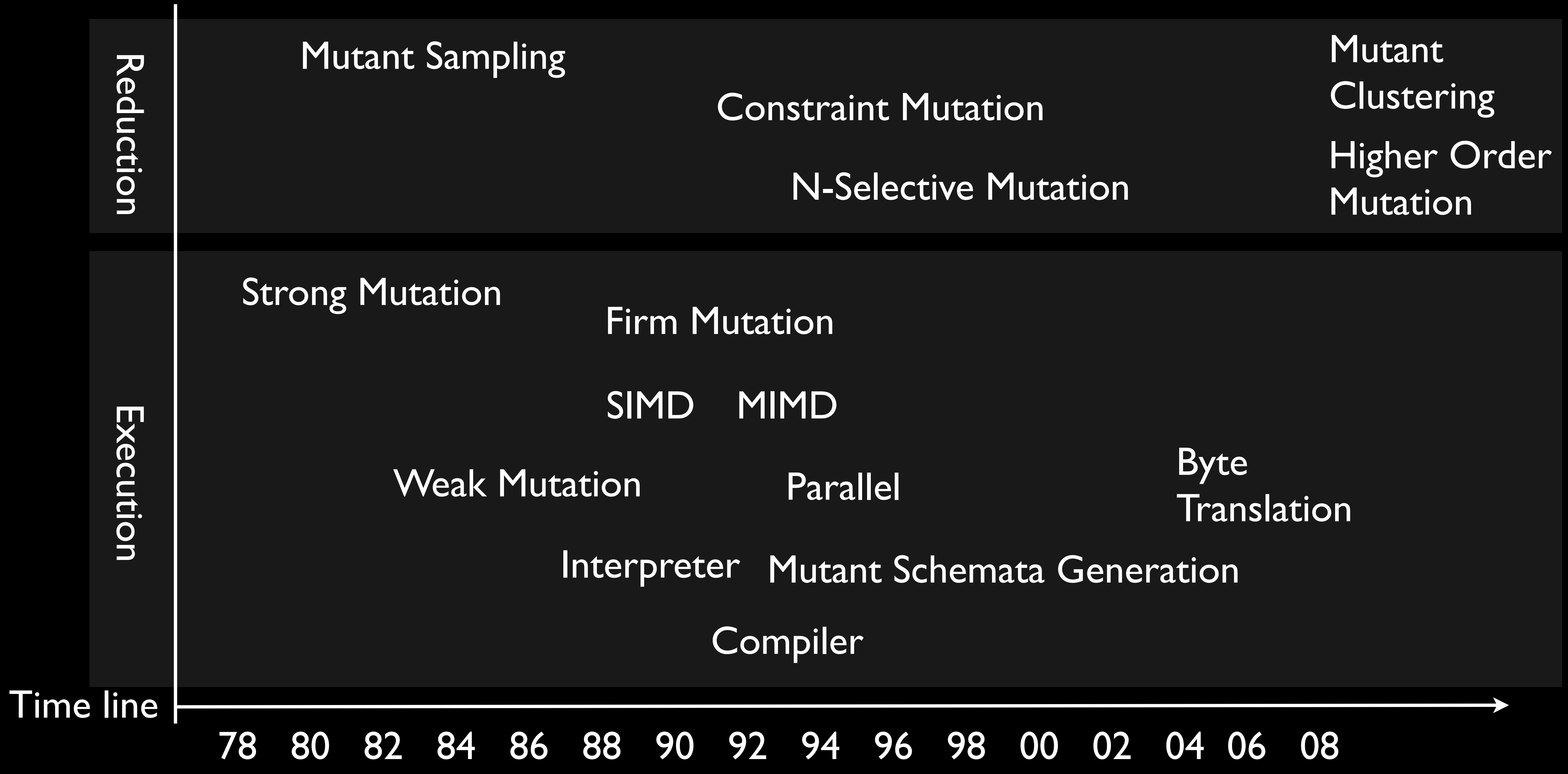
Reduction

Execution

Time line

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

Mutant Reduction Techniques



Reduction

Mutant Sampling

Constraint Mutation

N-Selective Mutation

Mutant Clustering

Higher Order Mutation

Execution

Strong Mutation

Firm Mutation

SIMD MIMD

Weak Mutation

Parallel

Byte Translation

Interpreter

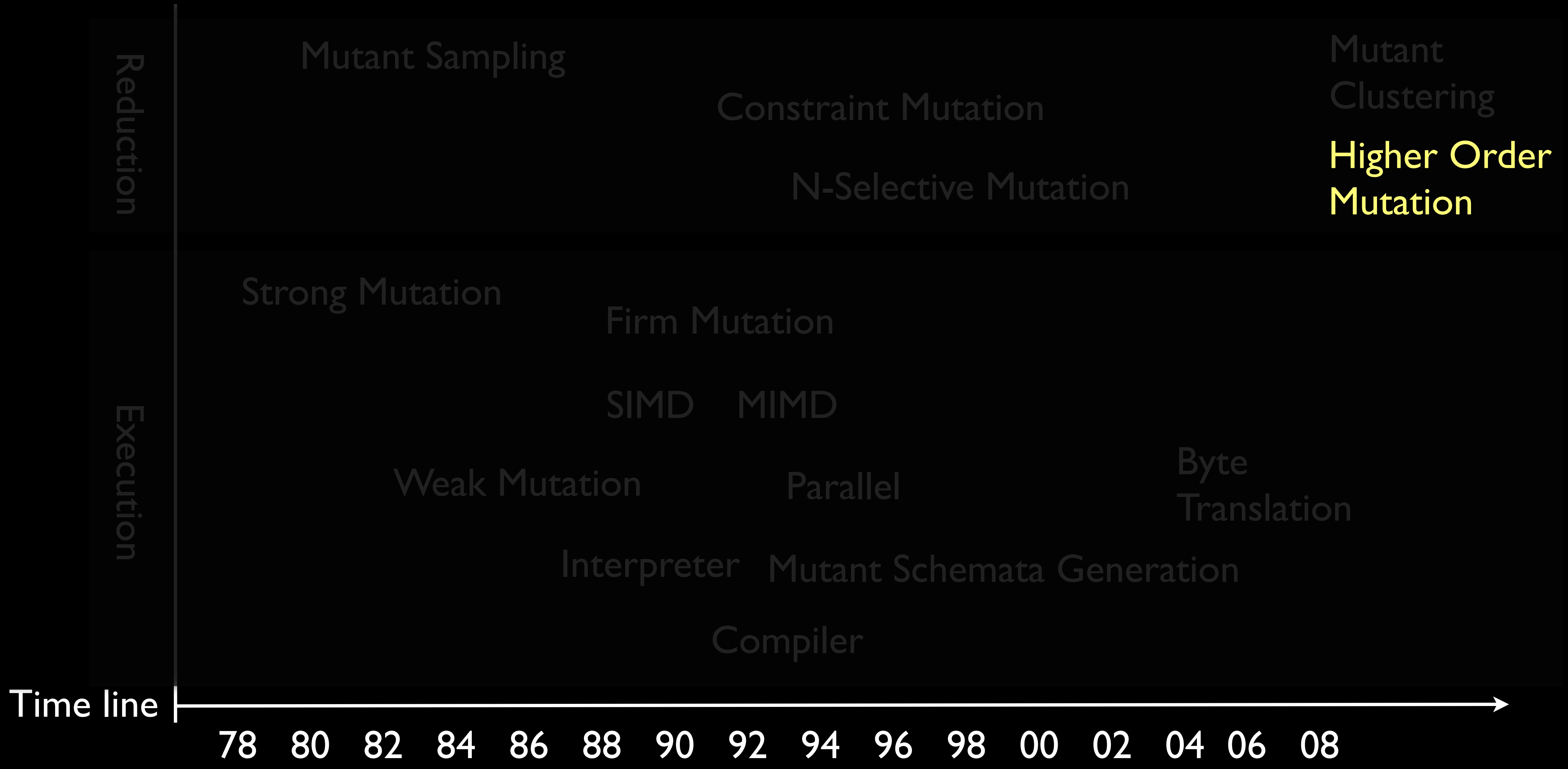
Mutant Schemata Generation

Compiler

Time line

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

Mutant Reduction Techniques



Reduction

Mutant Sampling

Constraint Mutation

N-Selective Mutation

Mutant Clustering

Higher Order Mutation

Execution

Strong Mutation

Firm Mutation

SIMD MIMD

Weak Mutation

Parallel

Byte Translation

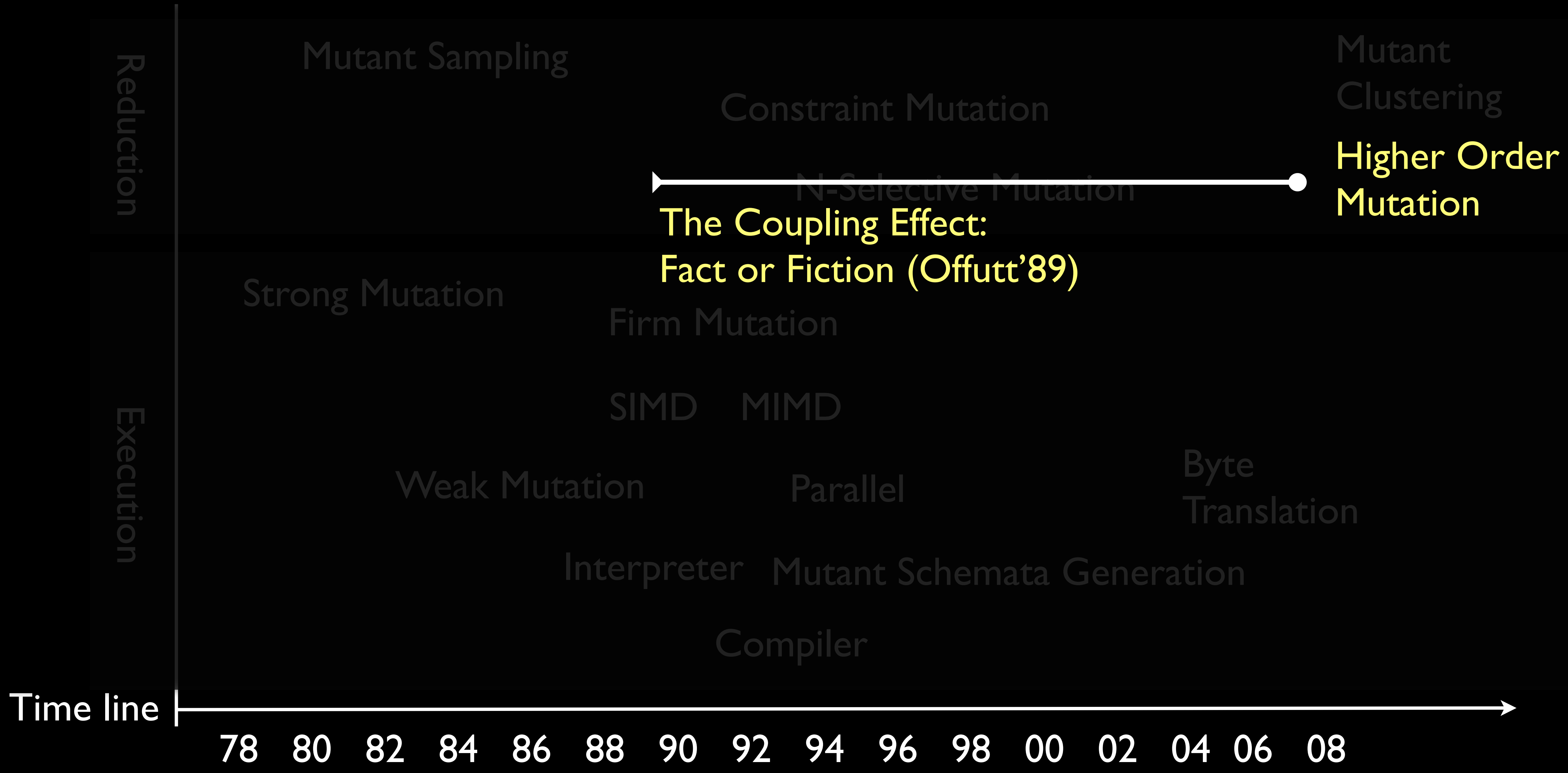
Interpreter Mutant Schemata Generation

Compiler

Time line

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

Mutant Reduction Techniques



Reduction

Mutant Sampling

Constraint Mutation

N-Selective Mutation

Mutant Clustering

Higher Order Mutation

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

Execution

Strong Mutation

Firm Mutation

SIMD MIMD

Weak Mutation

Parallel

Byte Translation

Interpreter

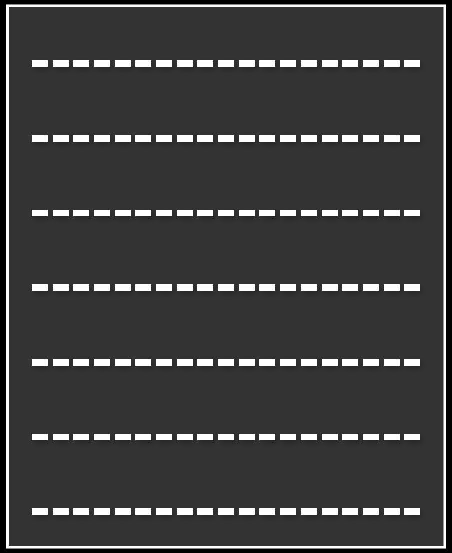
Mutant Schemata Generation

Compiler

Time line

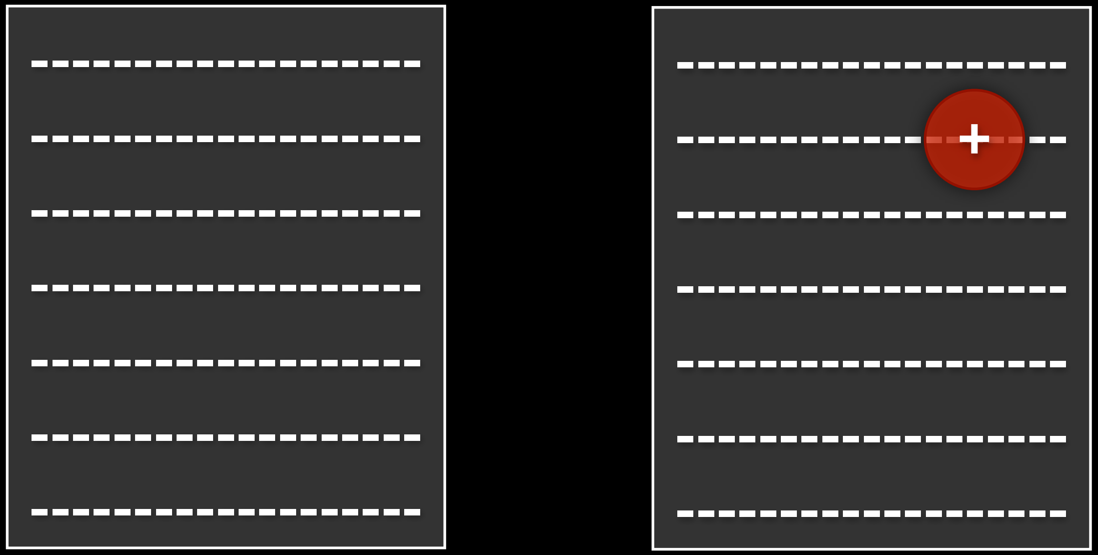
78 80 82 84 86 88 90 92 94 96 98 00 02 04 06 08

FOMs and HOMs



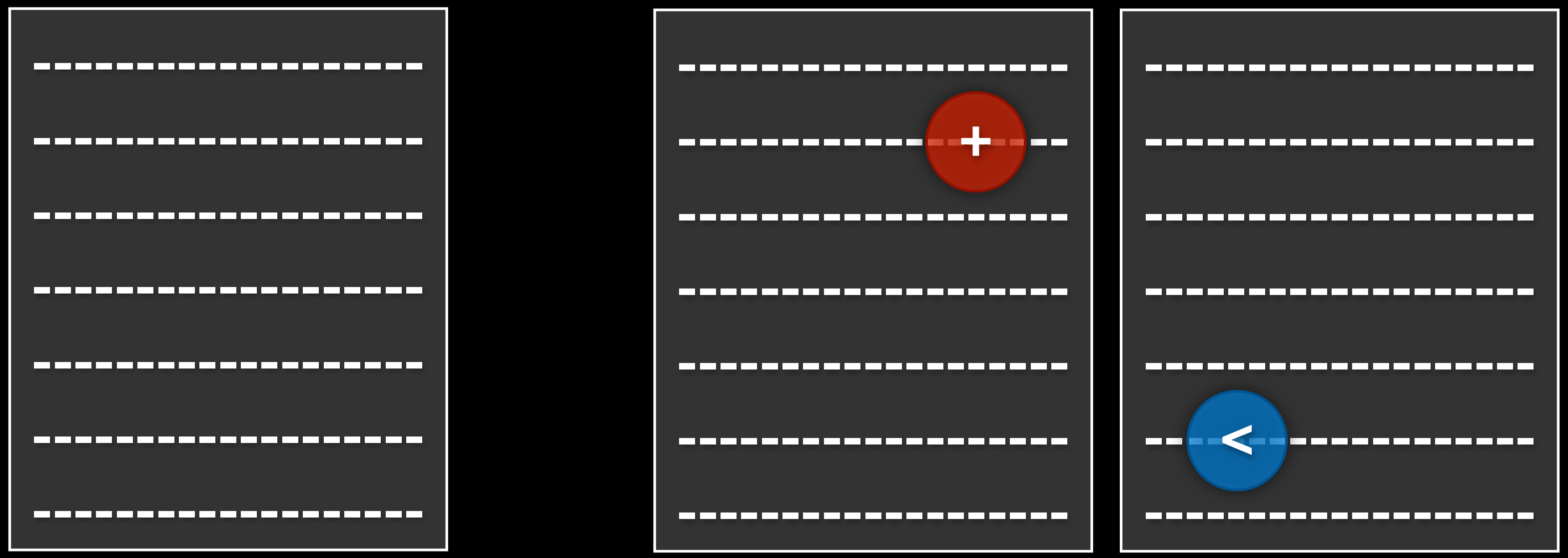
Original Program

FOMs and HOMs



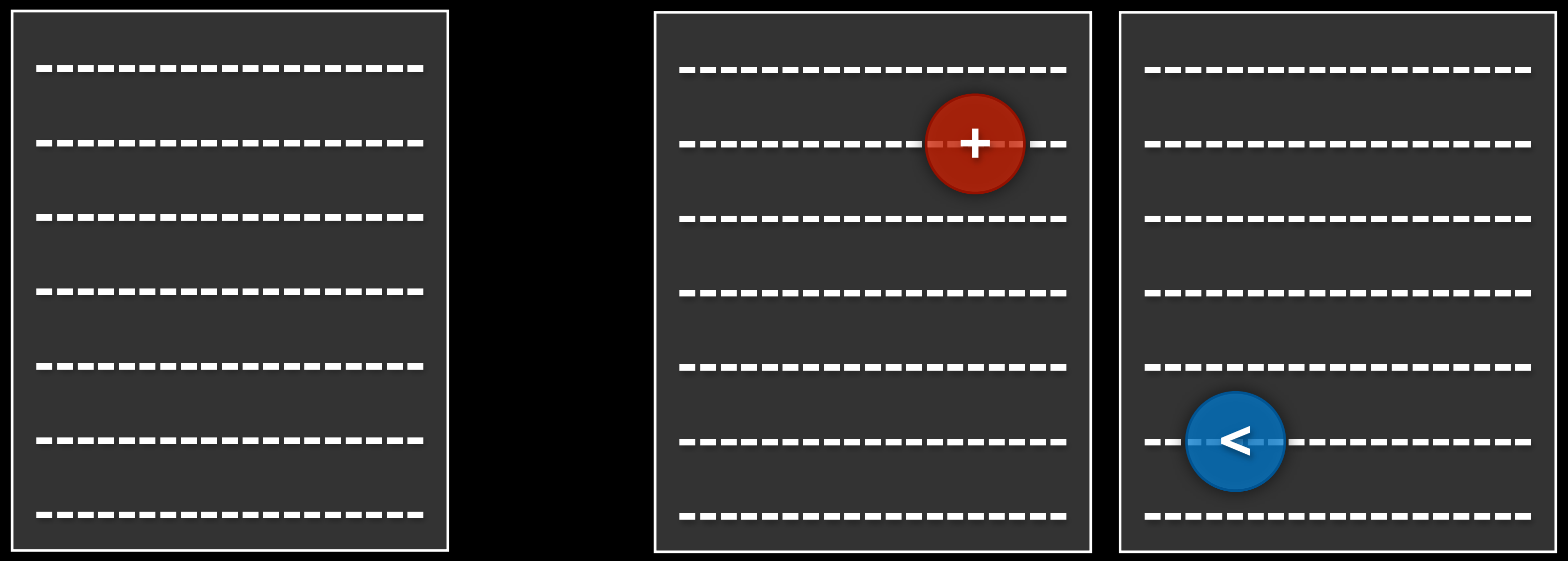
Original Program

FOMs and HOMs



Original Program

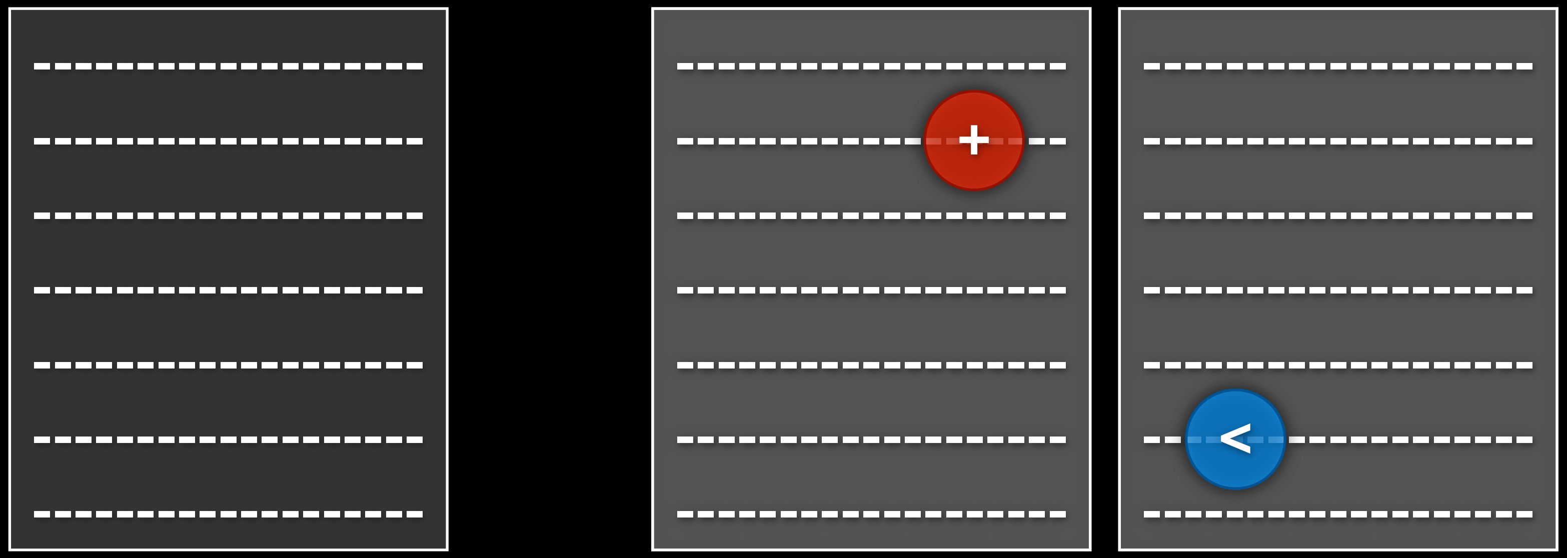
FOMs and HOMs



Original Program

First Order Mutants

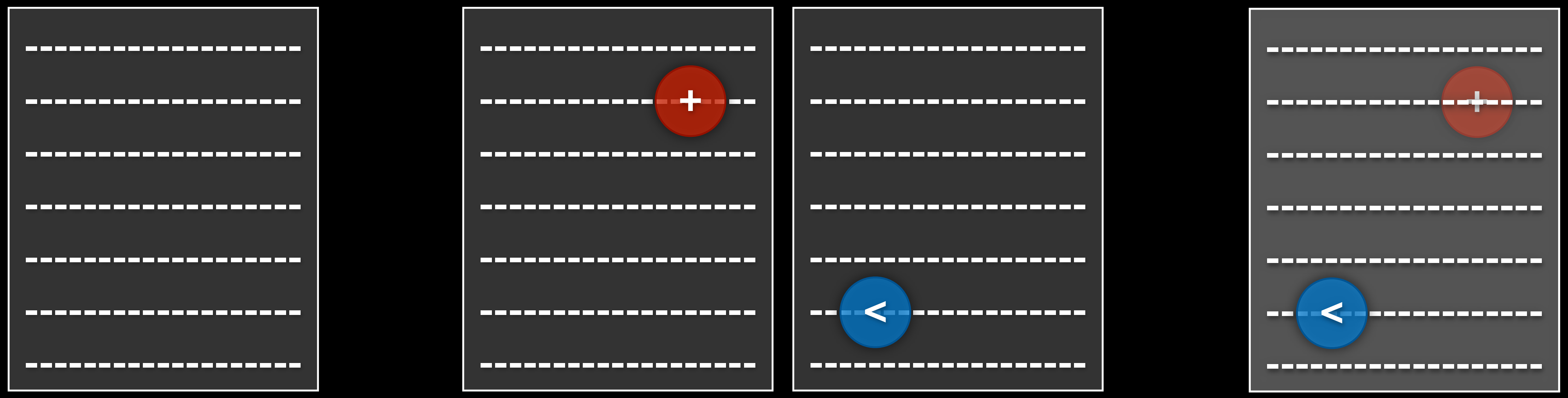
FOMs and HOMs



Original Program

First Order Mutants

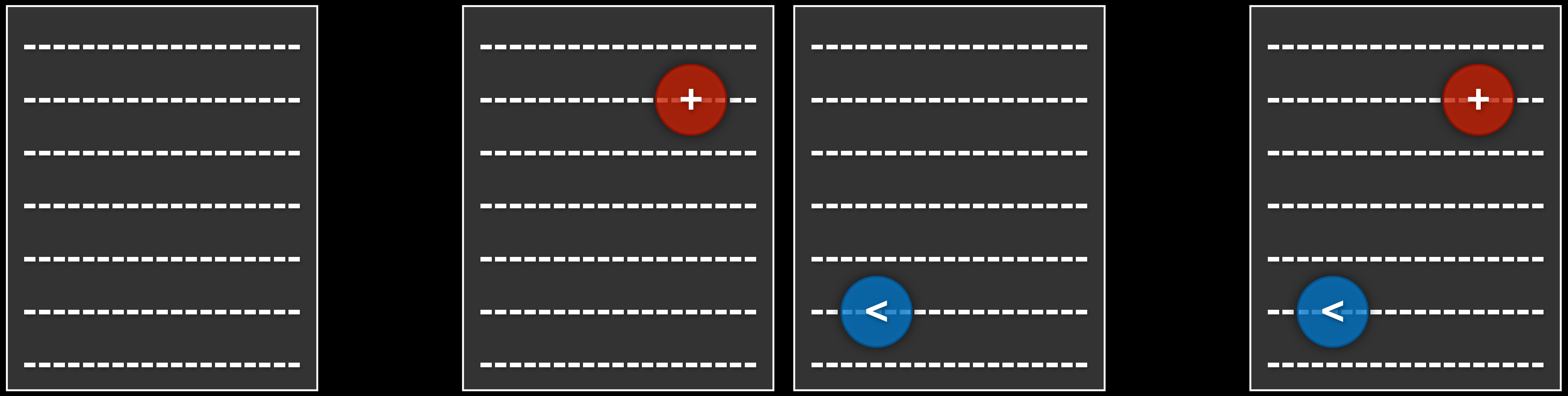
FOMs and HOMs



Original Program

First Order Mutants

FOMs and HOMs



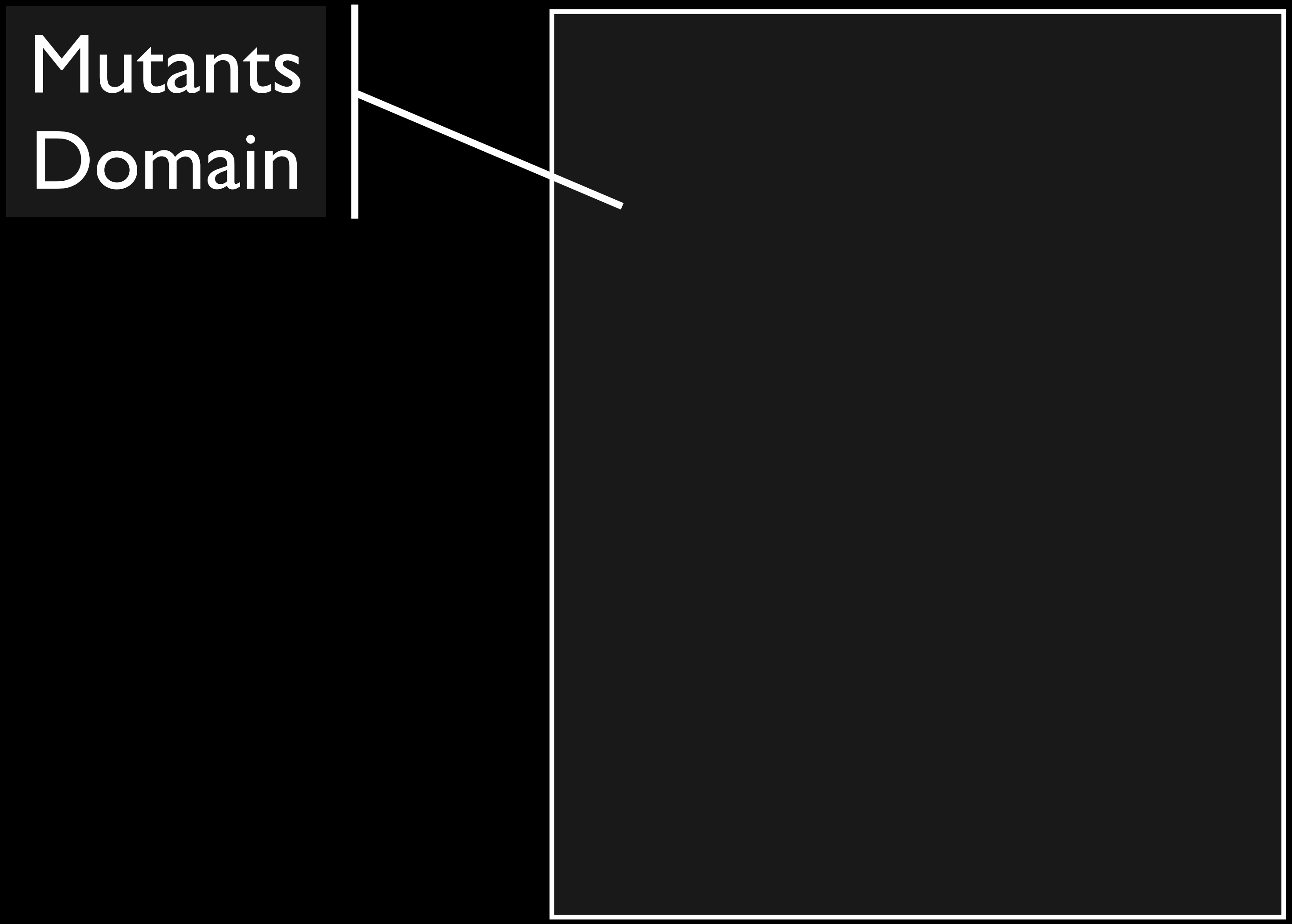
Original Program

First Order Mutants

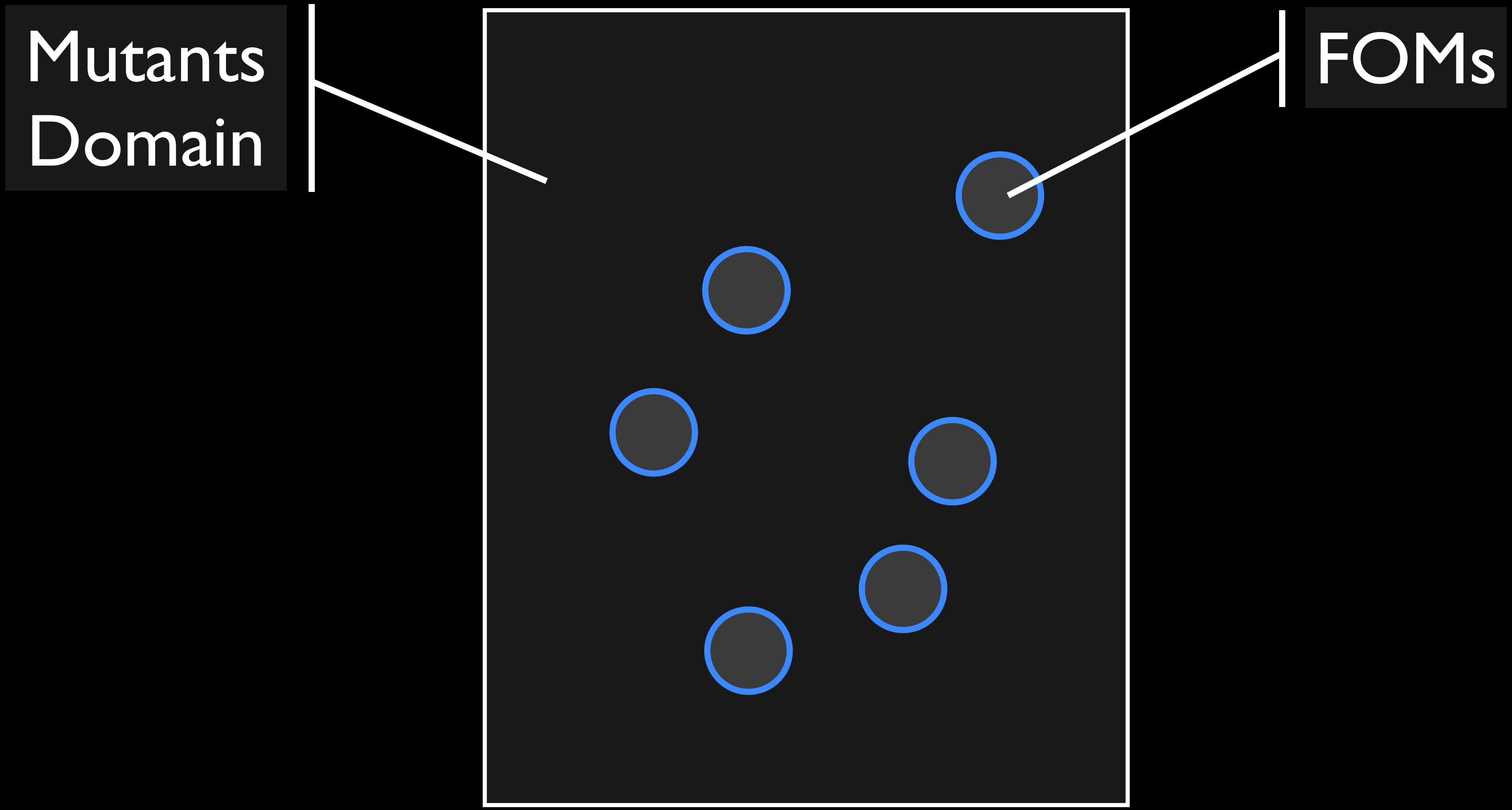
Higher Order Mutants

FOMs and HOMs

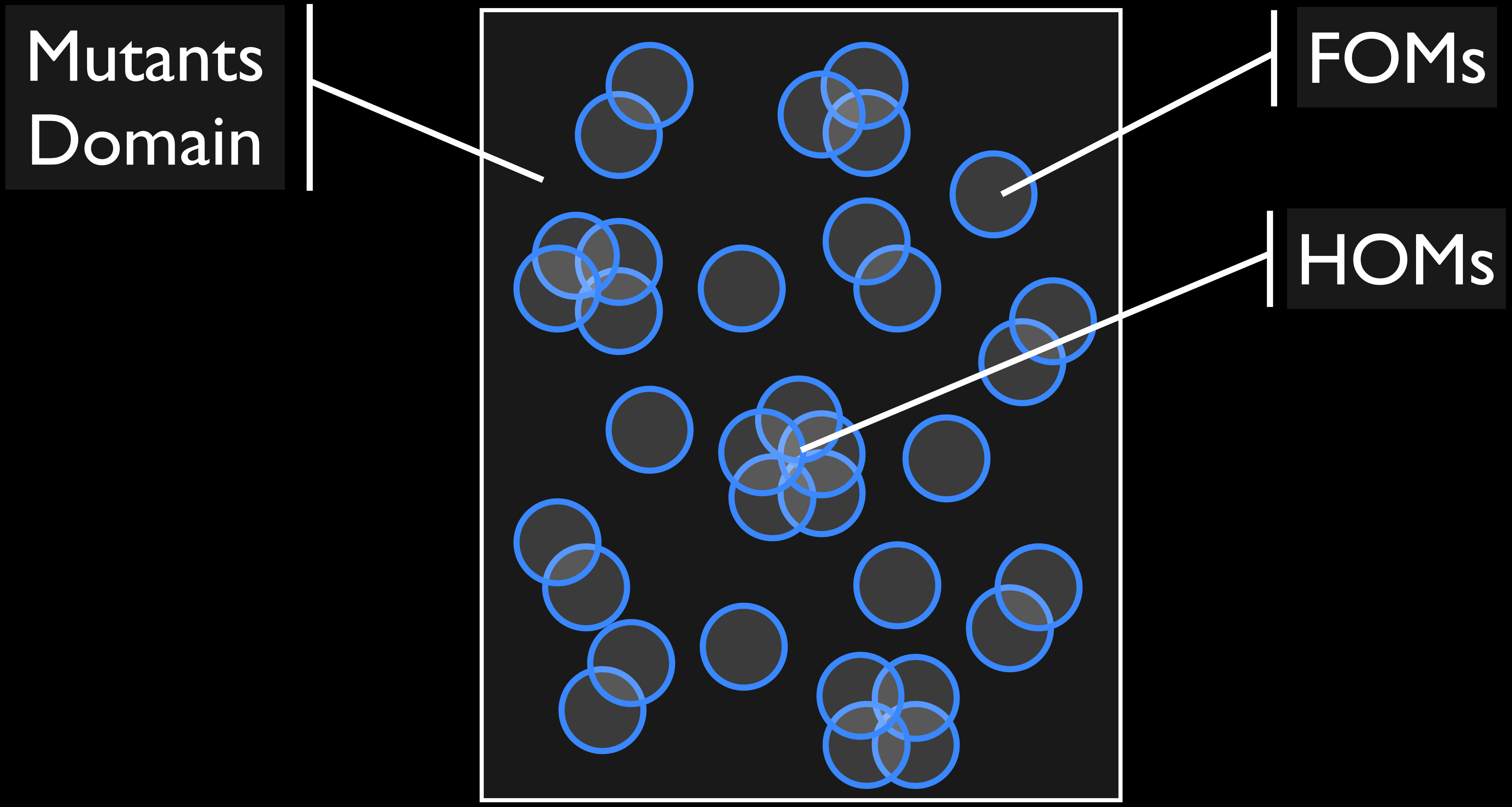
FOMs and HOMs



FOMs and HOMs



FOMs and HOMs



Traditional FOM Testing

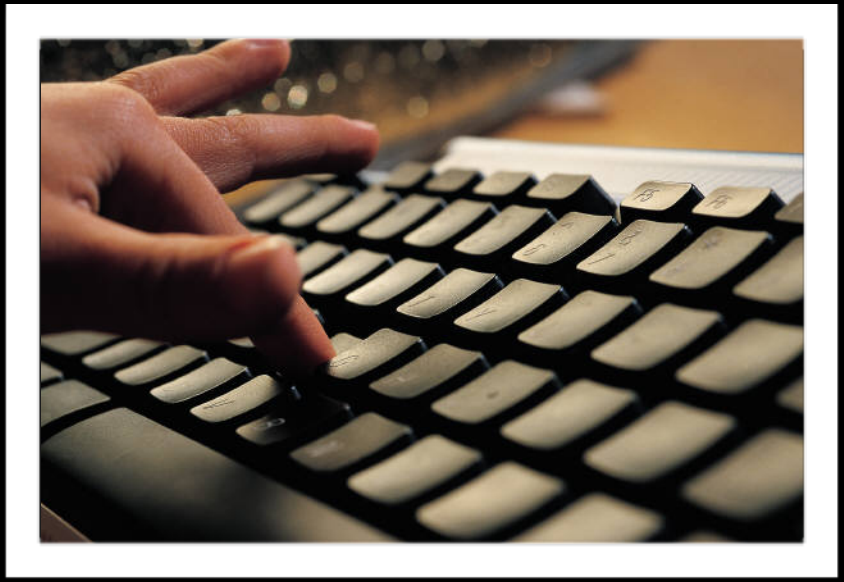
Traditional FOM Testing

Higher Order Mutants are far too numerous

Traditional FOM Testing

Higher Order Mutants are far too numerous

Competent Programmer

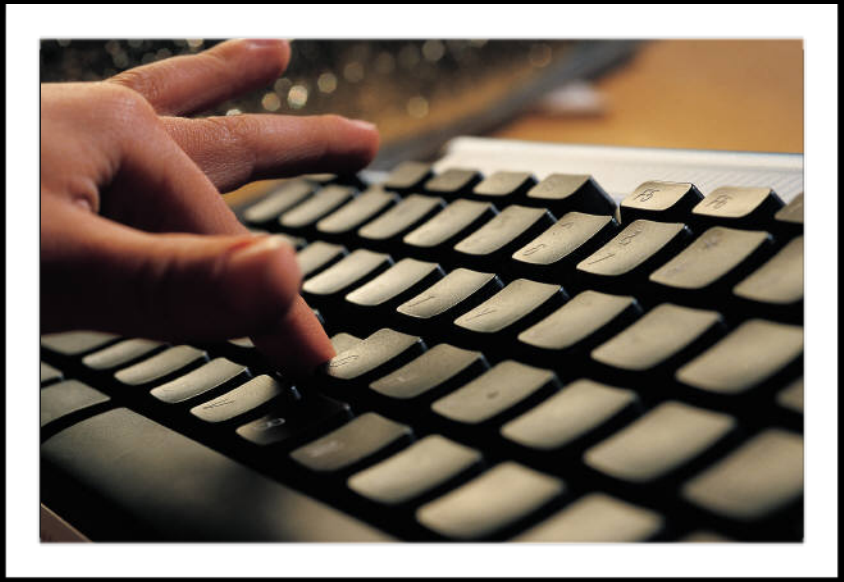


Traditional FOM Testing

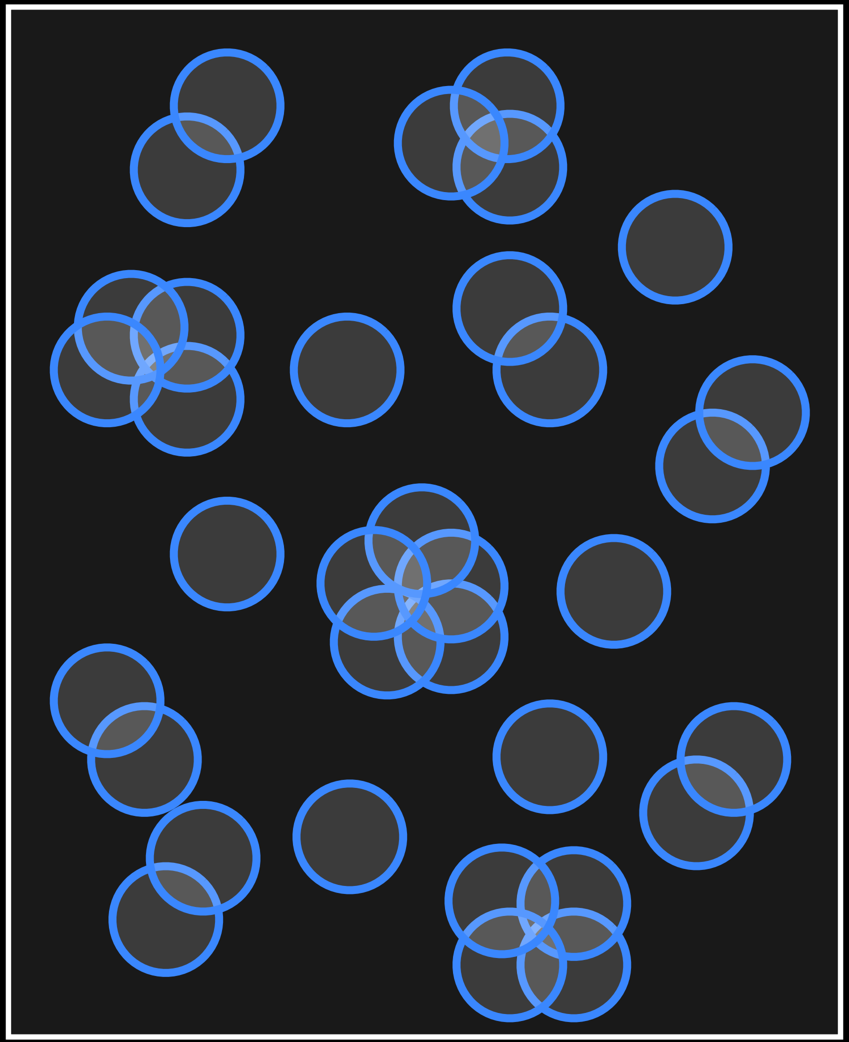
Higher Order Mutants are far too numerous

Competent Programmer

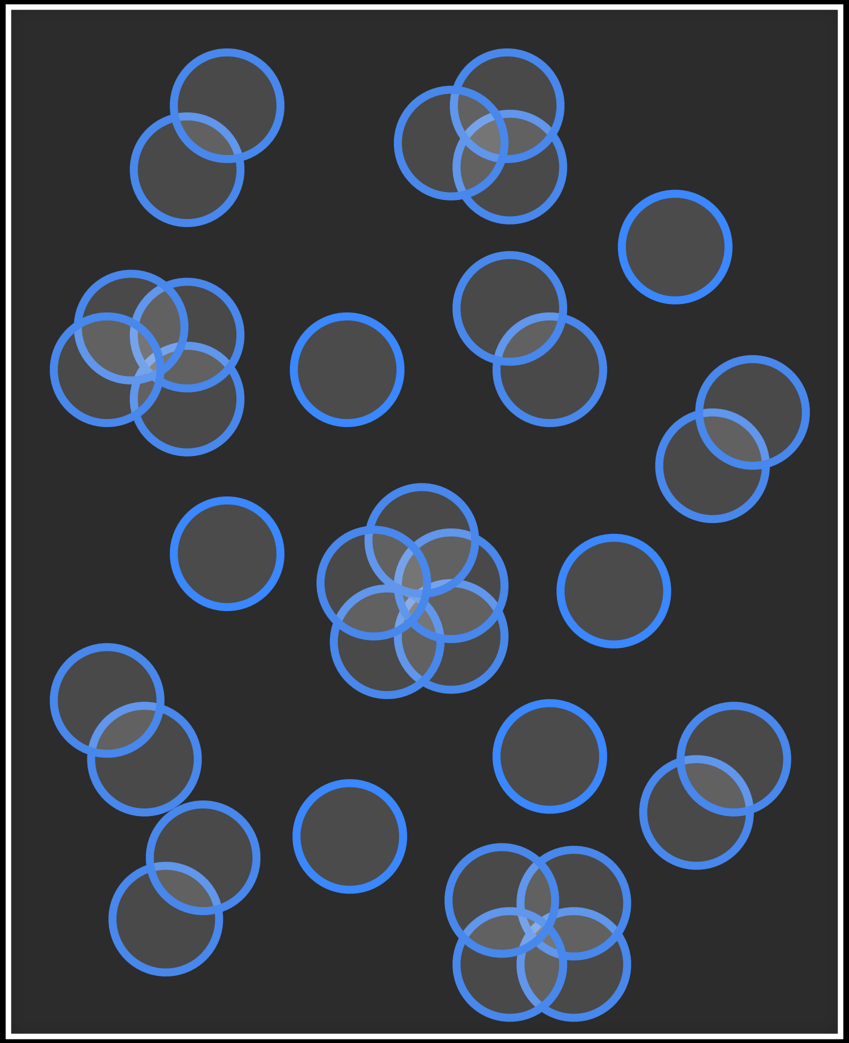
Mutation Coupling Effect



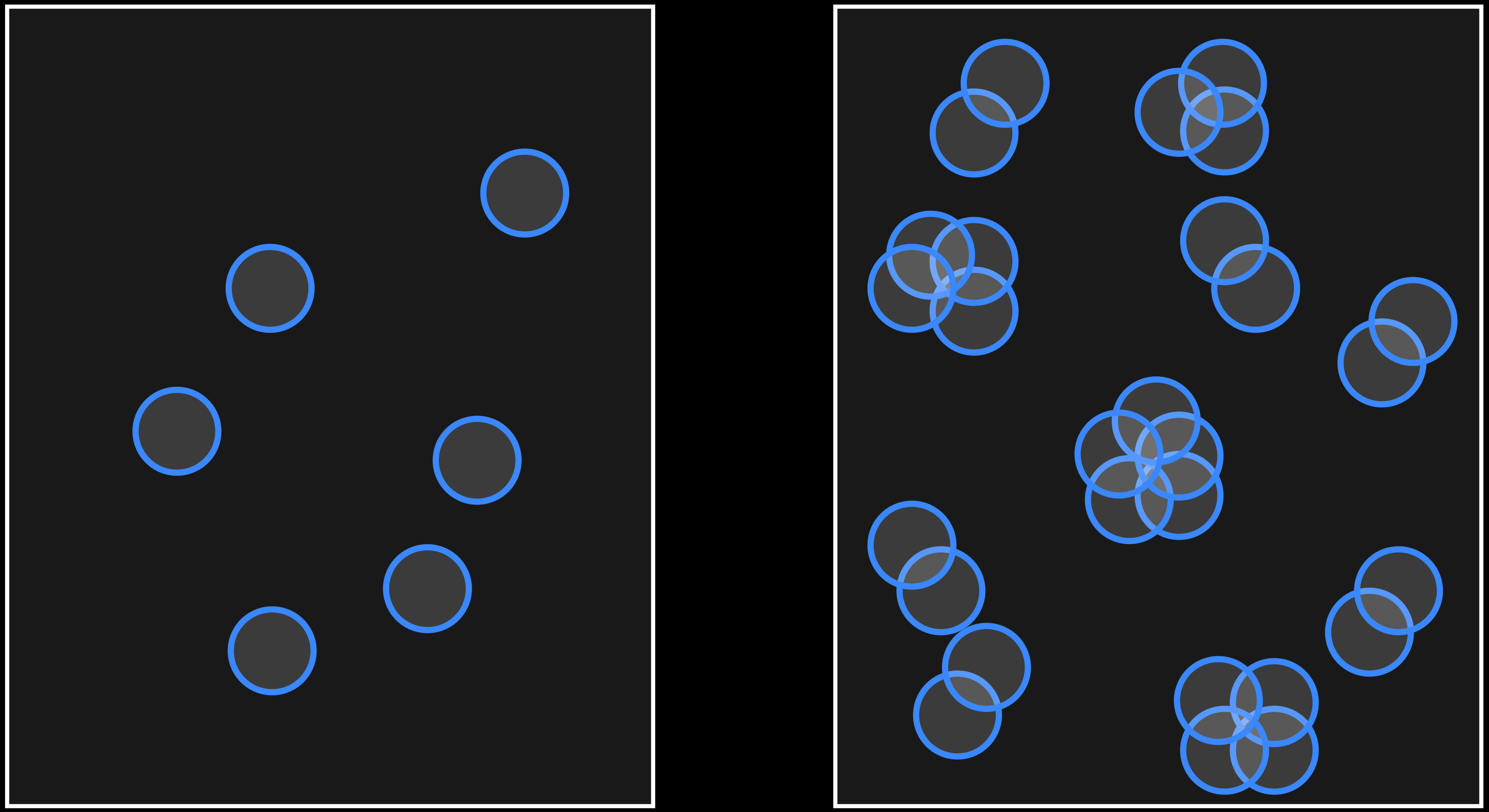
Mutation Coupling Effect



Mutation Coupling Effect

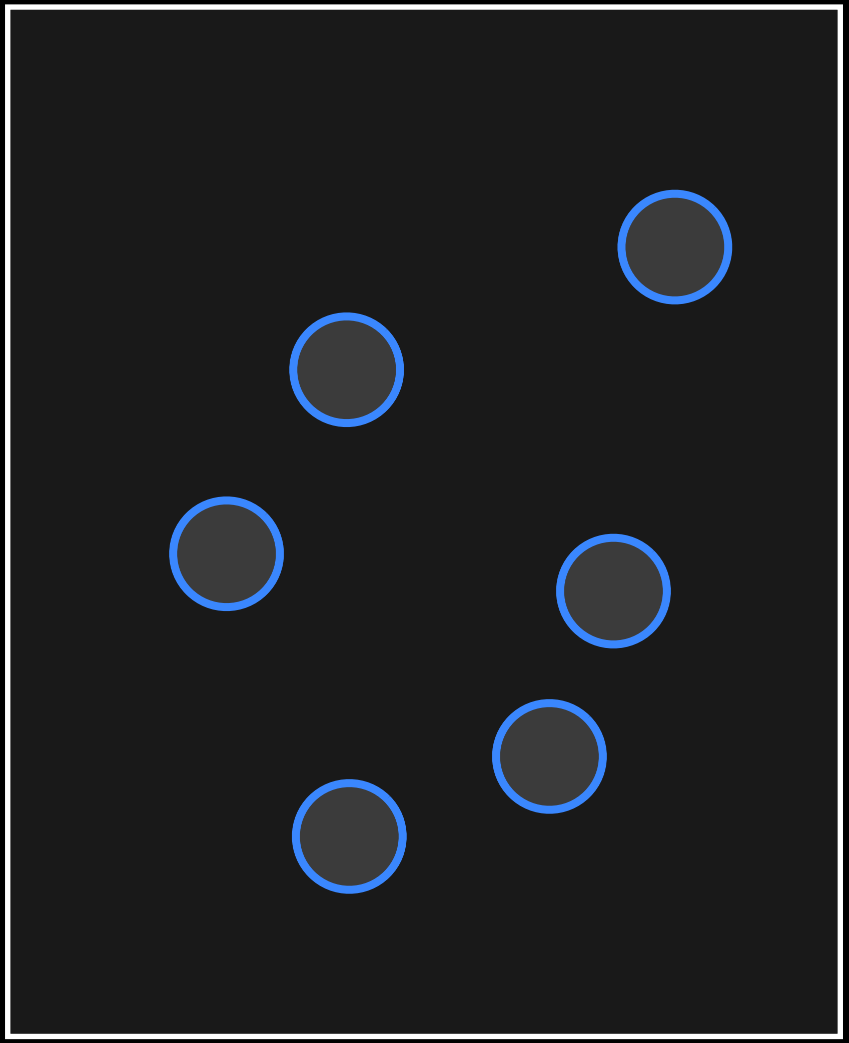


Mutation Coupling Effect

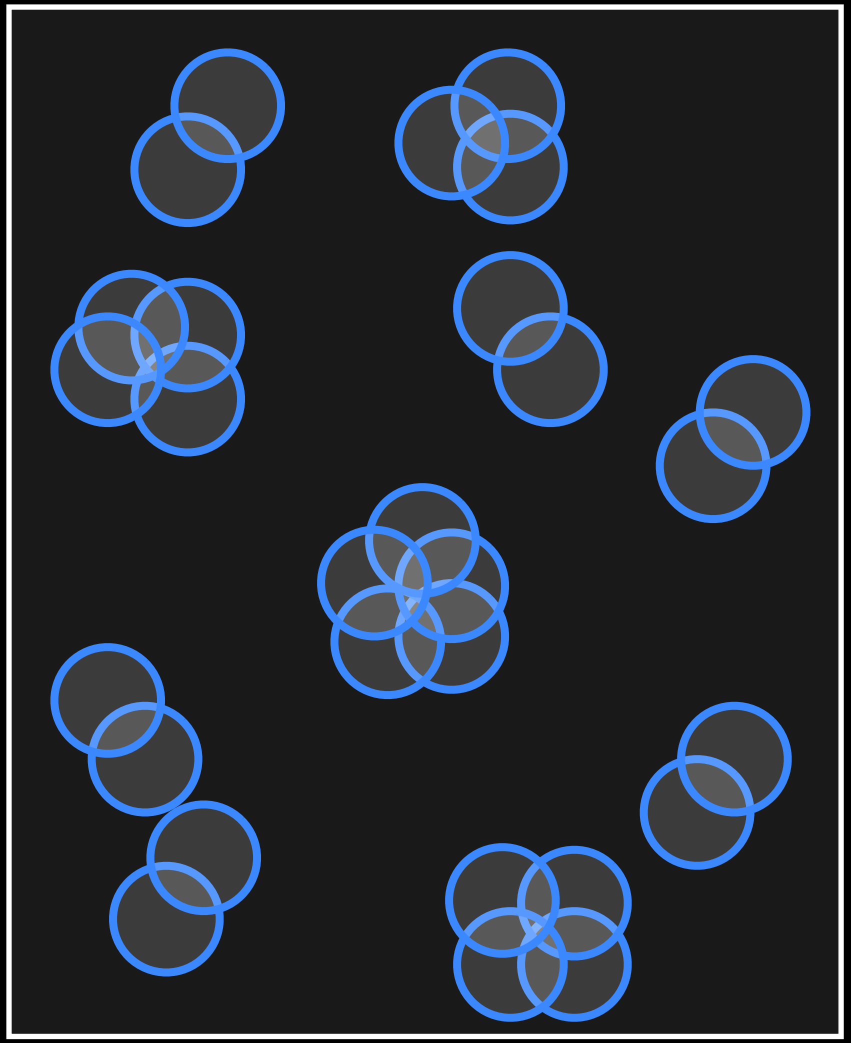


Mutation Coupling Effect

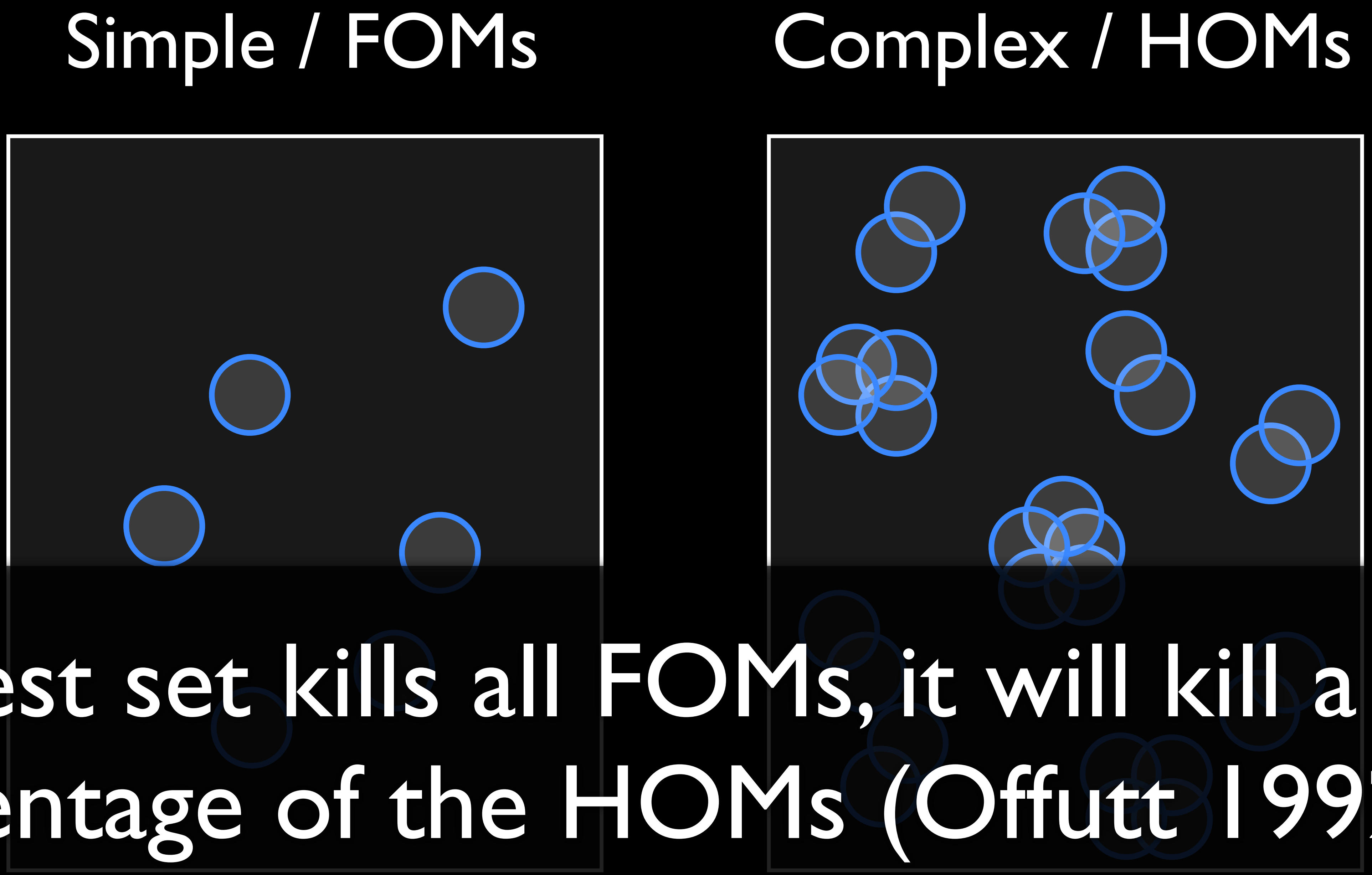
Simple / FOMs



Complex / HOMs



Mutation Coupling Effect

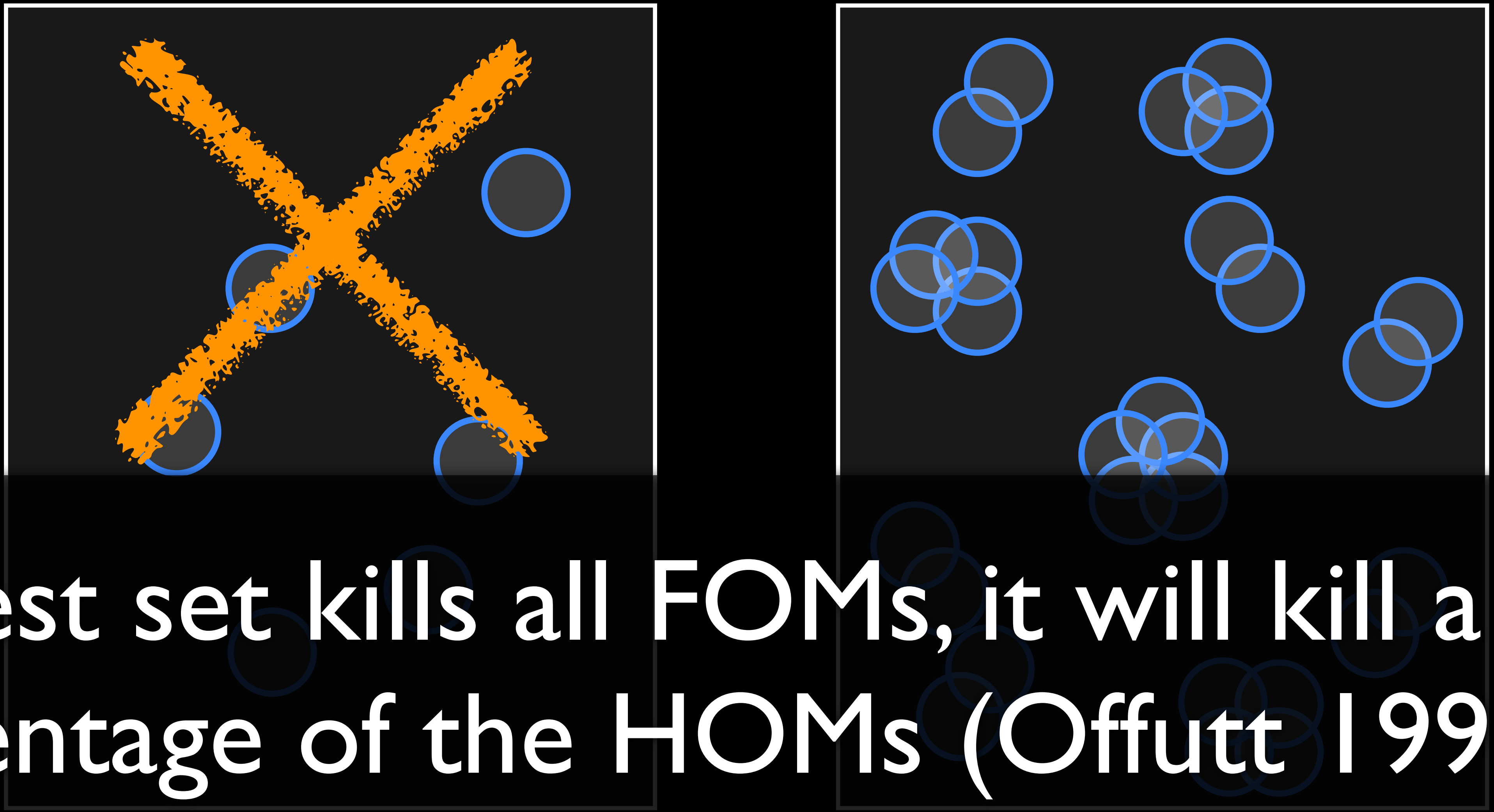


If a test set kills all FOMs, it will kill a large percentage of the HOMs (Offutt 1992)

Mutation Coupling Effect

Simple / FOMs

Complex / HOMs

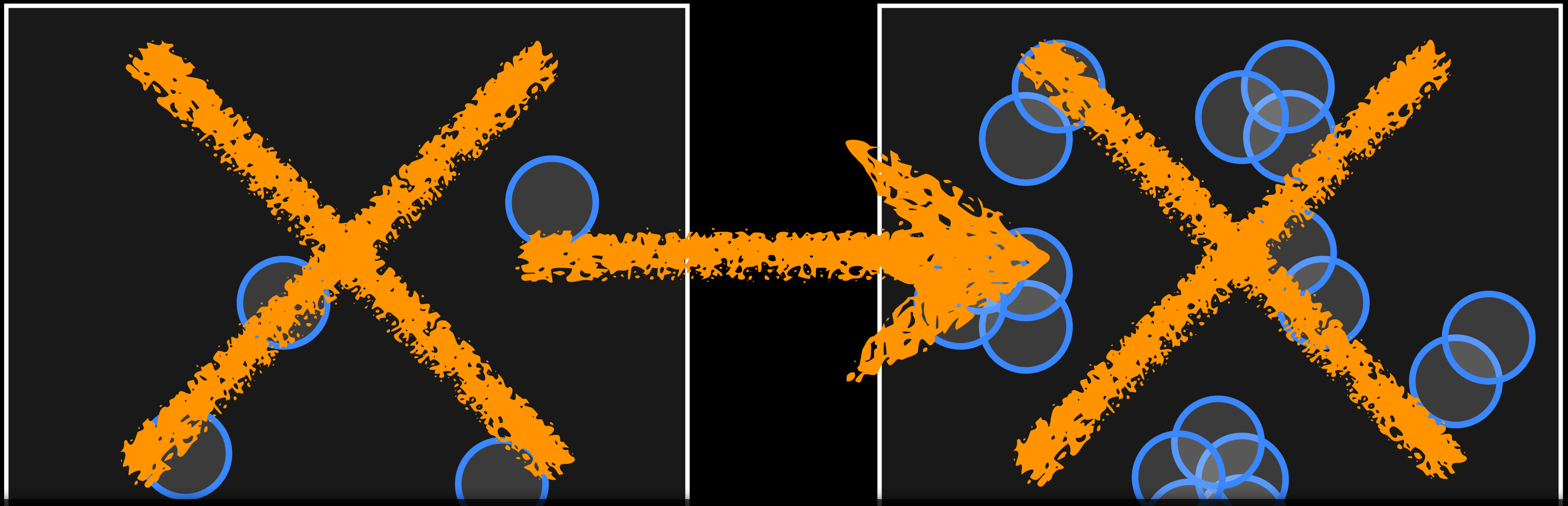


If a test set kills all FOMs, it will kill a large percentage of the HOMs (Offutt 1992)

Mutation Coupling Effect

Simple / FOMs

Complex / HOMs

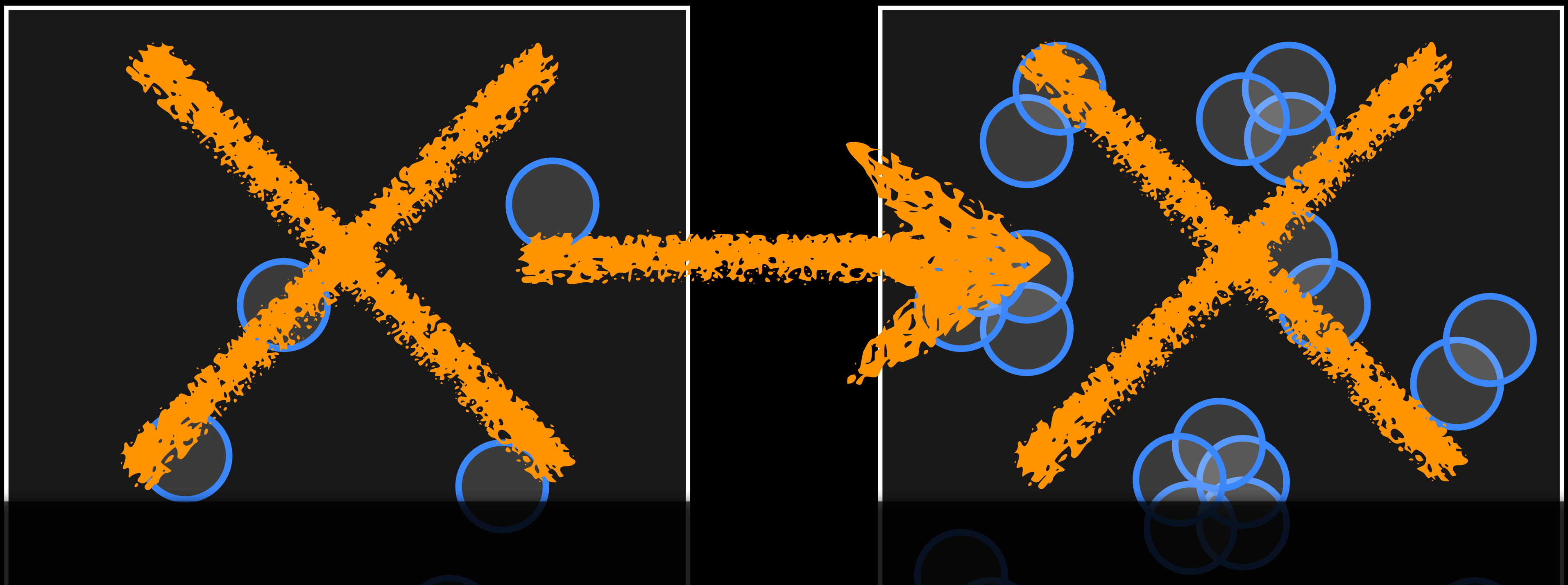


If a test set kills all FOMs, it will kill a large percentage of the HOMs (Offutt 1992)

Mutation Coupling Effect

Simple / FOMs

Complex / HOMs



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. + → -

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

Particle Swarm Optimization

Ant Colonies

Genetic Algorithms

Hill Climbing

Genetic Programming

Simulated Annealing

Random

Greedy

LP

Estimation of Distribution Algorithms

Search based MT

Why not search for mutants that are hard to kill?

Tabu Search

Ant Colonies

Particle Swarm Optimization

Genetic Algorithms

Hill Climbing

Genetic Programming

Simulated Annealing

Random

Greedy

LP

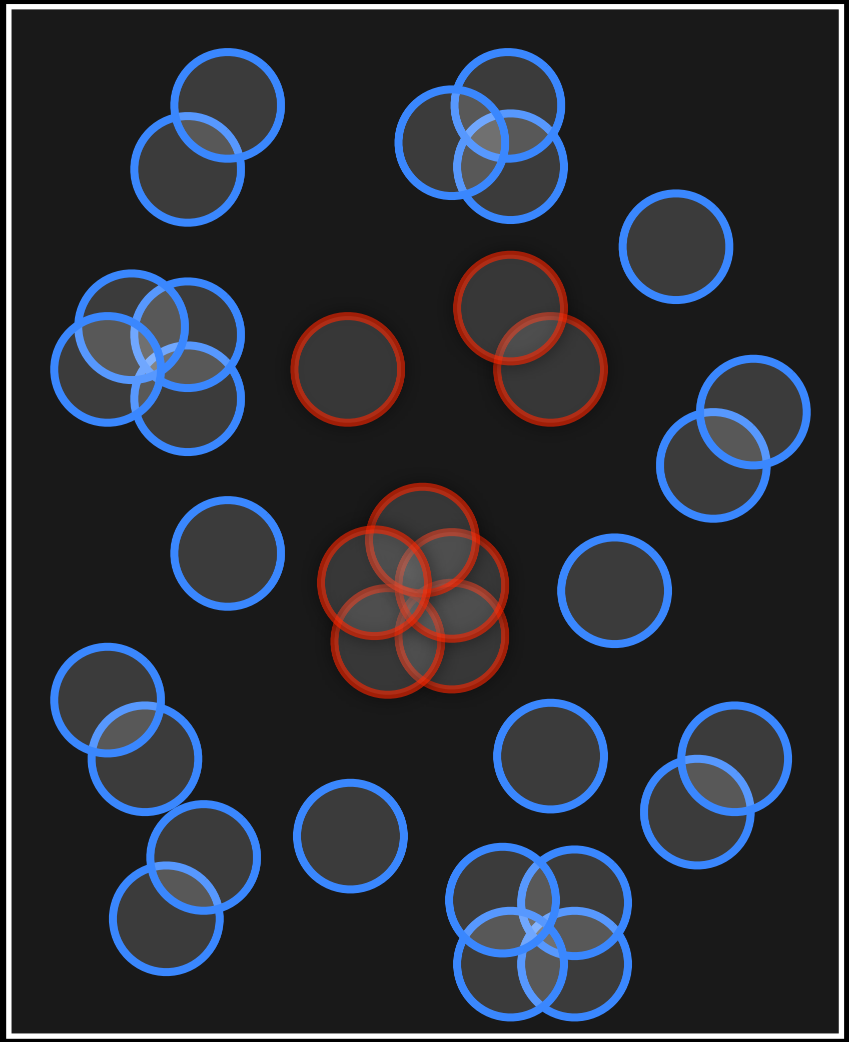
Estimation of Distribution Algorithms

Aim

Searching

avoids

enumerating all mutants



Approaches

Single Objective

- Genetic Algorithm
- Hill Climbing
- Greedy

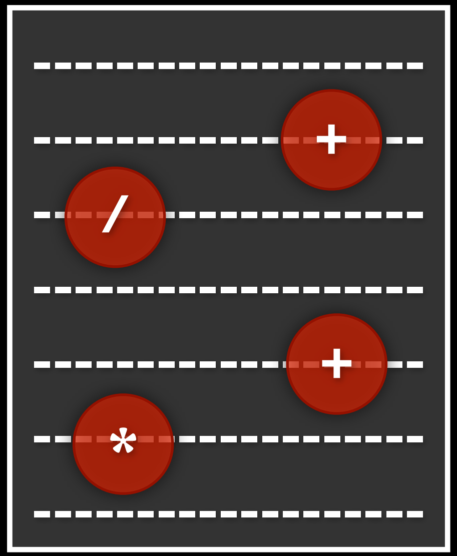
Multi Objective

- Genetic Programming

Single Objective

Data Representation

- Index
- Position

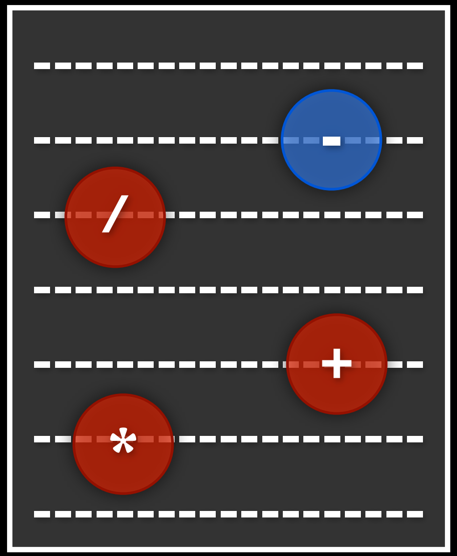


Single Objective

Data Representation

- Index
- Position

1	0	0	0



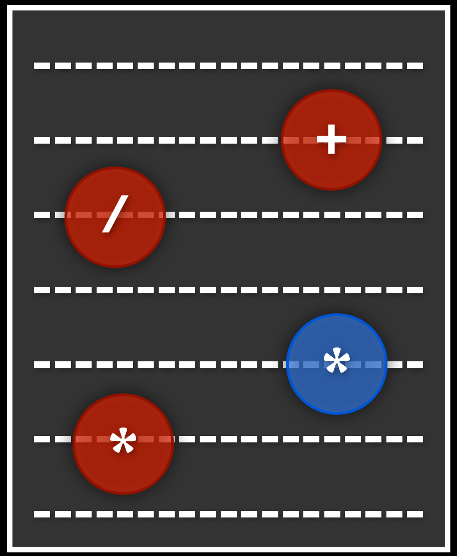
Single Objective

Data Representation

Index

Position

1	0	0	0
0	0	2	0

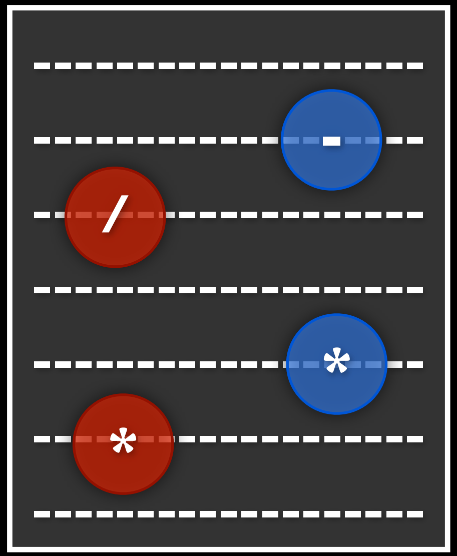


Single Objective

Data Representation

- Index
- Position

1	0	0	0
0	0	2	0



Single Objective

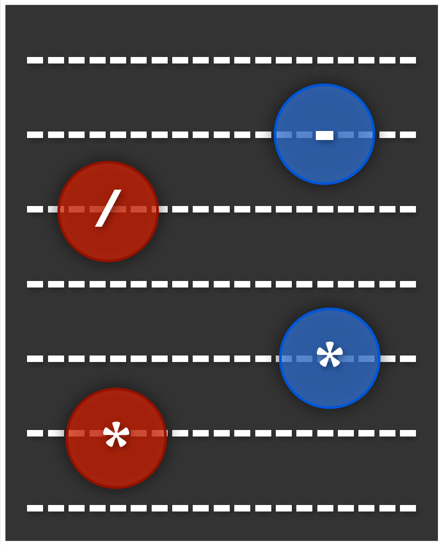
Data Representation

Index

Position

1	0	0	0
0	0	2	0

1	0	2	0



Single Objective

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

fitness >= 1, easier to be killed

fitness < 1, harder to be killed

fitness = 0, cannot to be killed

Algorithms

GA

Crossover + Mutation

Greedy

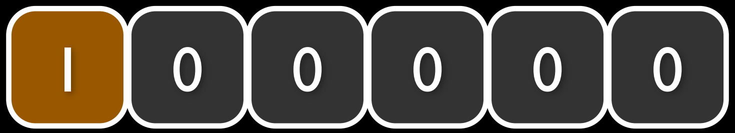
HC

Algorithms

GA

Crossover + Mutation

Greedy



1 0 0 0 0 0

HC

Algorithms

GA

Crossover + Mutation

Greedy

1	0	0	0	0	0

HC

Algorithms

GA

Crossover + Mutation

Greedy

1	0	0	0	2	0

HC

Algorithms

GA

Crossover + Mutation

Greedy

1	0	0	0	2	0

HC

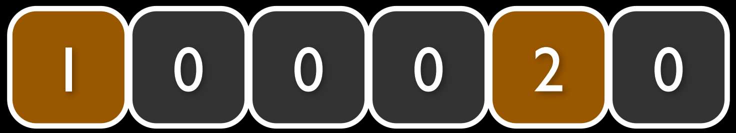
1	0	0	3	0	1

Algorithms

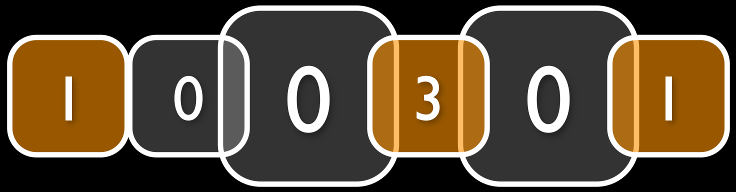
GA

Crossover + Mutation

Greedy



HC



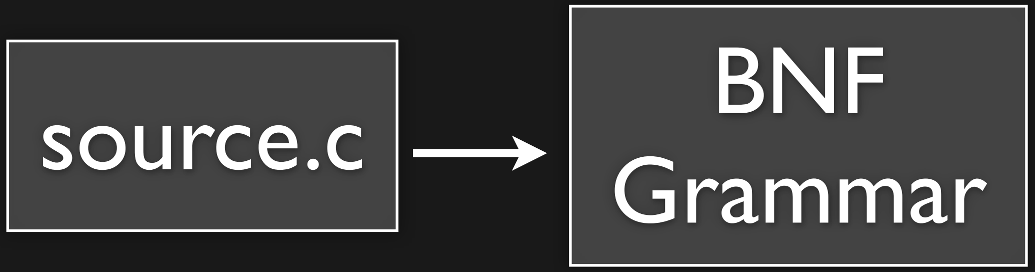
Multi Objective

Multi Objective

source.c

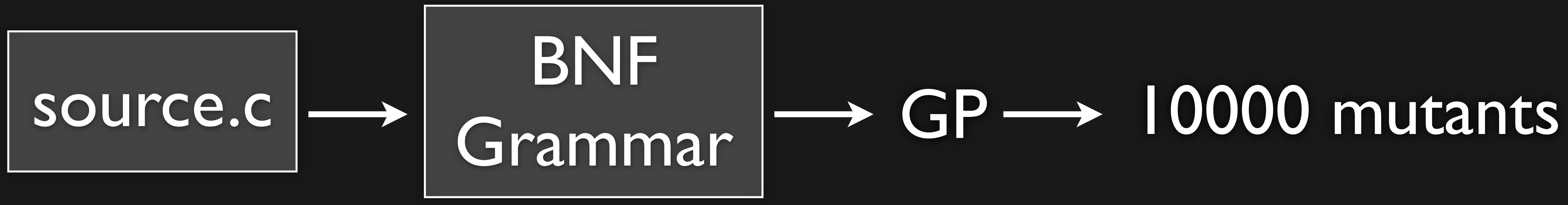
Pareto Evolution

Multi Objective



Pareto Evolution

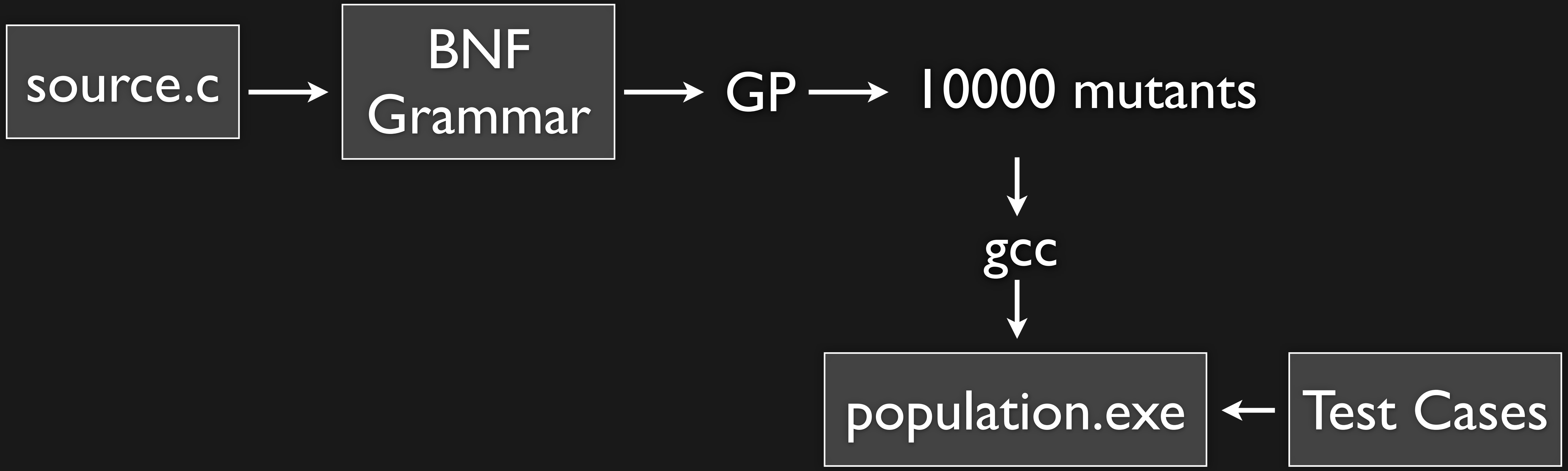
Multi Objective



source.c → BNF Grammar → GP → 10000 mutants

Pareto Evolution

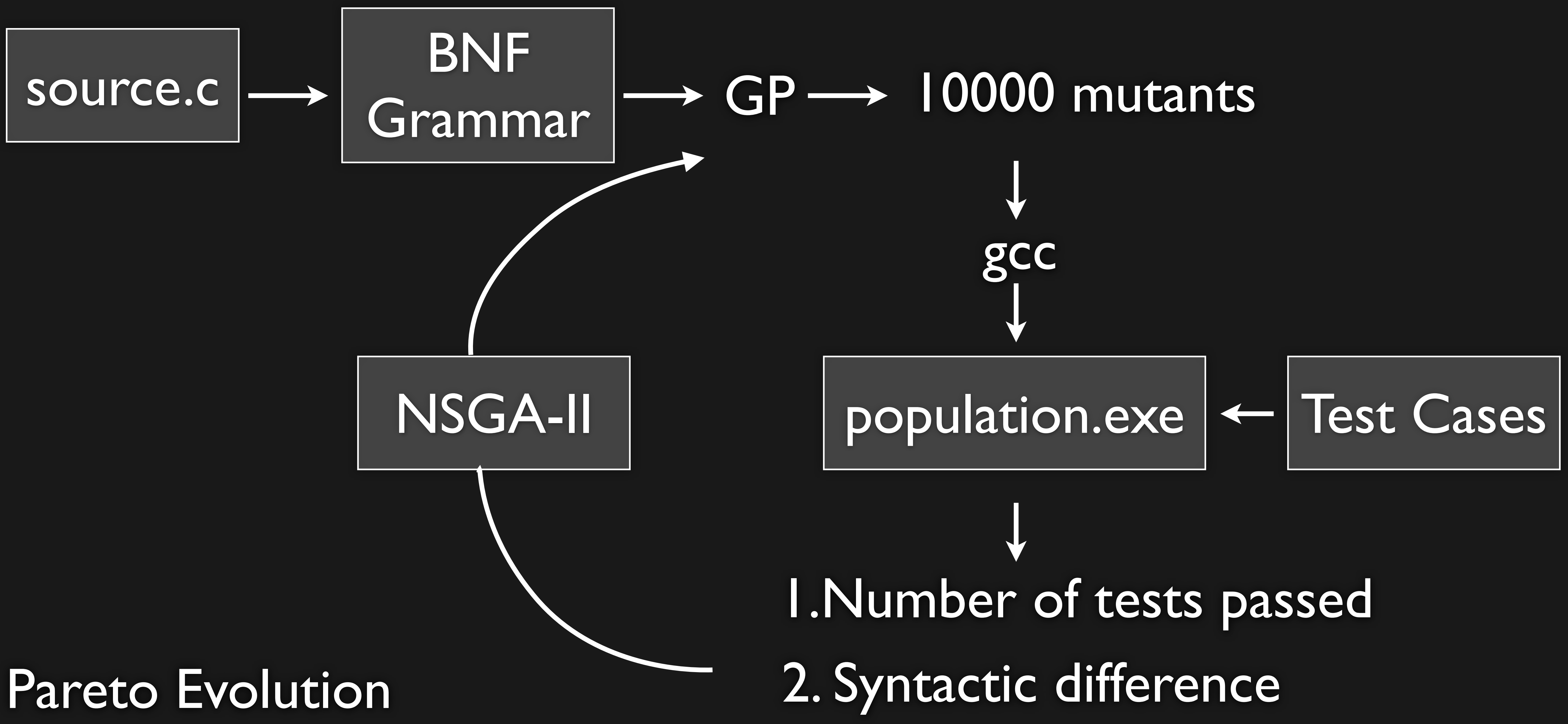
Multi Objective



source.c → BNF Grammar → GP → 10000 mutants → gcc → population.exe ← Test Cases

Pareto Evolution

Multi Objective

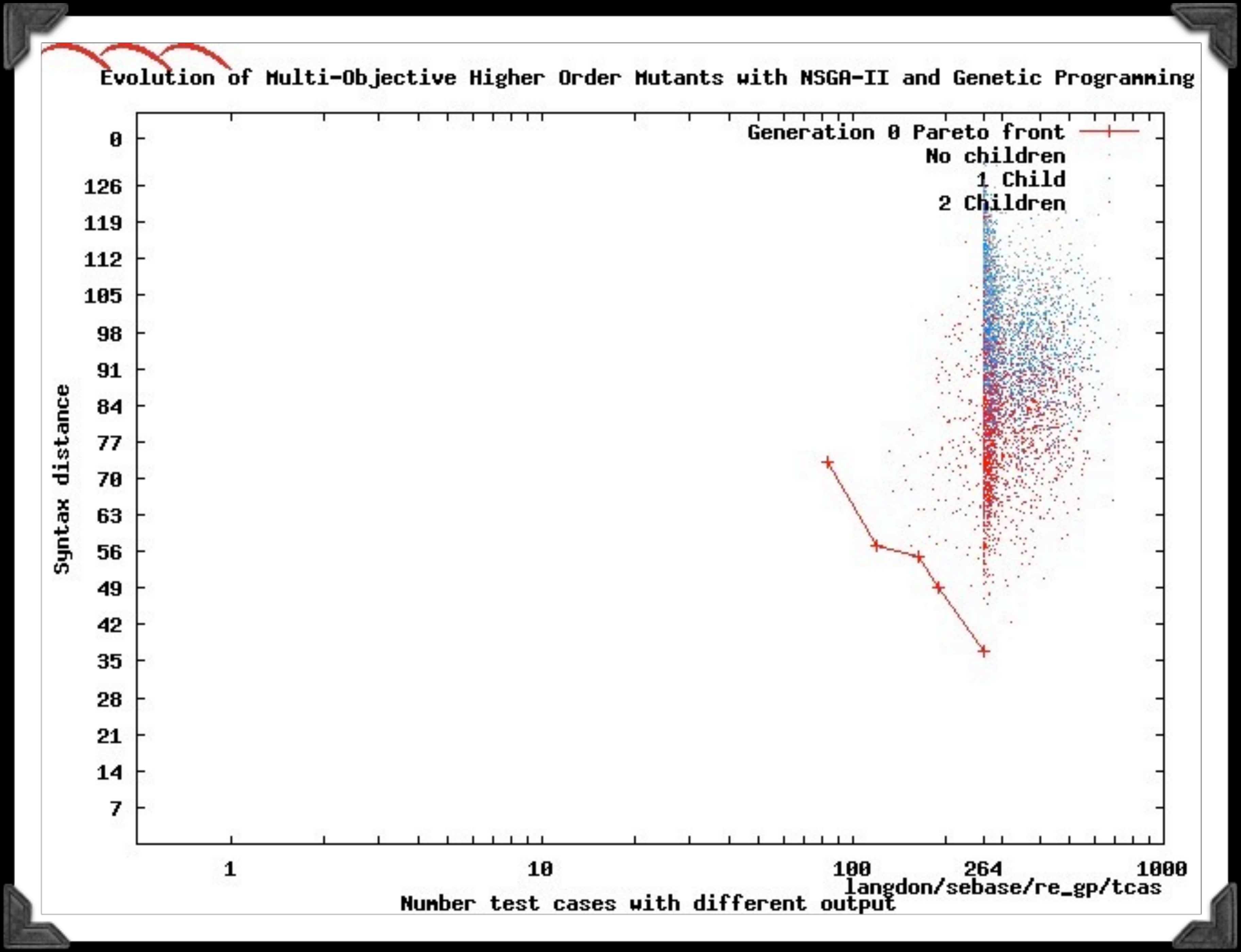


1.Number of tests passed

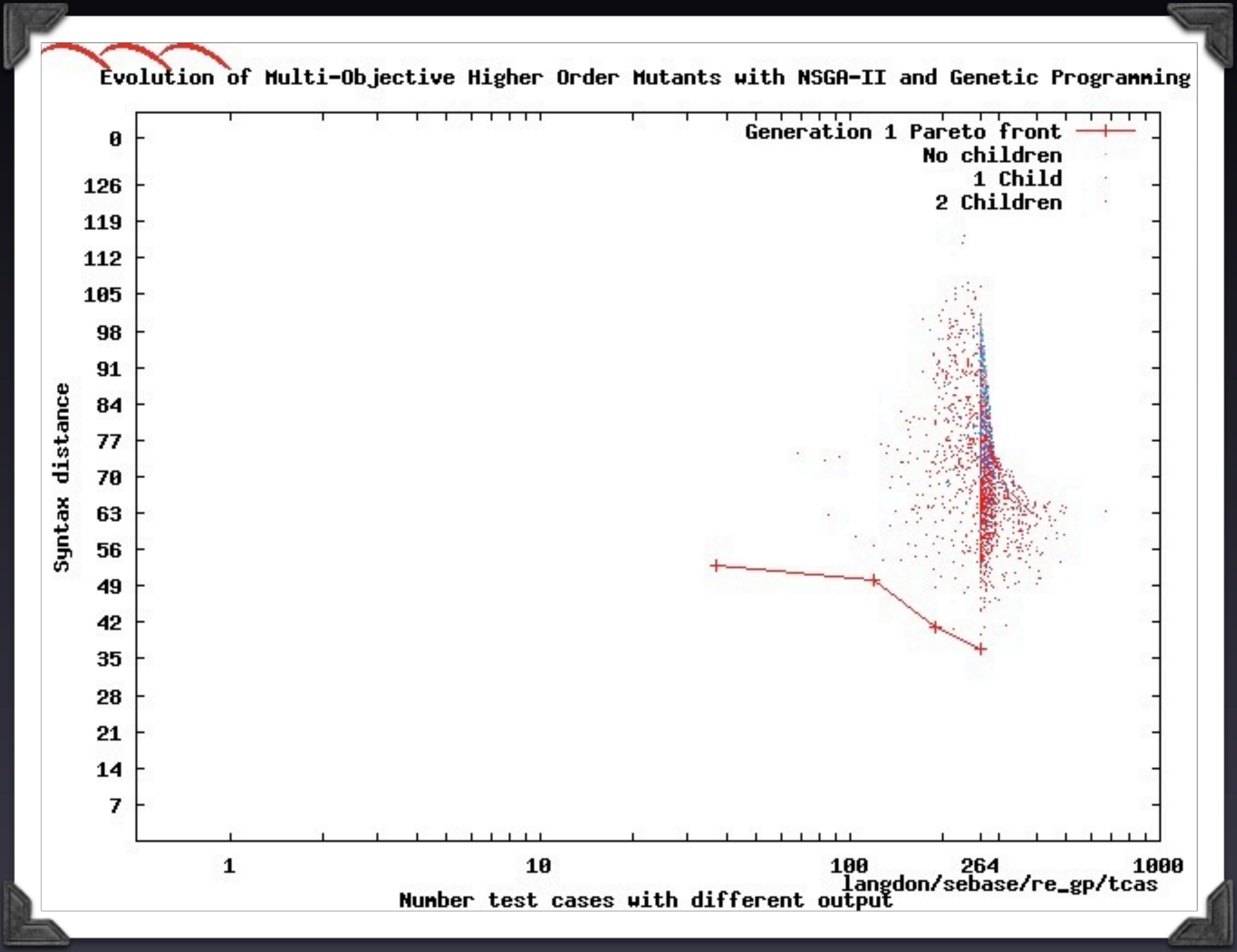
2. Syntactic difference

Pareto Evolution

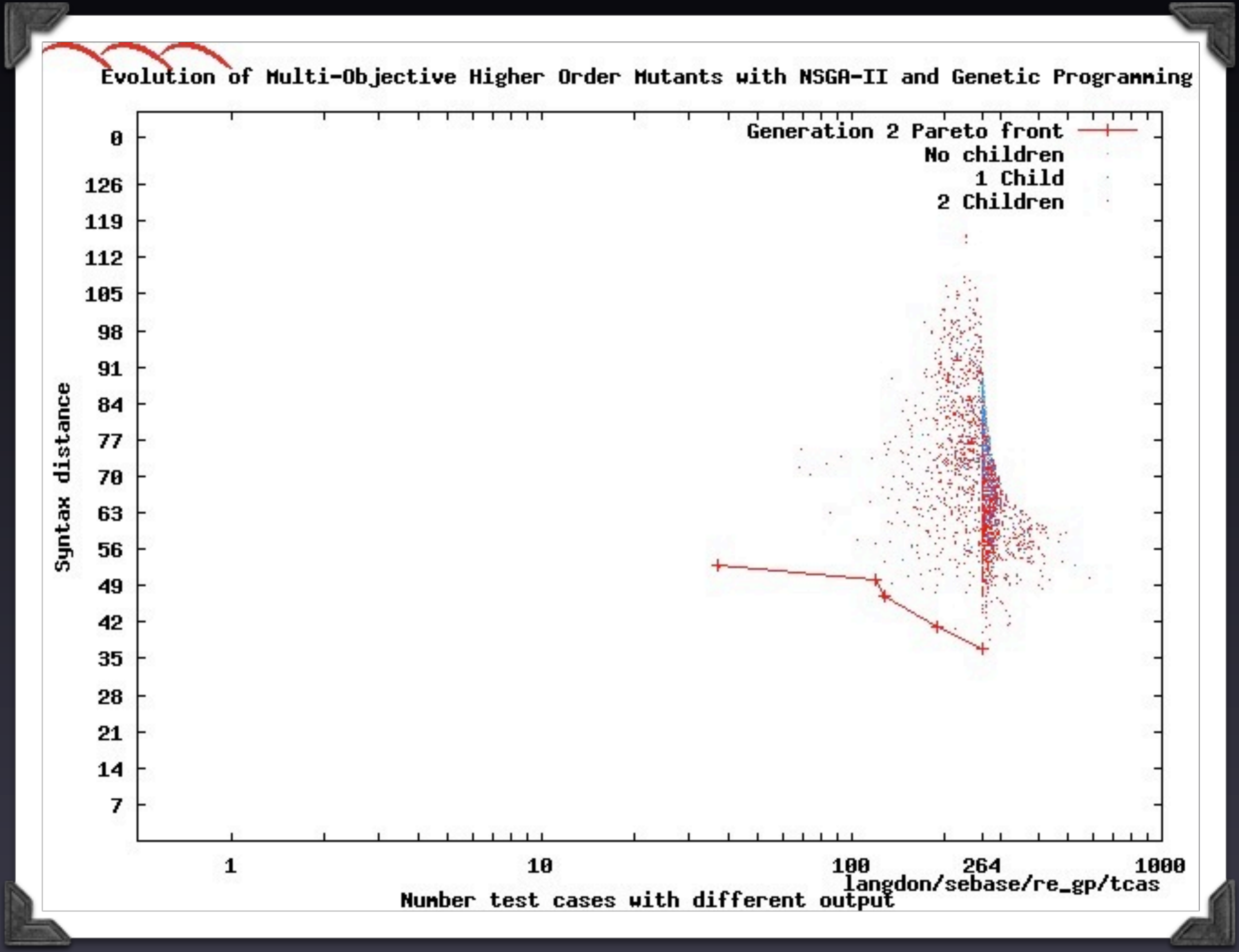
Multi Objective



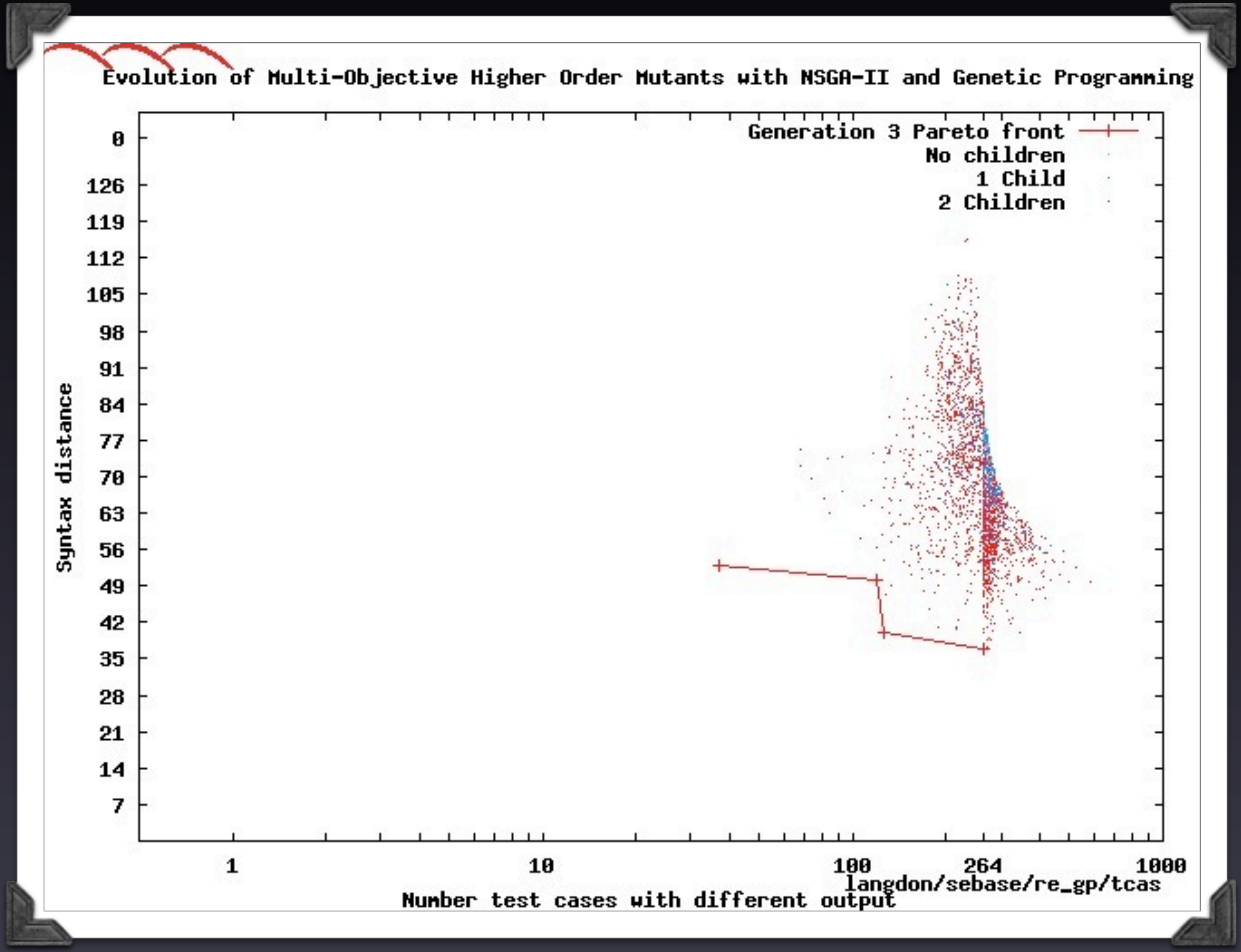
Multi Objective



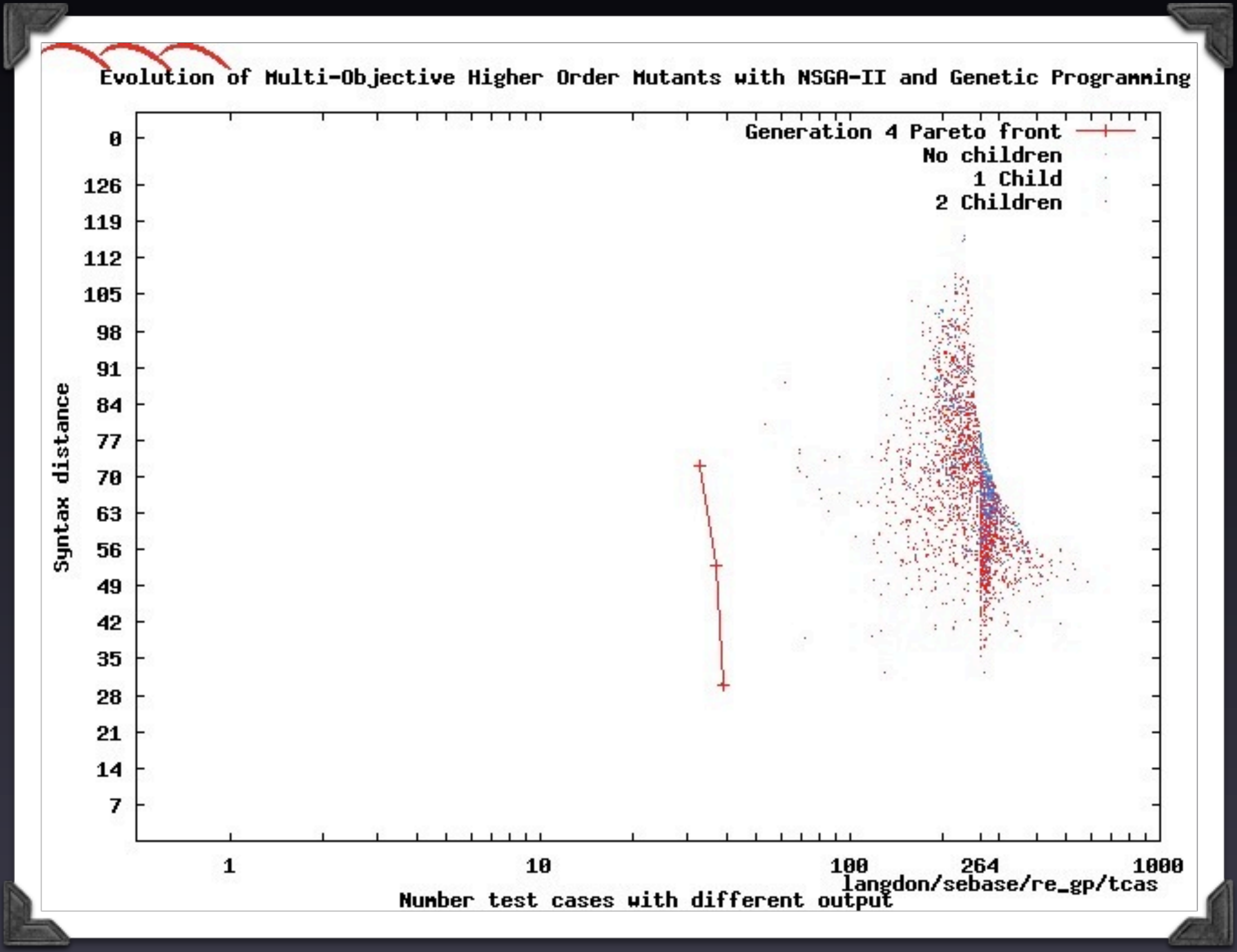
Multi Objective



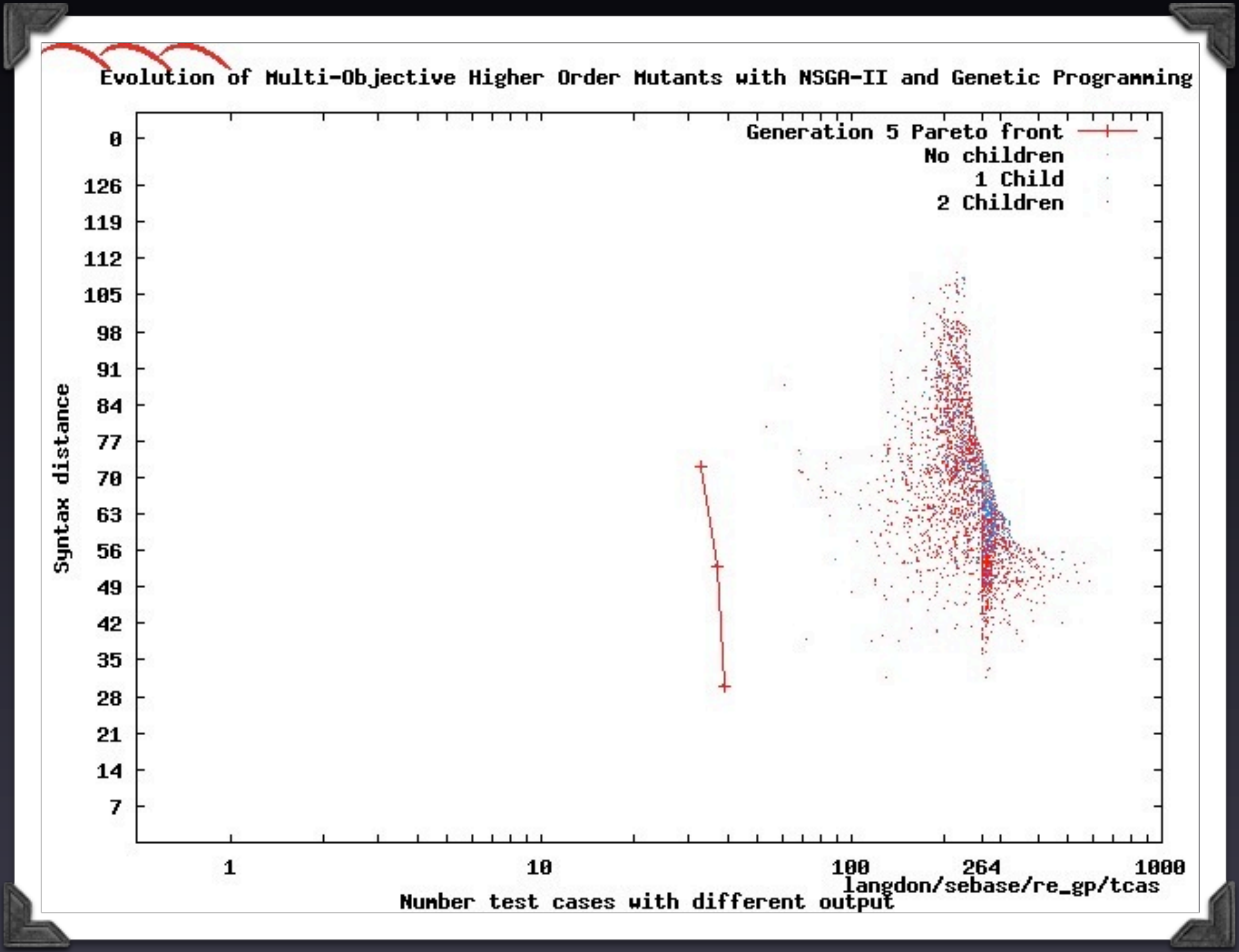
Multi Objective



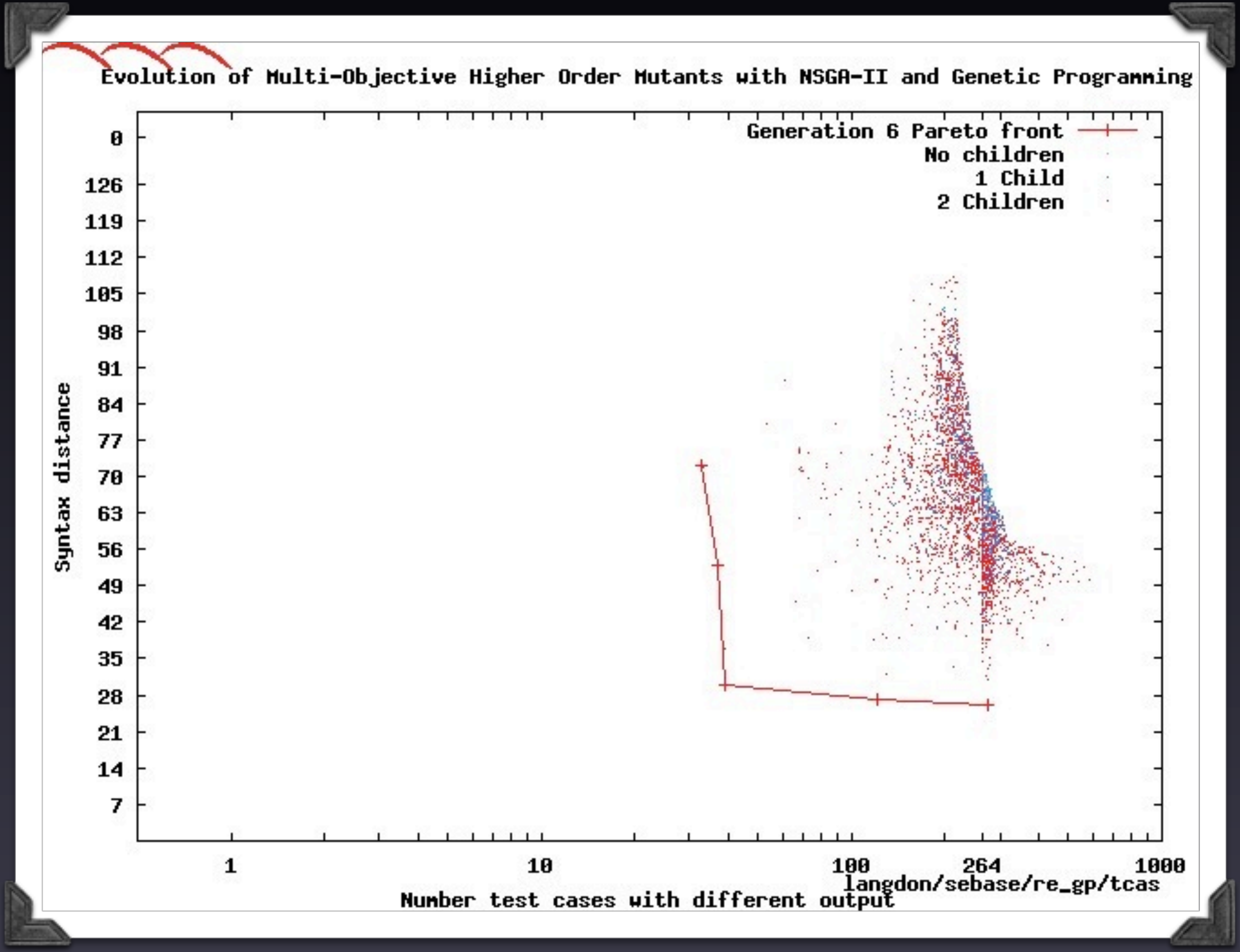
Multi Objective



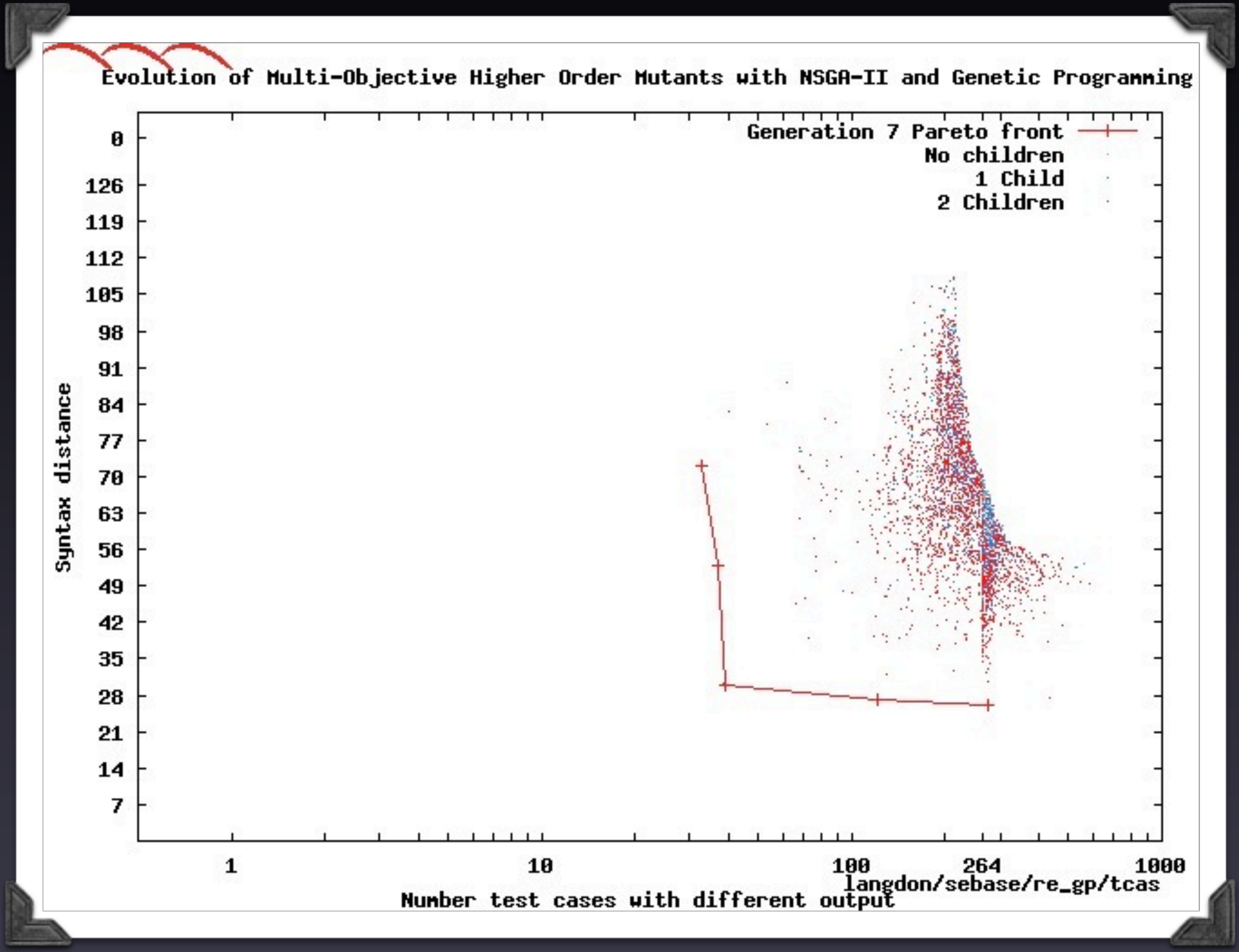
Multi Objective



Multi Objective

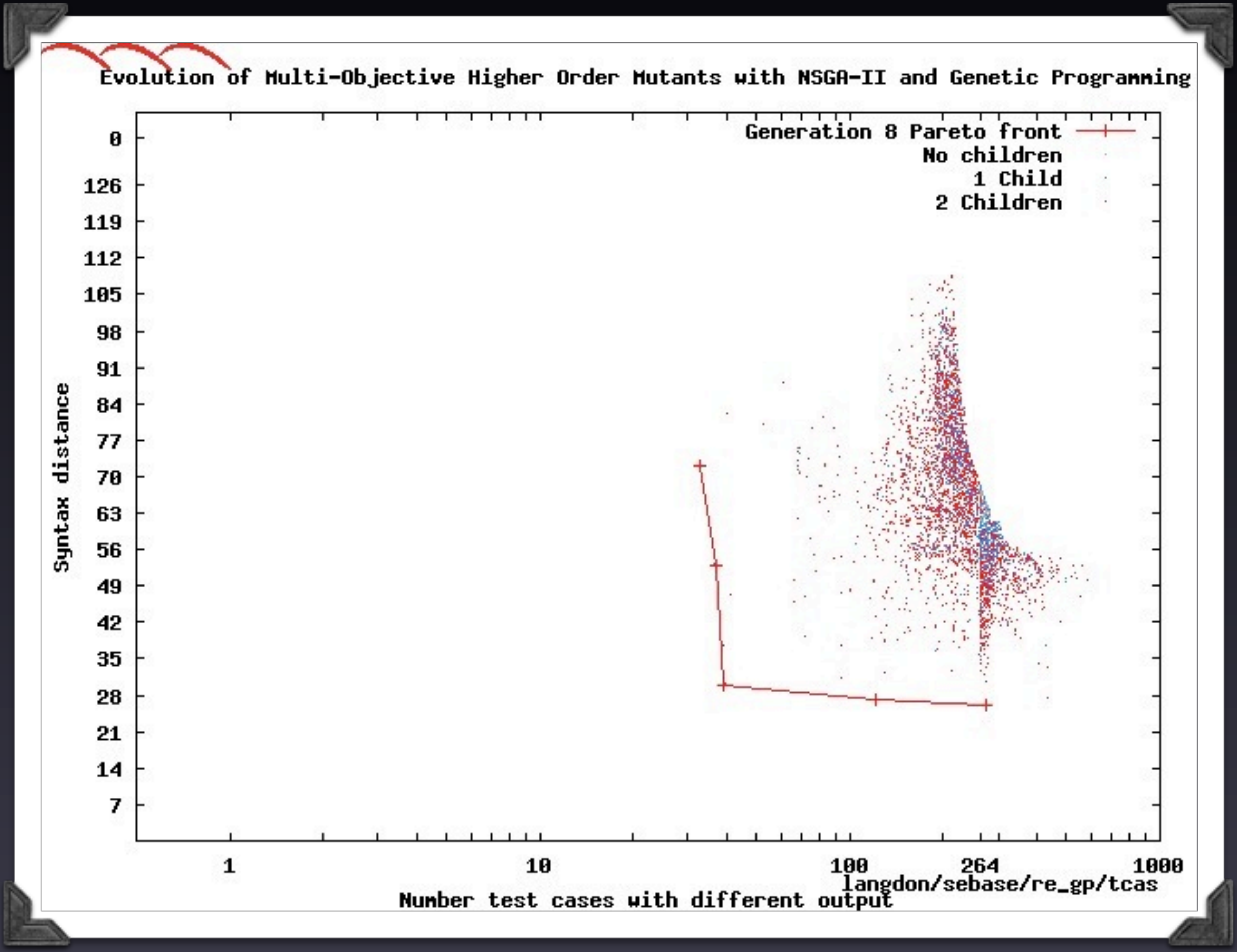


Multi Objective

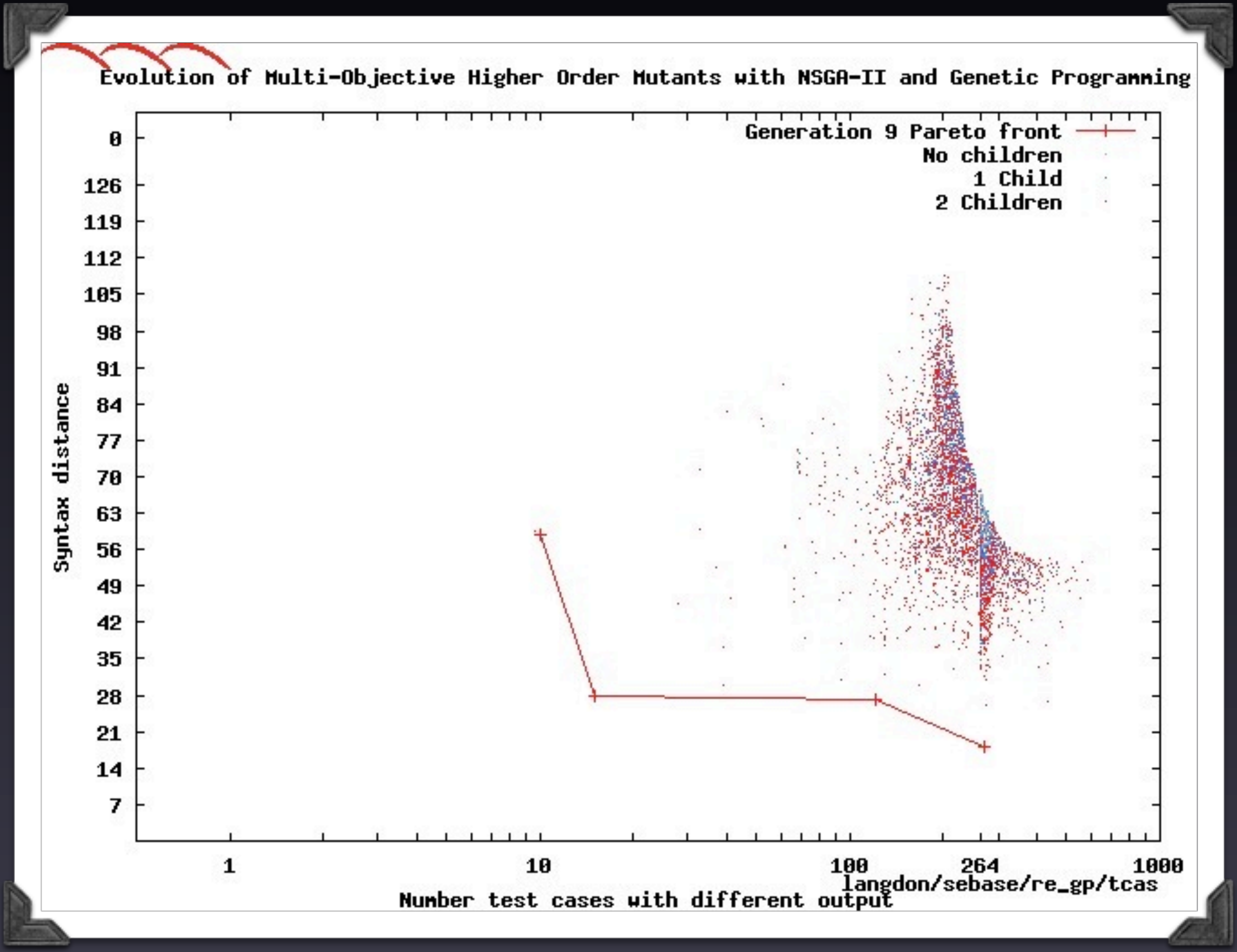


Overview FOM Restriction HOM Testing **Approaches** Classification Future Work

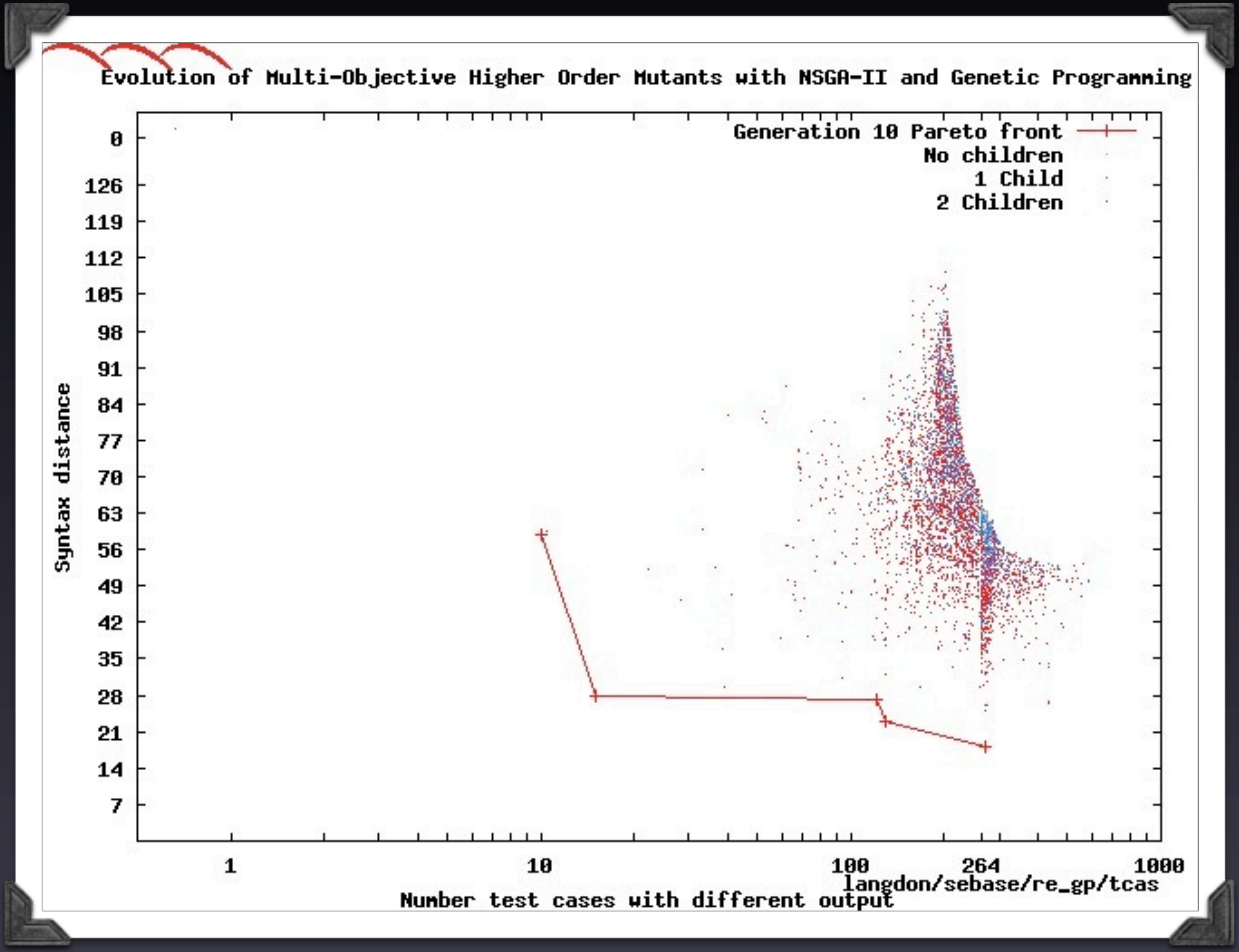
Multi Objective



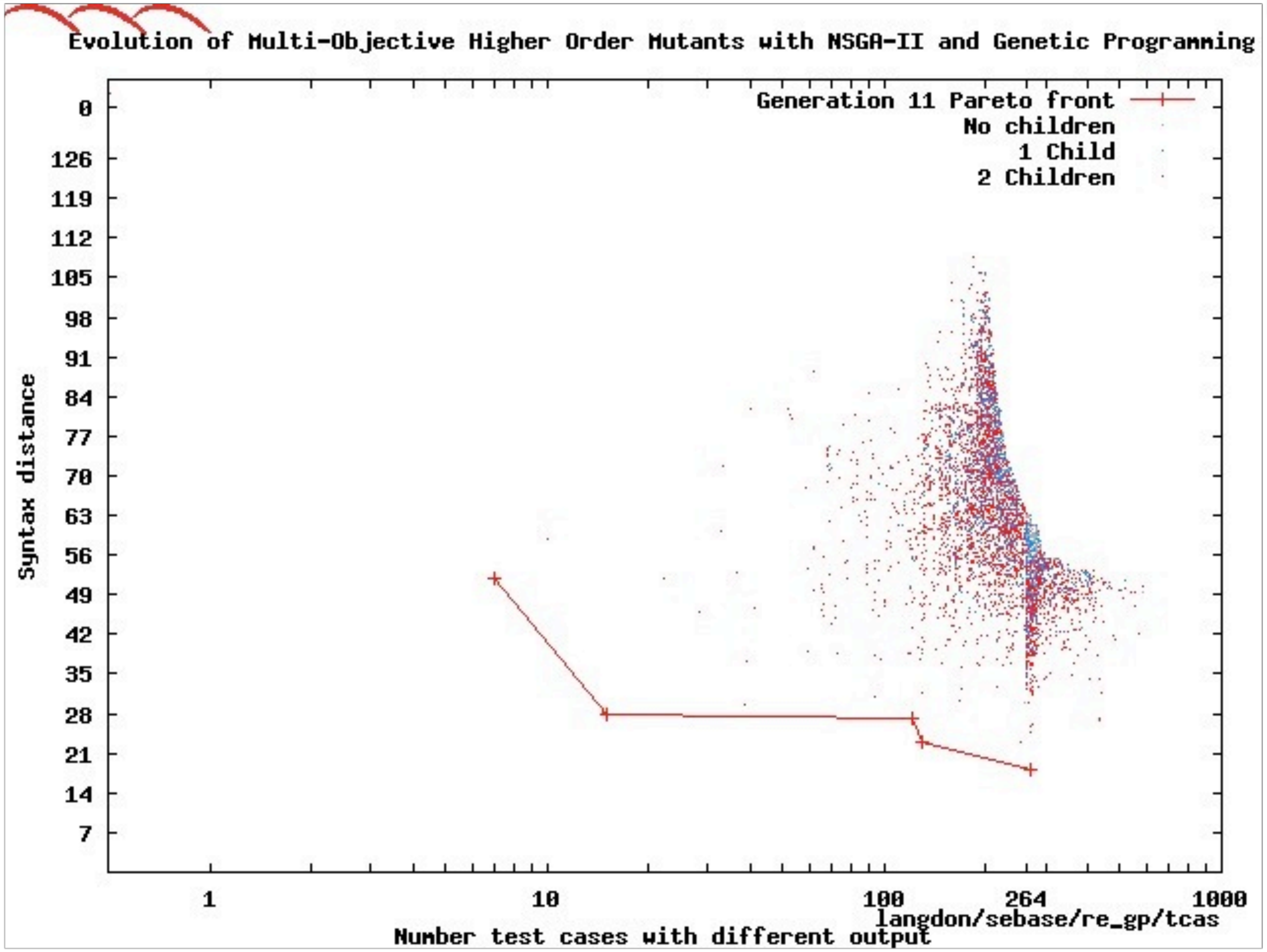
Multi Objective



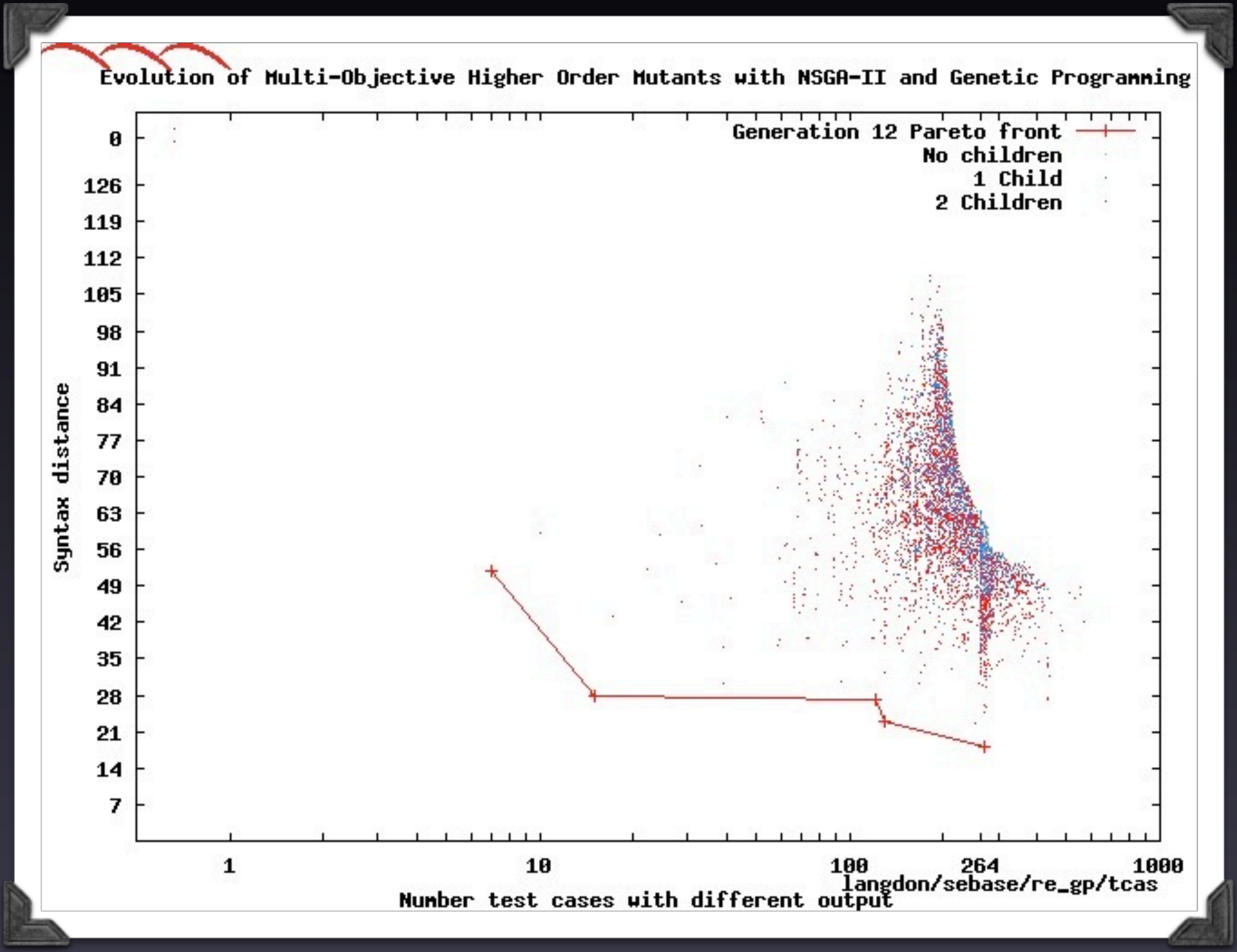
Multi Objective



Multi Objective

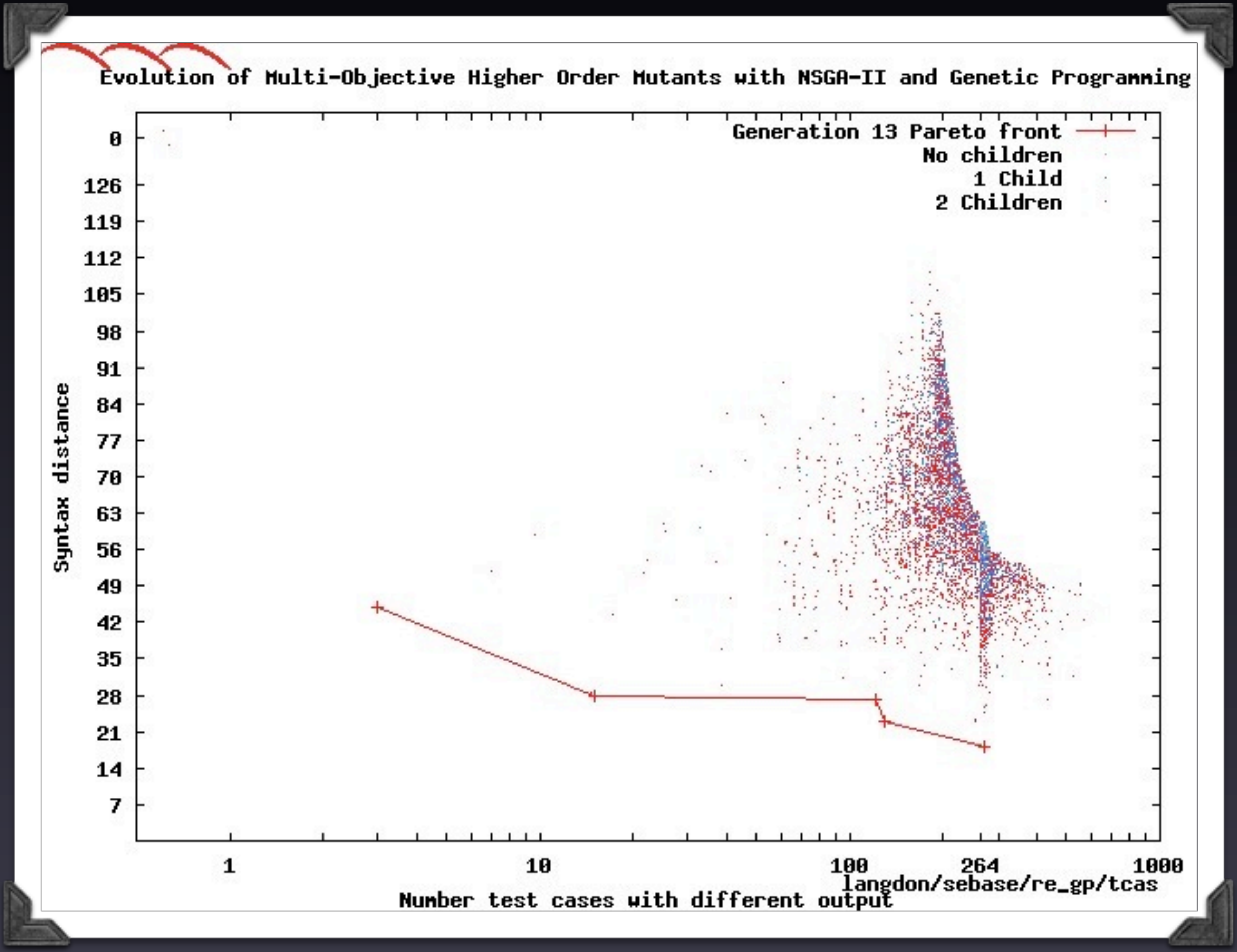


Multi Objective

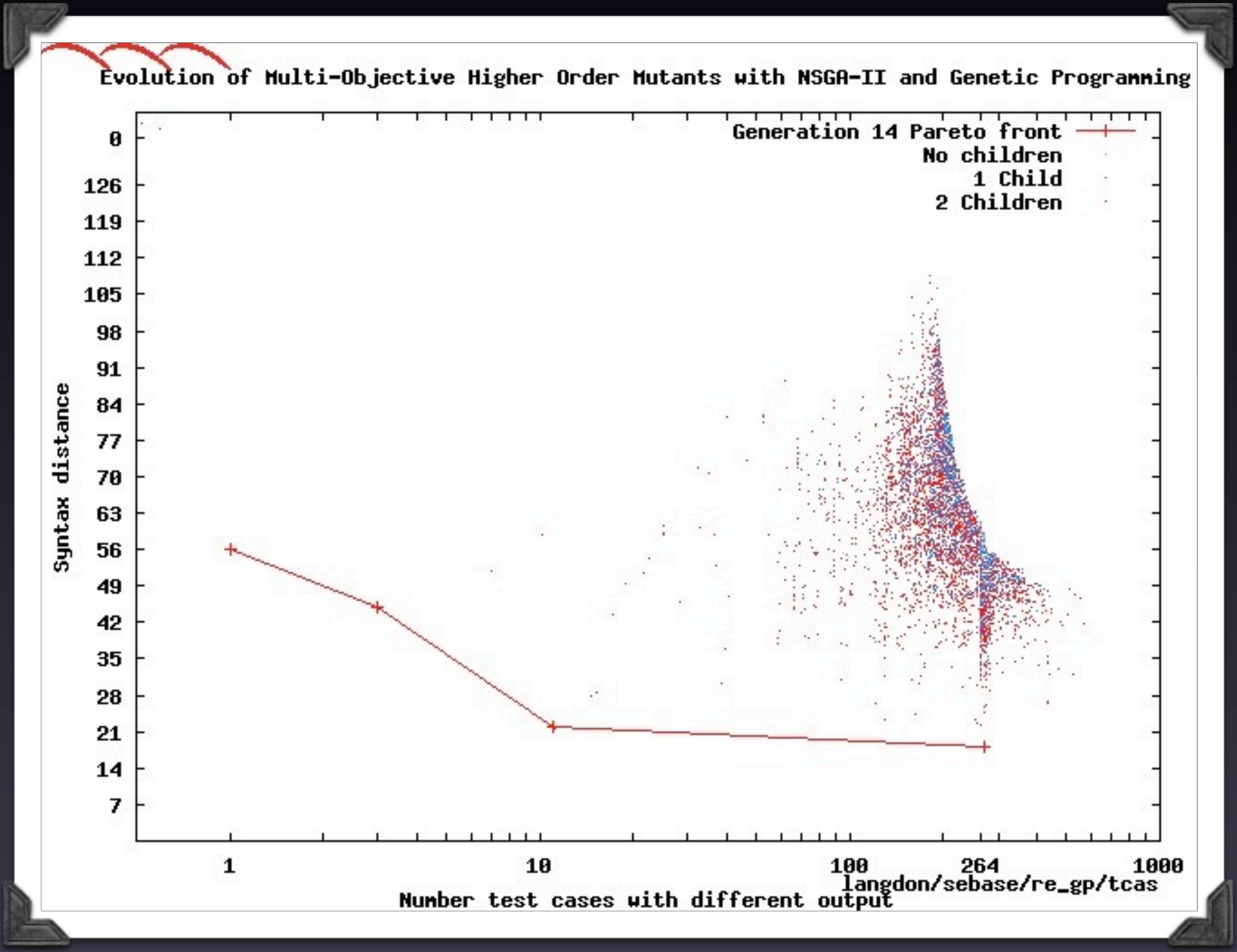


Overview FOM Restriction HOM Testing **Approaches** Classification Future Work

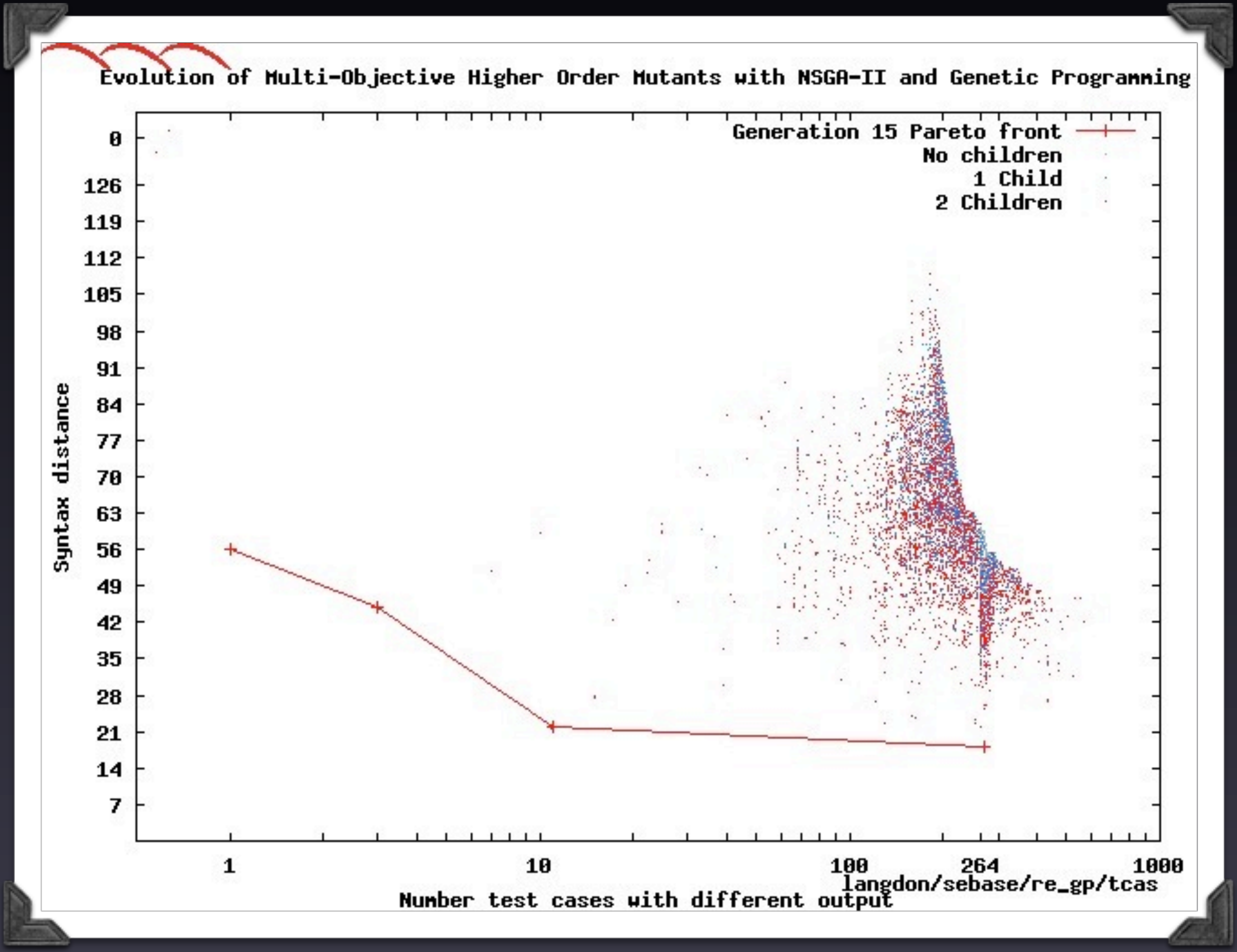
Multi Objective



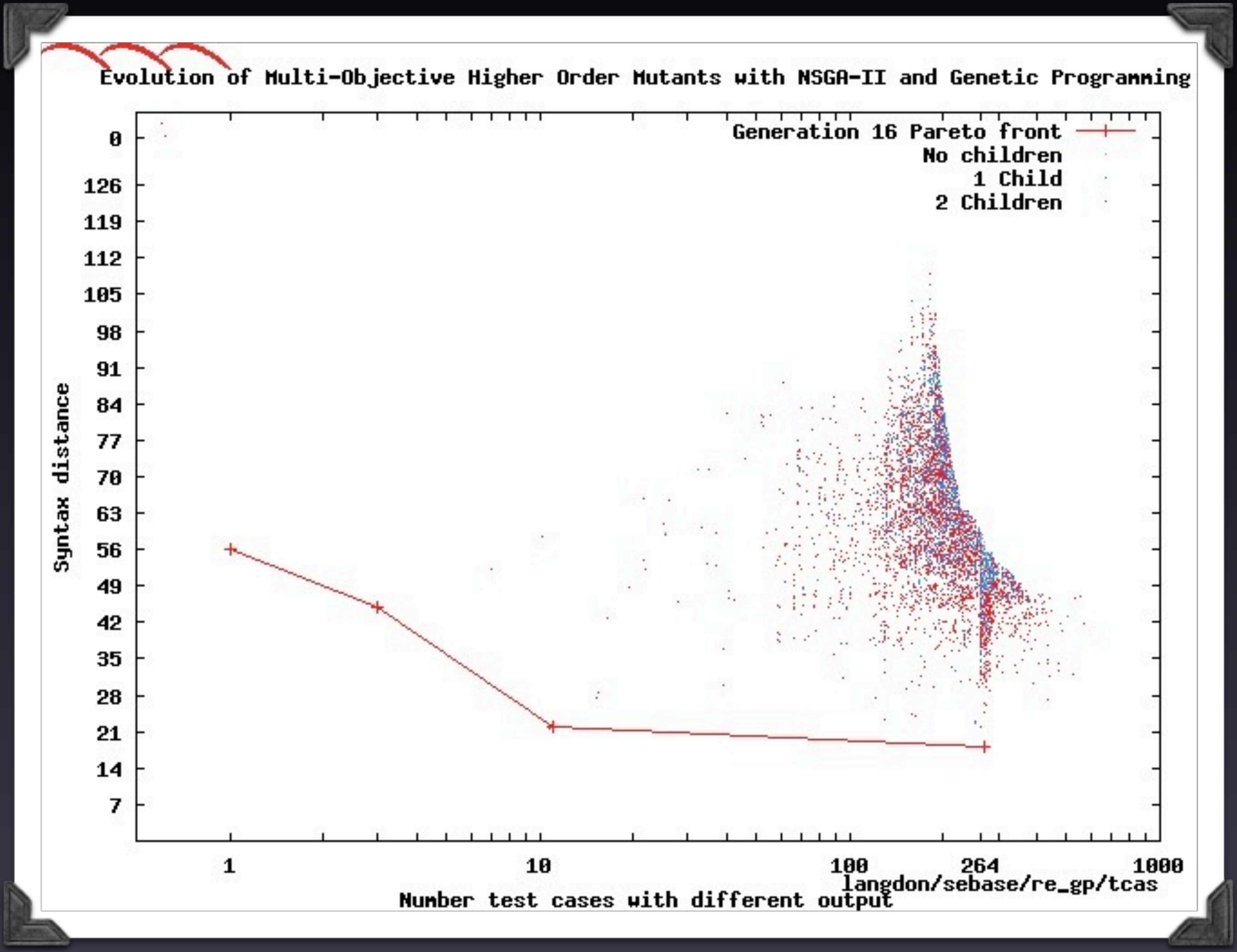
Multi Objective



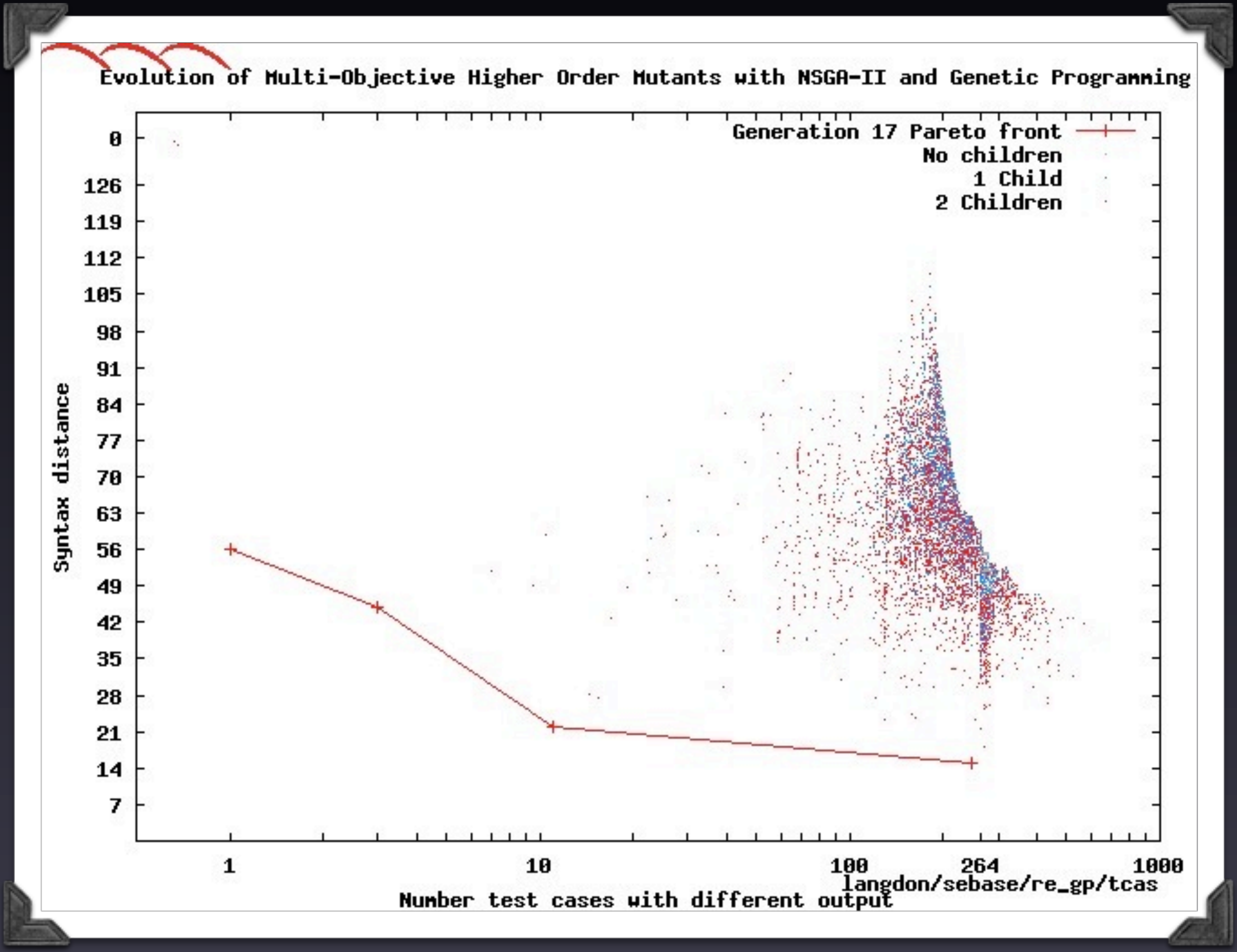
Multi Objective



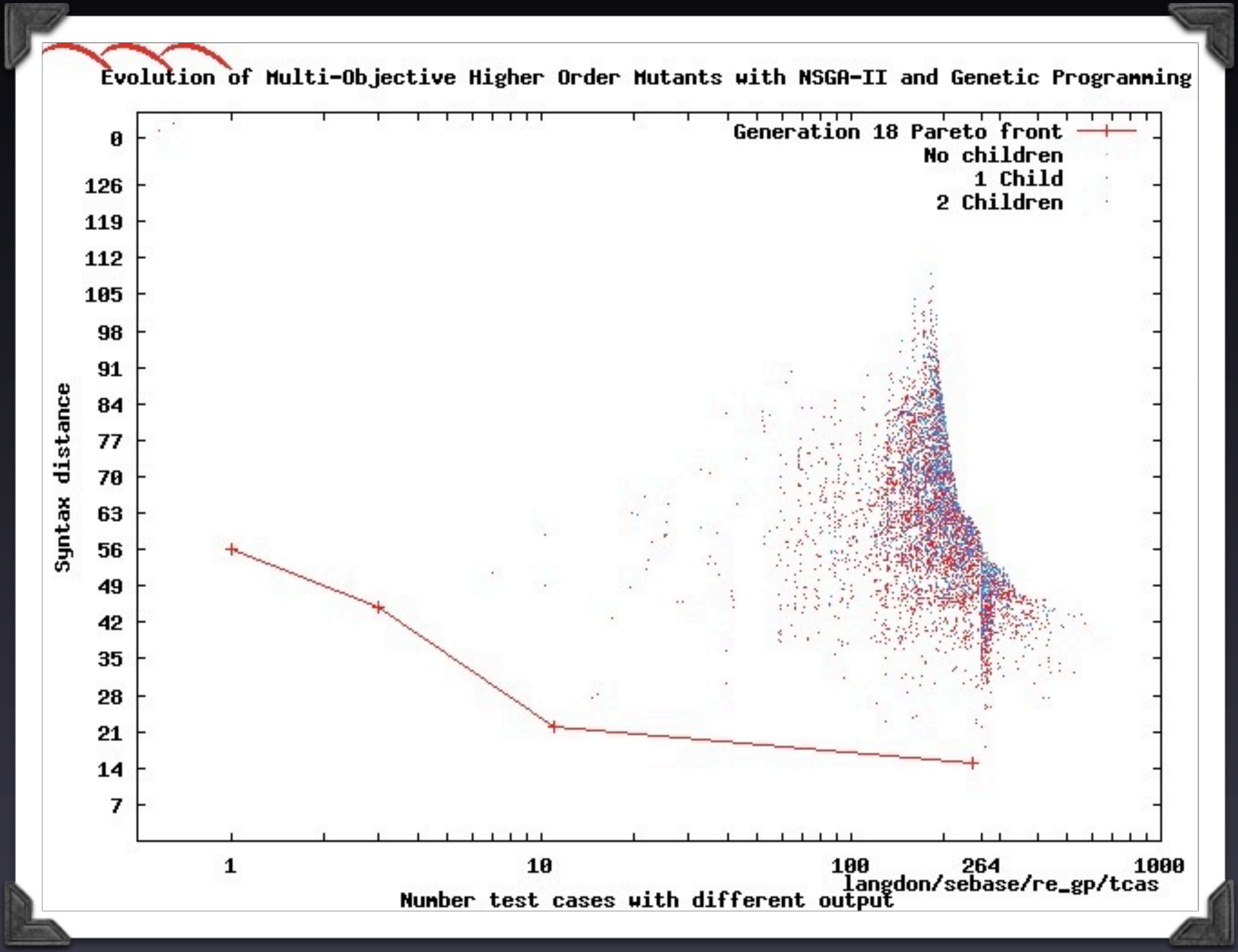
Multi Objective



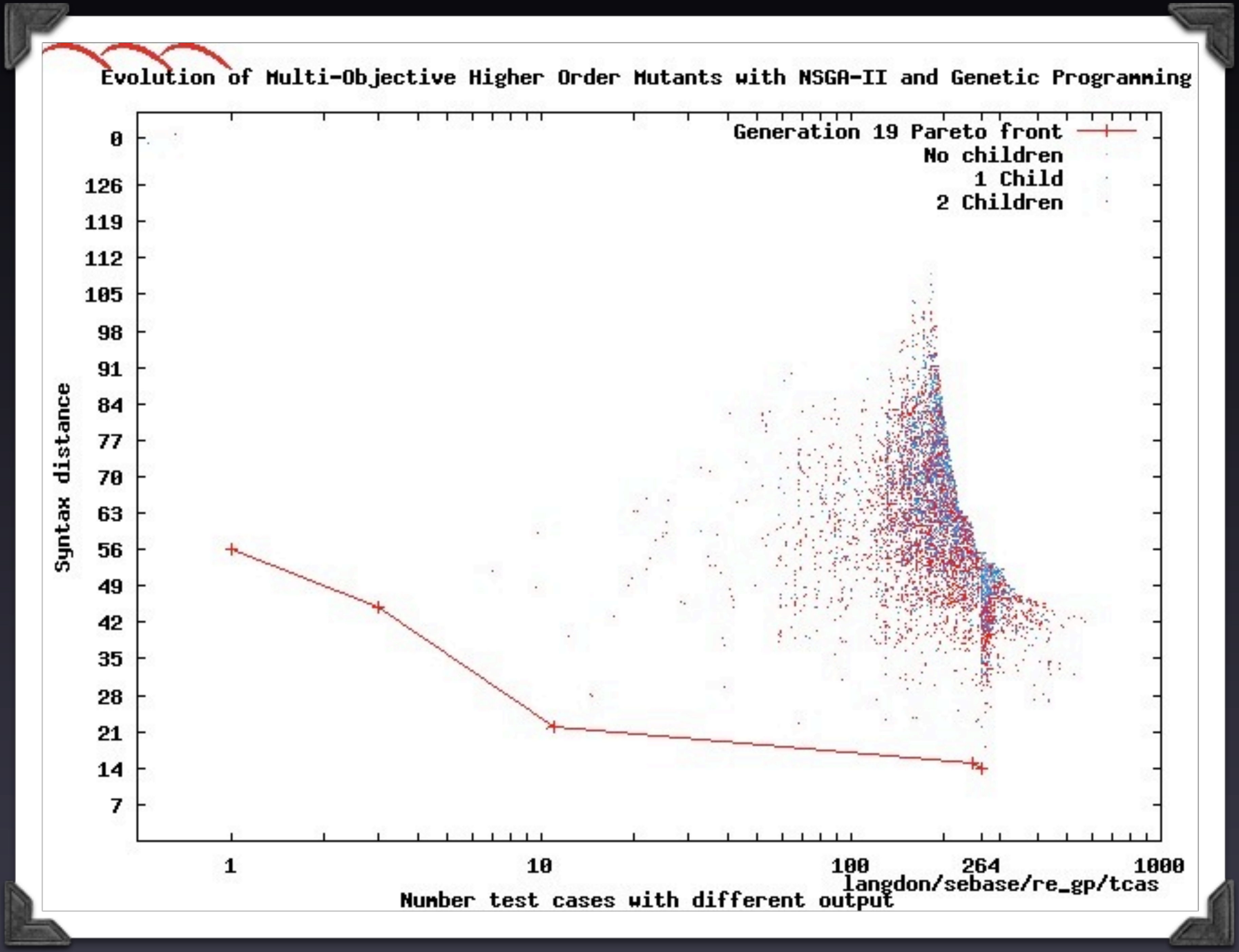
Multi Objective



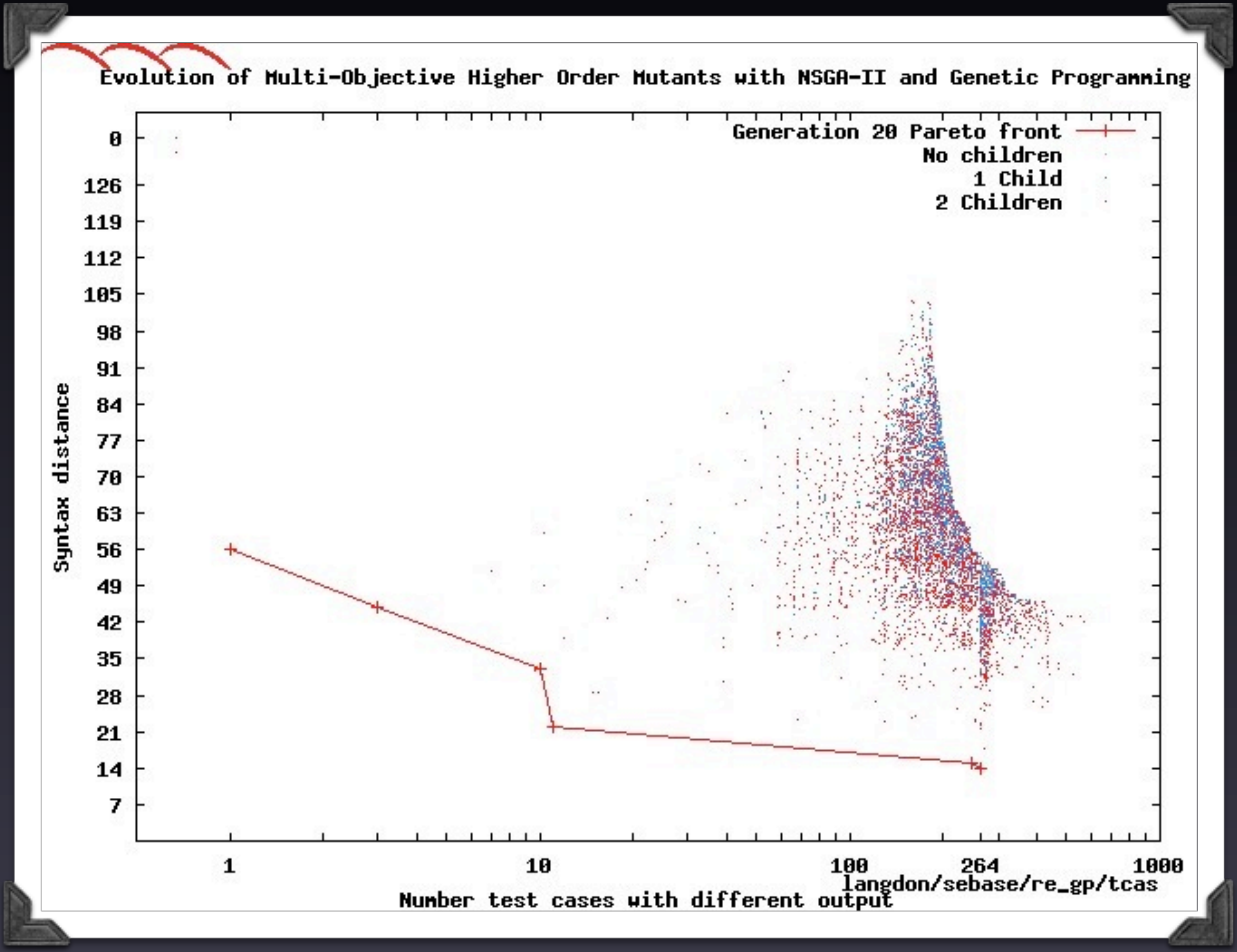
Multi Objective



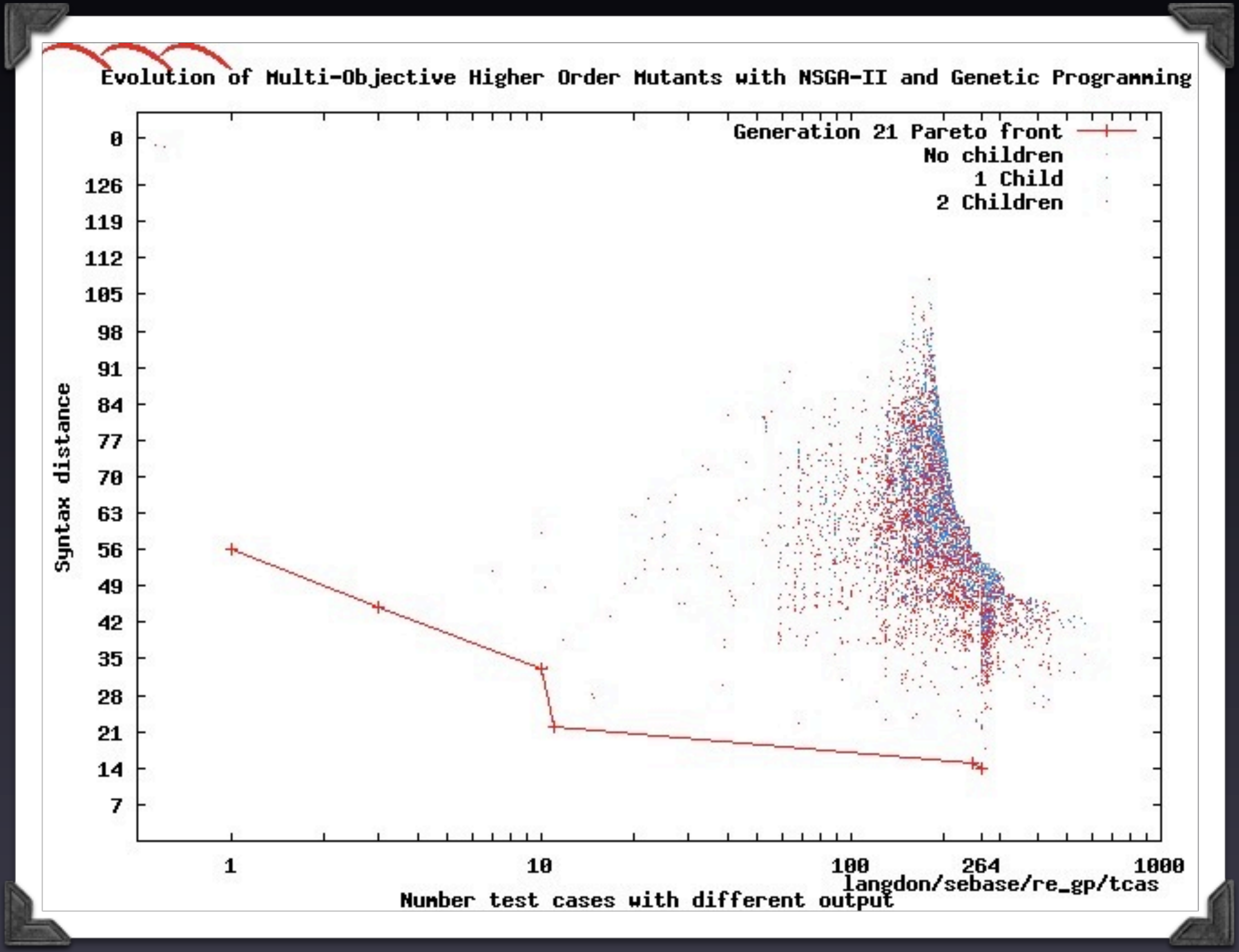
Multi Objective



Multi Objective

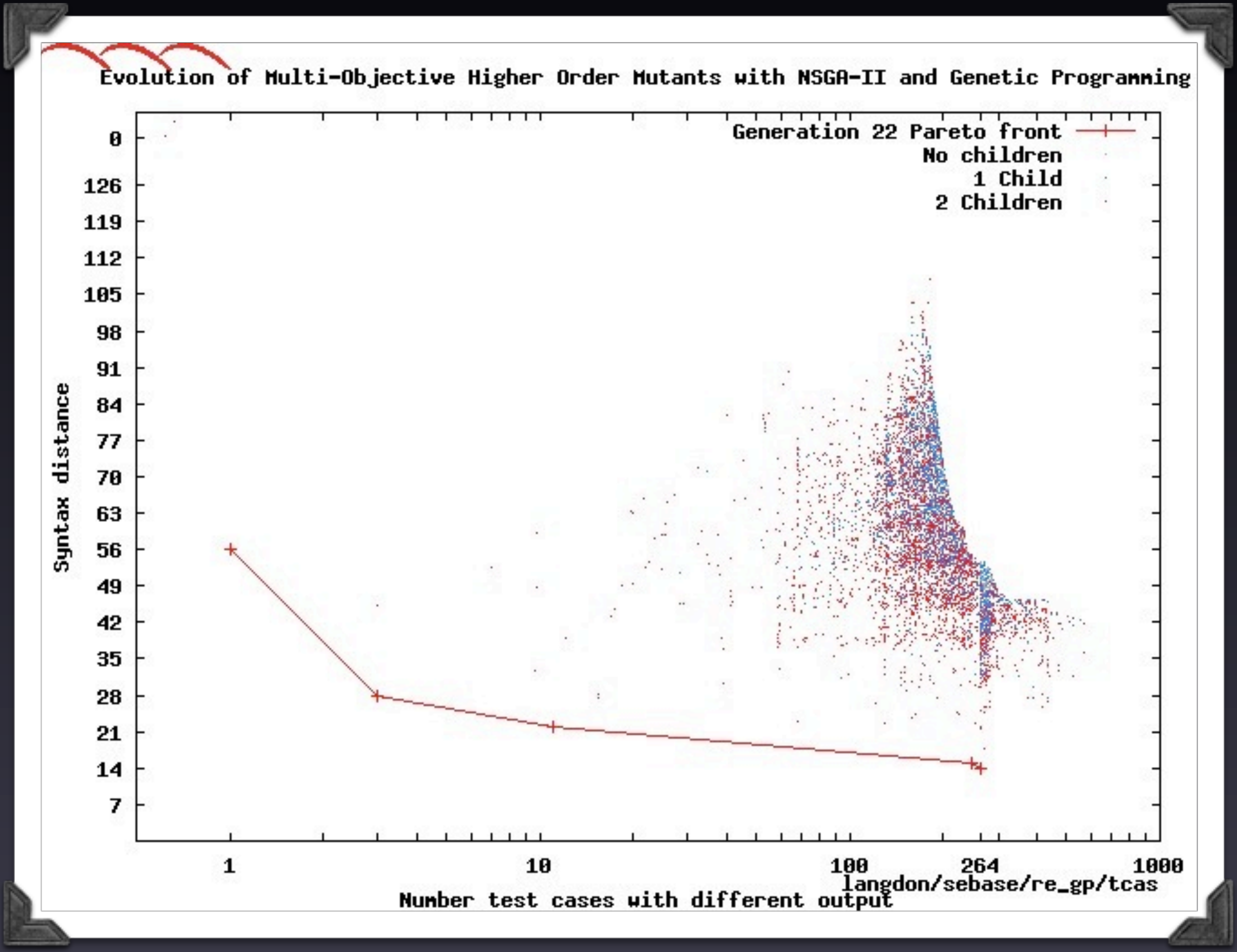


Multi Objective

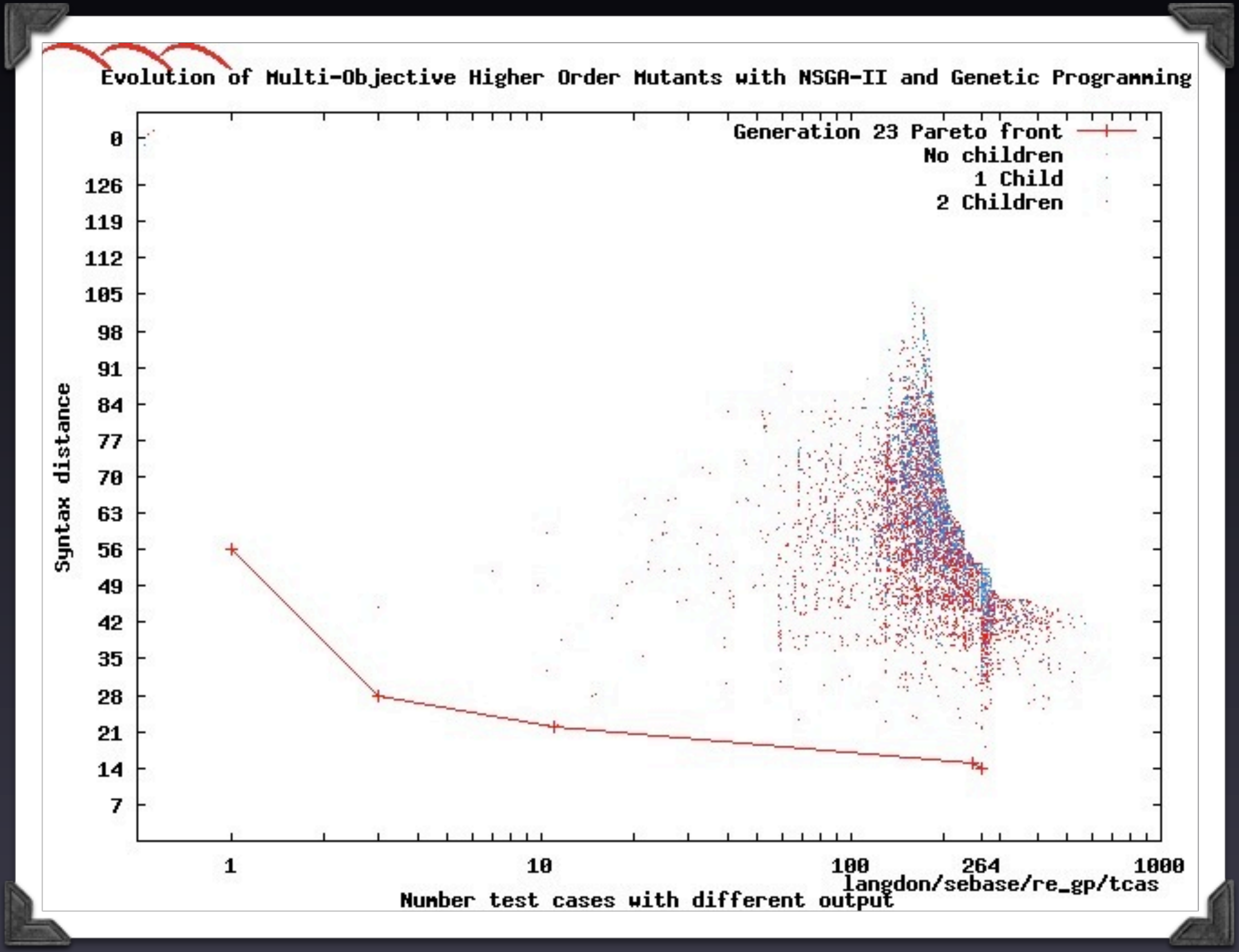


Overview FOM Restriction HOM Testing **Approaches** Classification Future Work

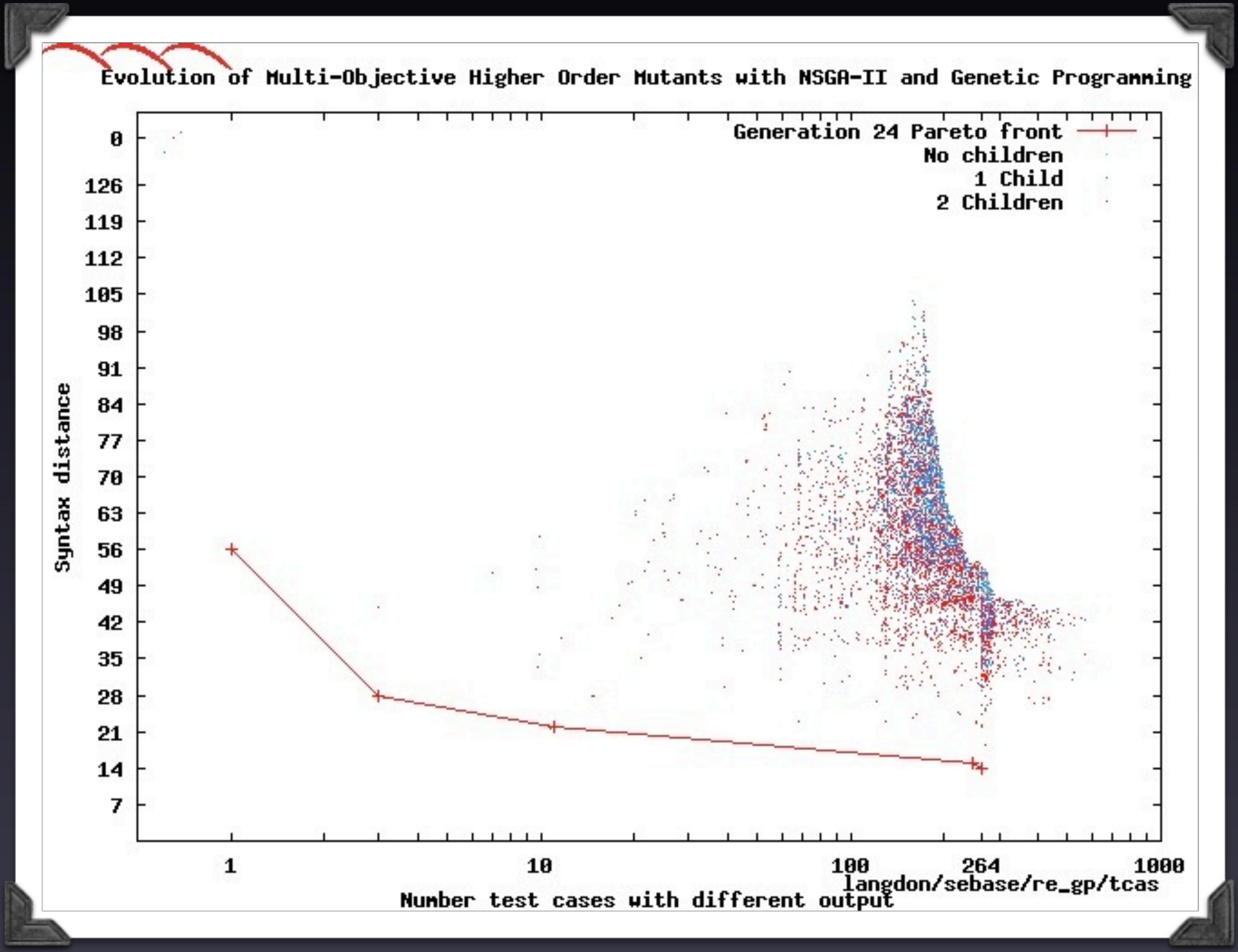
Multi Objective



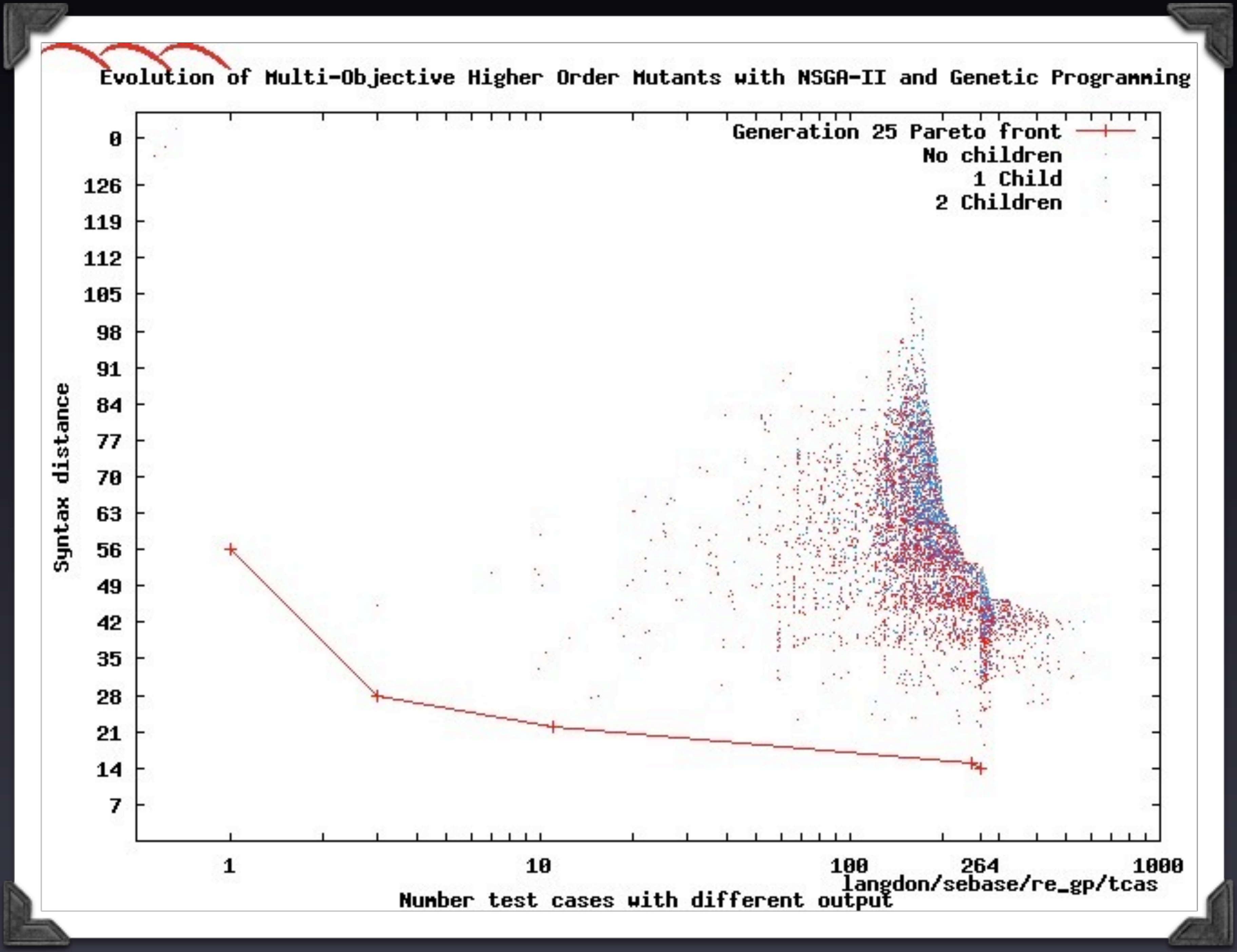
Multi Objective



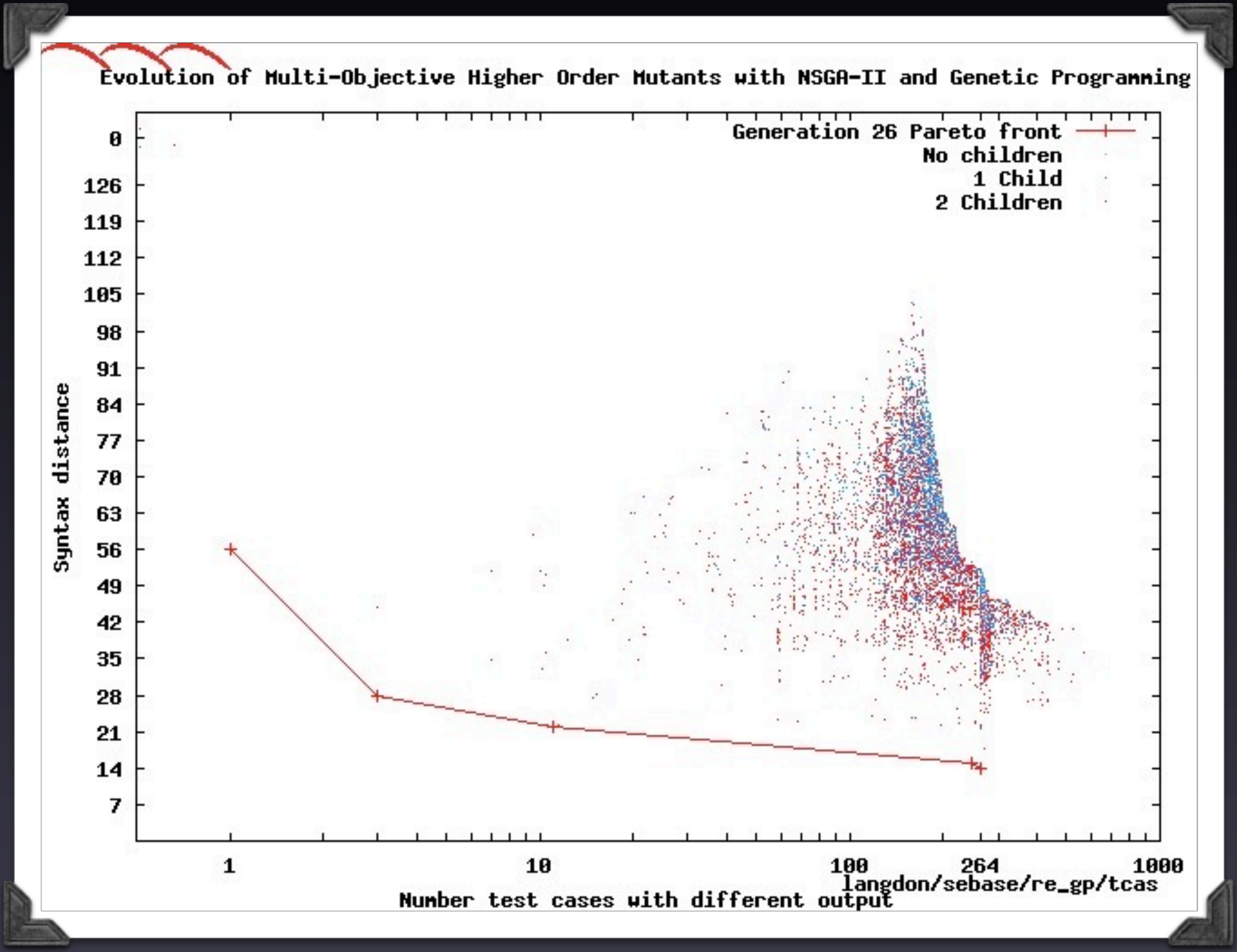
Multi Objective



Multi Objective

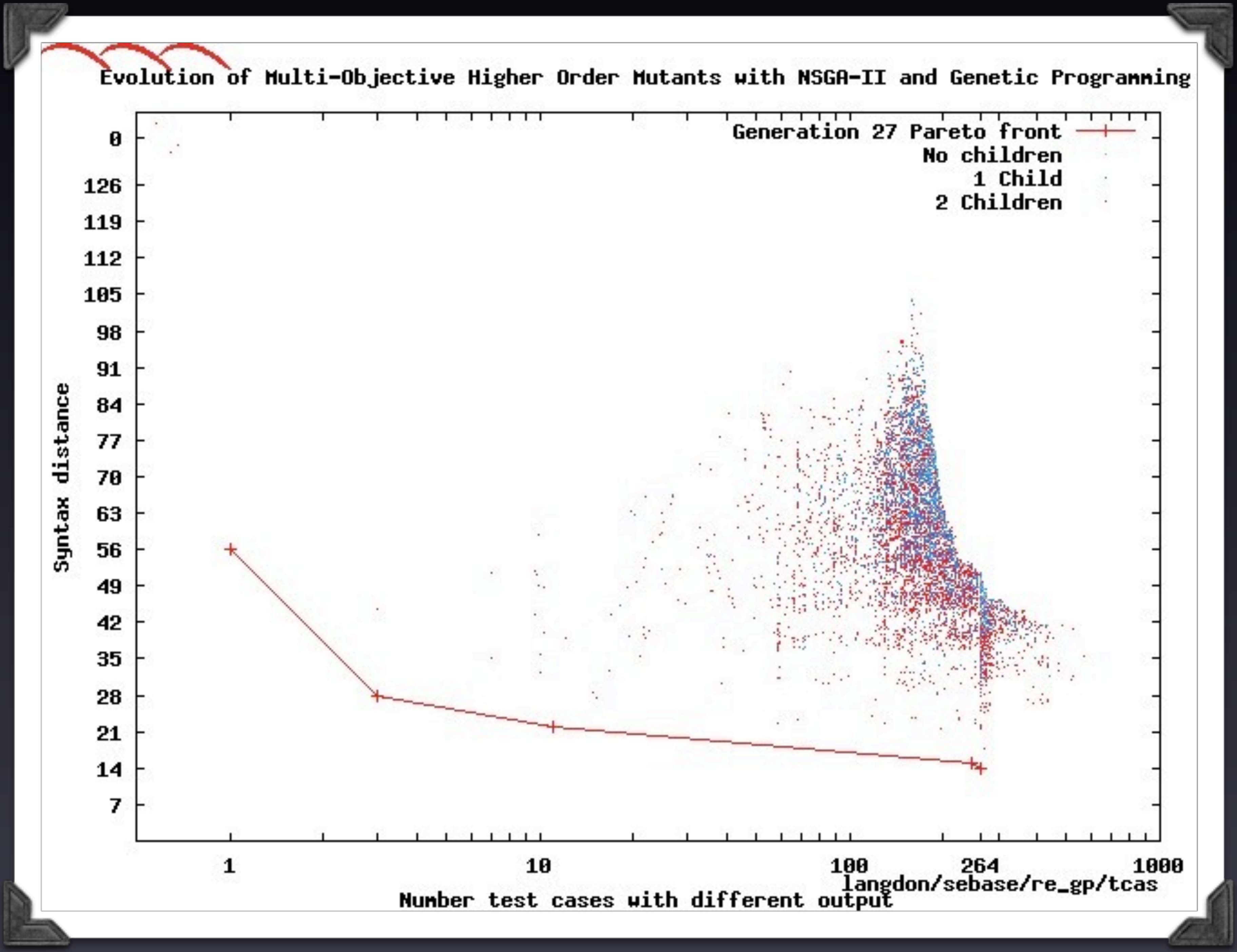


Multi Objective

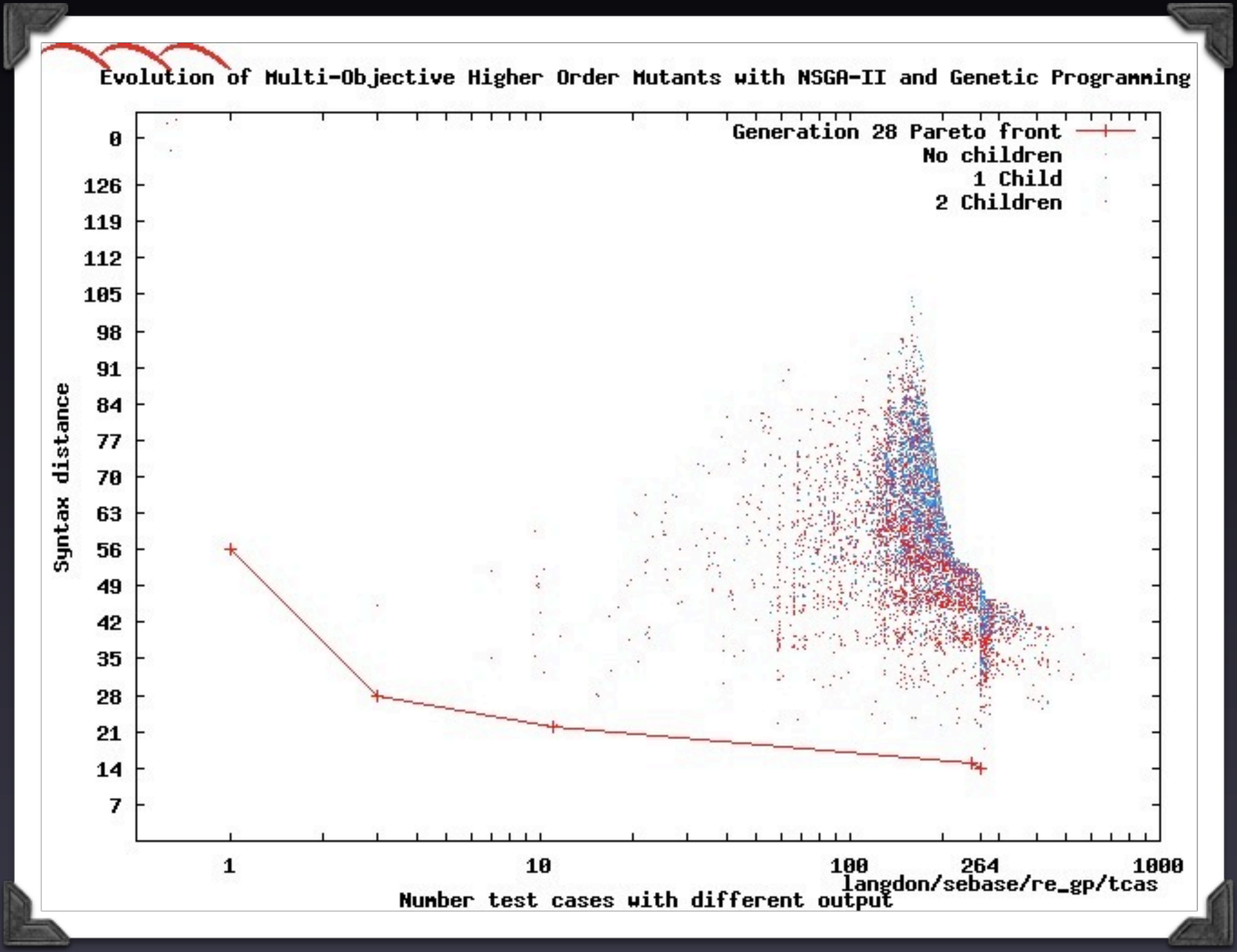


Overview FOM Restriction HOM Testing **Approaches** Classification Future Work

Multi Objective

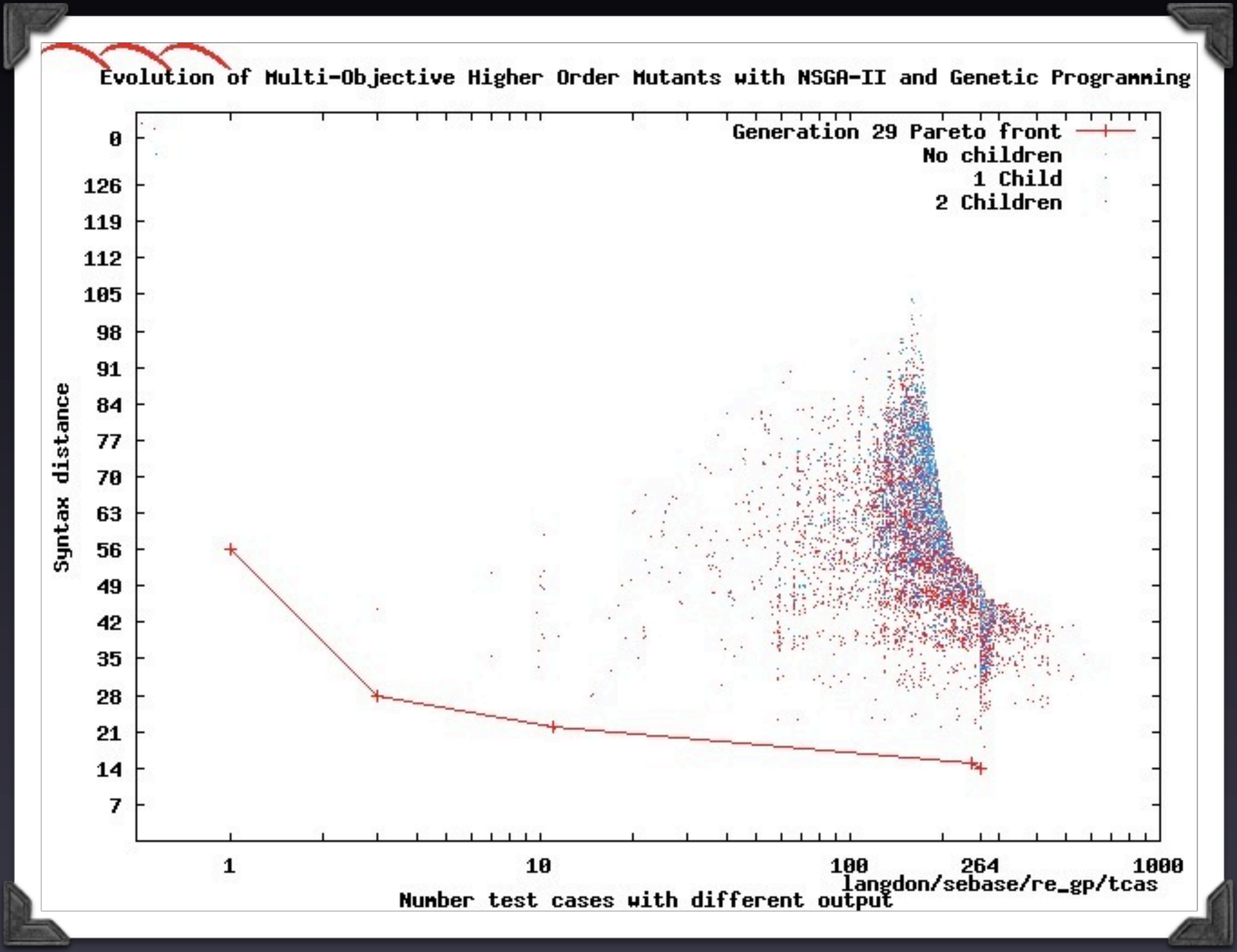


Multi Objective



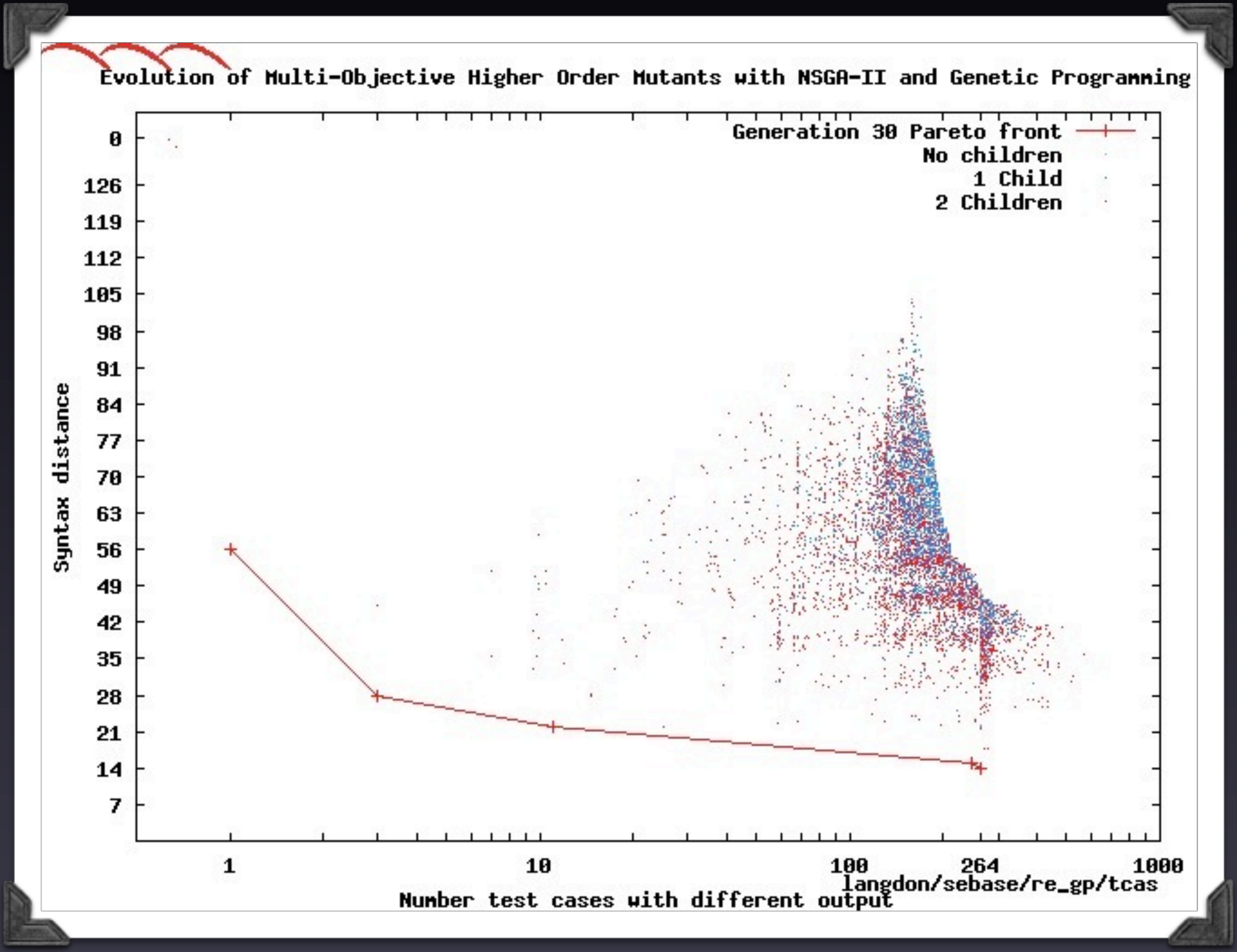
Overview FOM Restriction HOM Testing **Approaches** Classification Future Work

Multi Objective

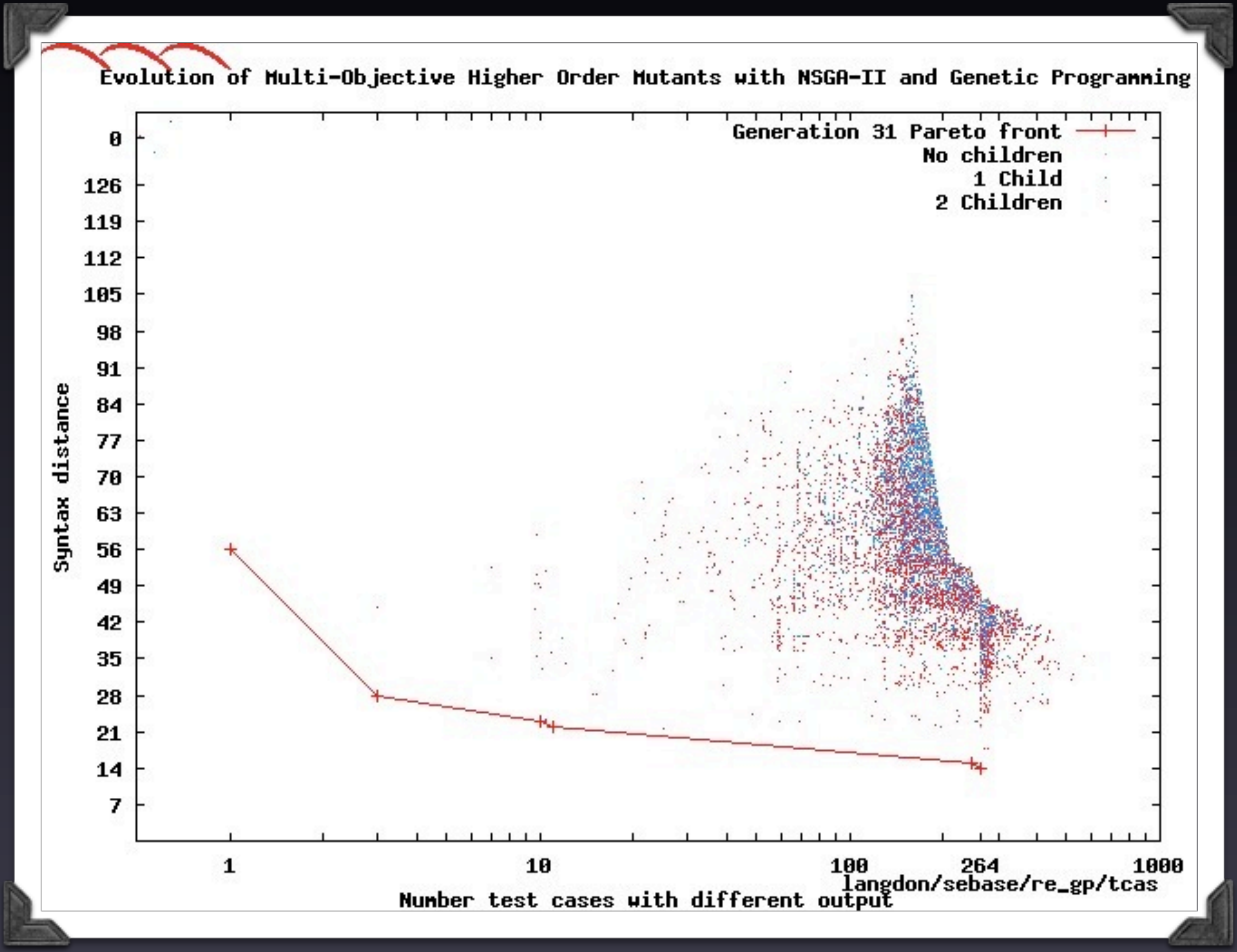


Overview FOM Restriction HOM Testing **Approaches** Classification Future Work

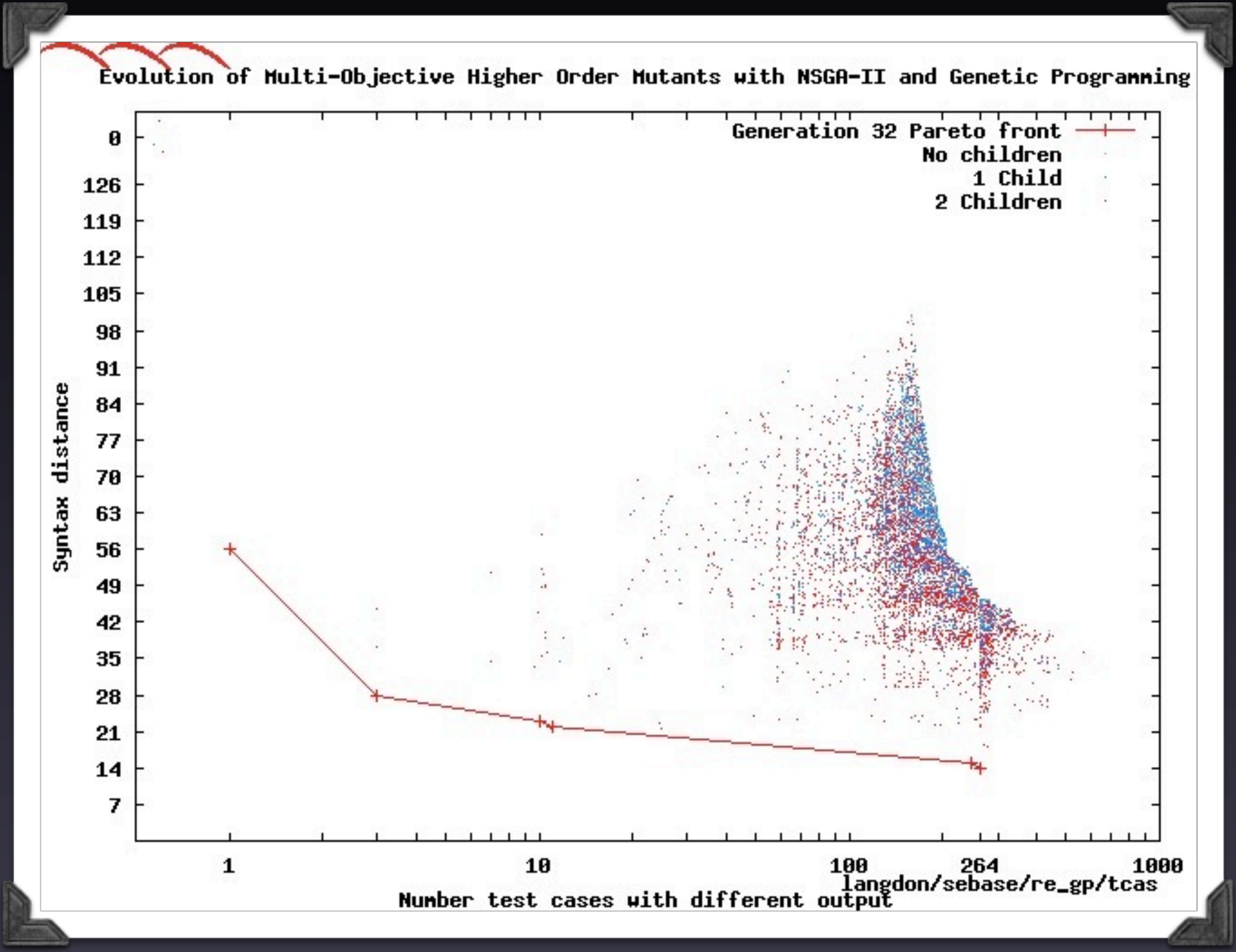
Multi Objective



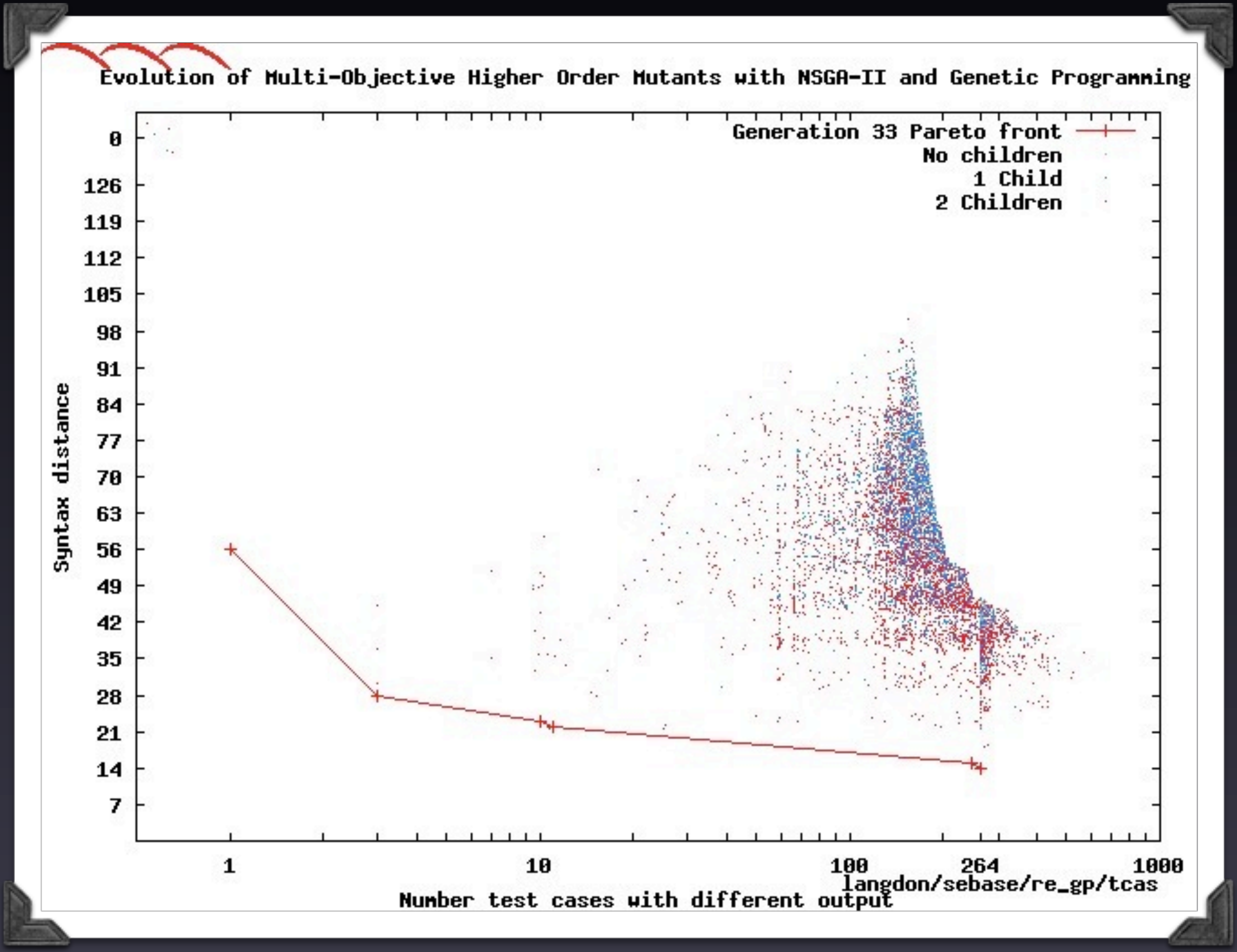
Multi Objective



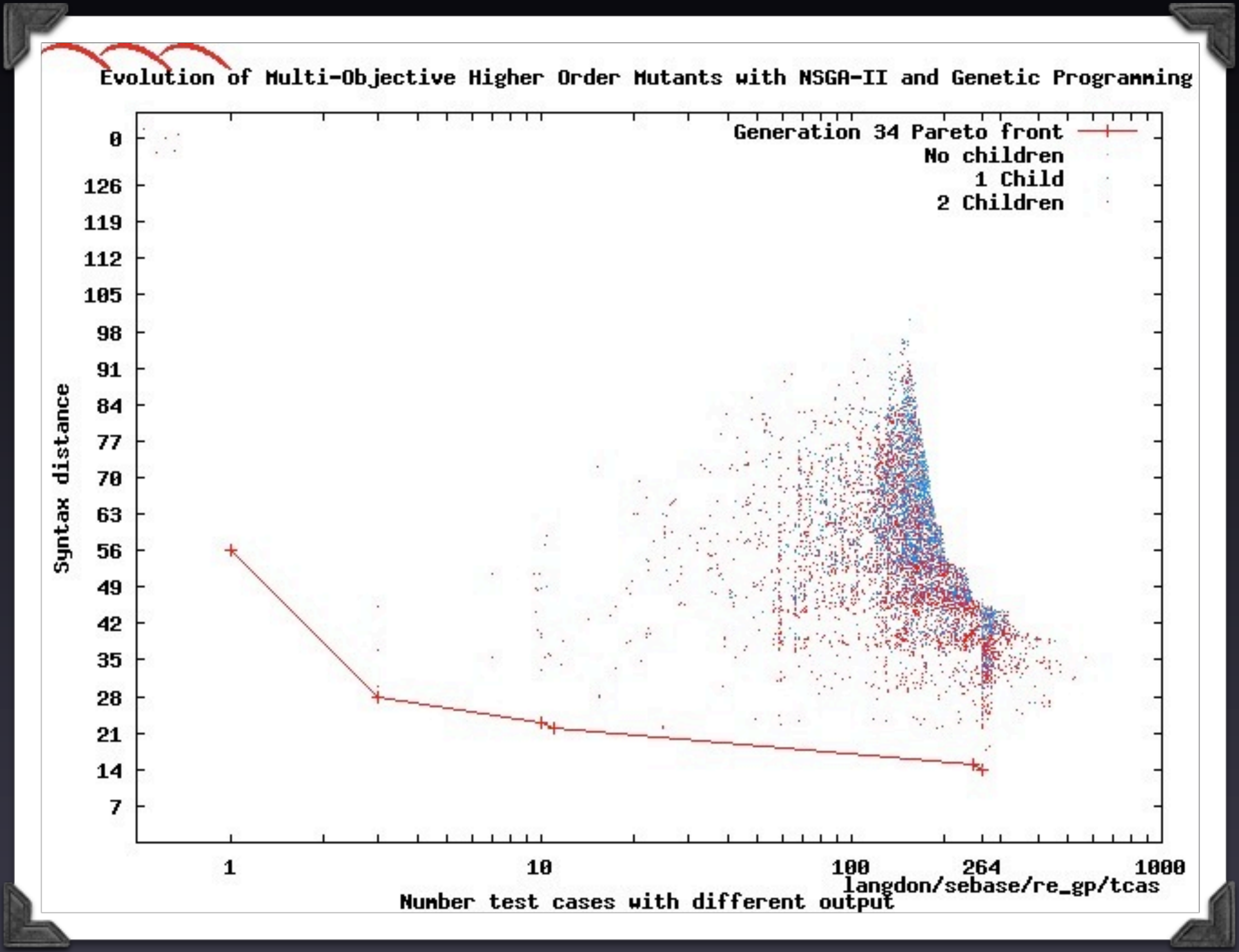
Multi Objective



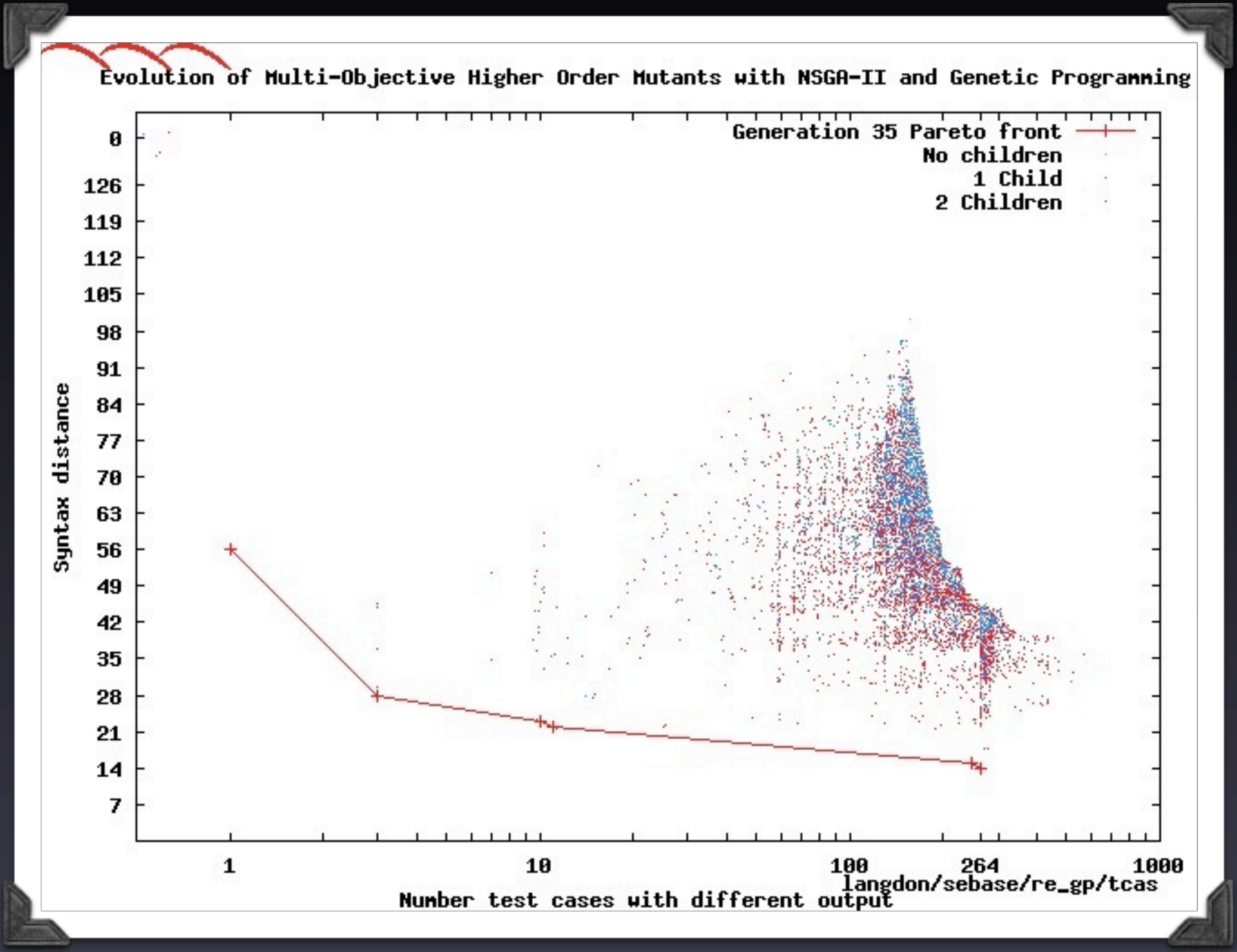
Multi Objective



Multi Objective

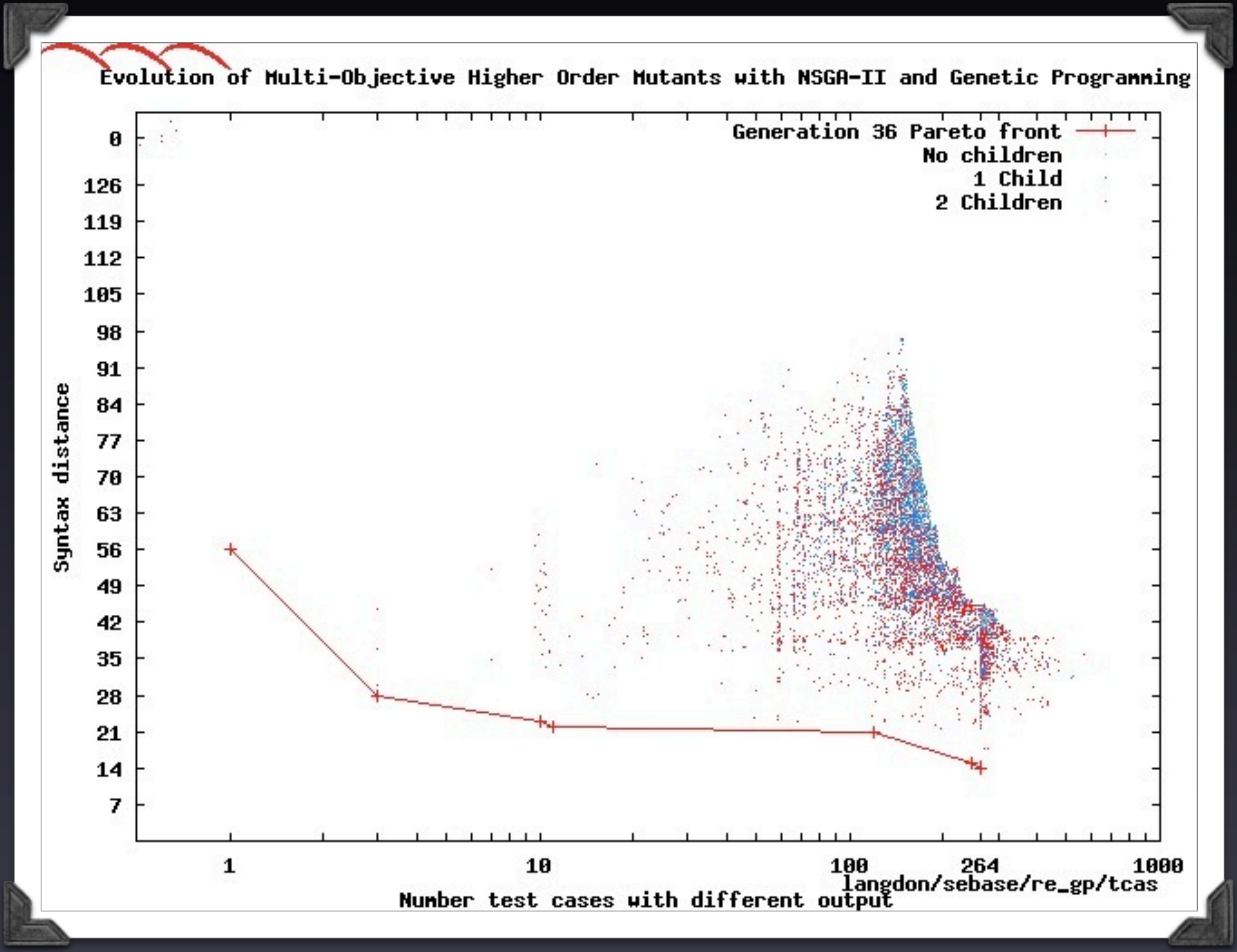


Multi Objective

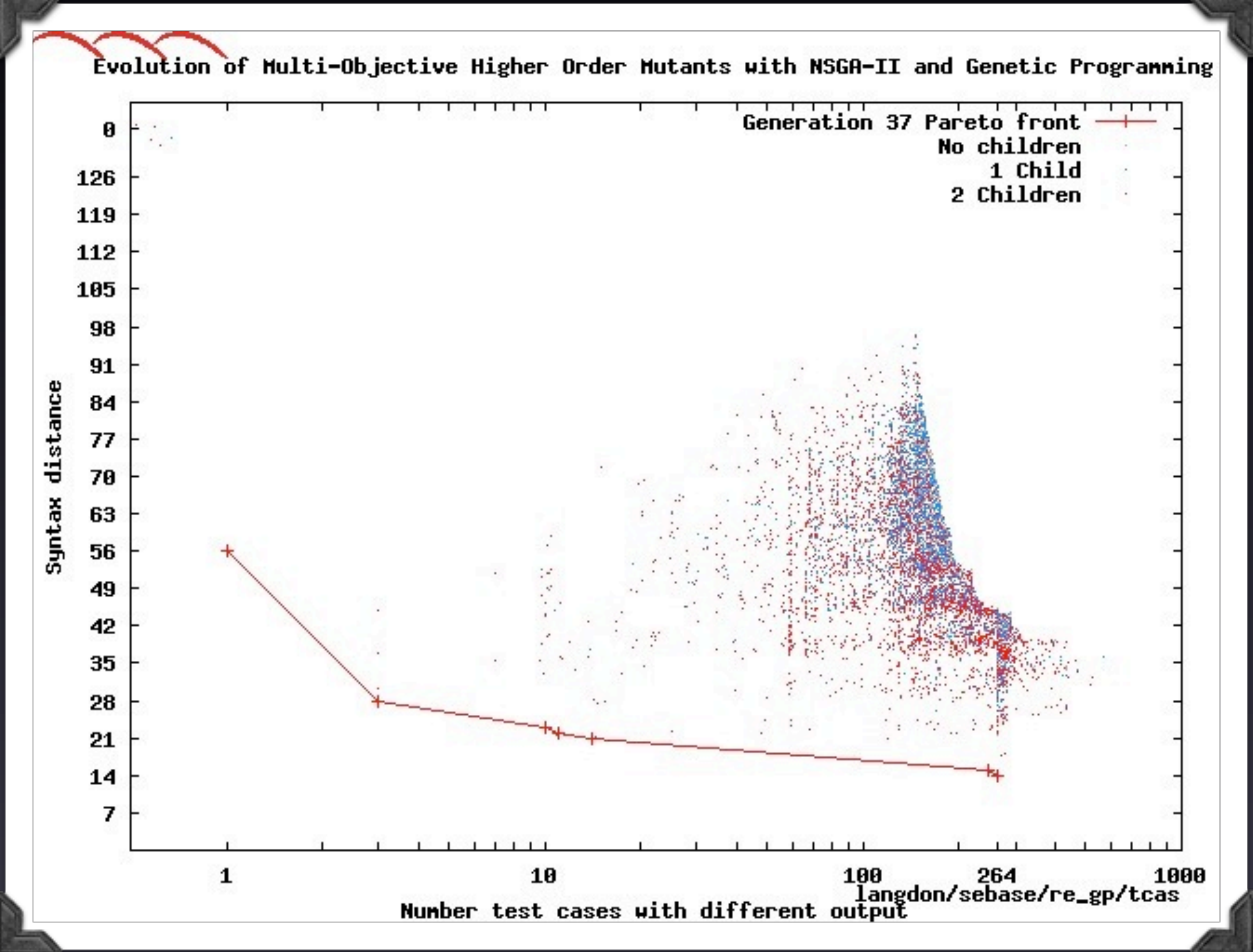


Overview FOM Restriction HOM Testing **Approaches** Classification Future Work

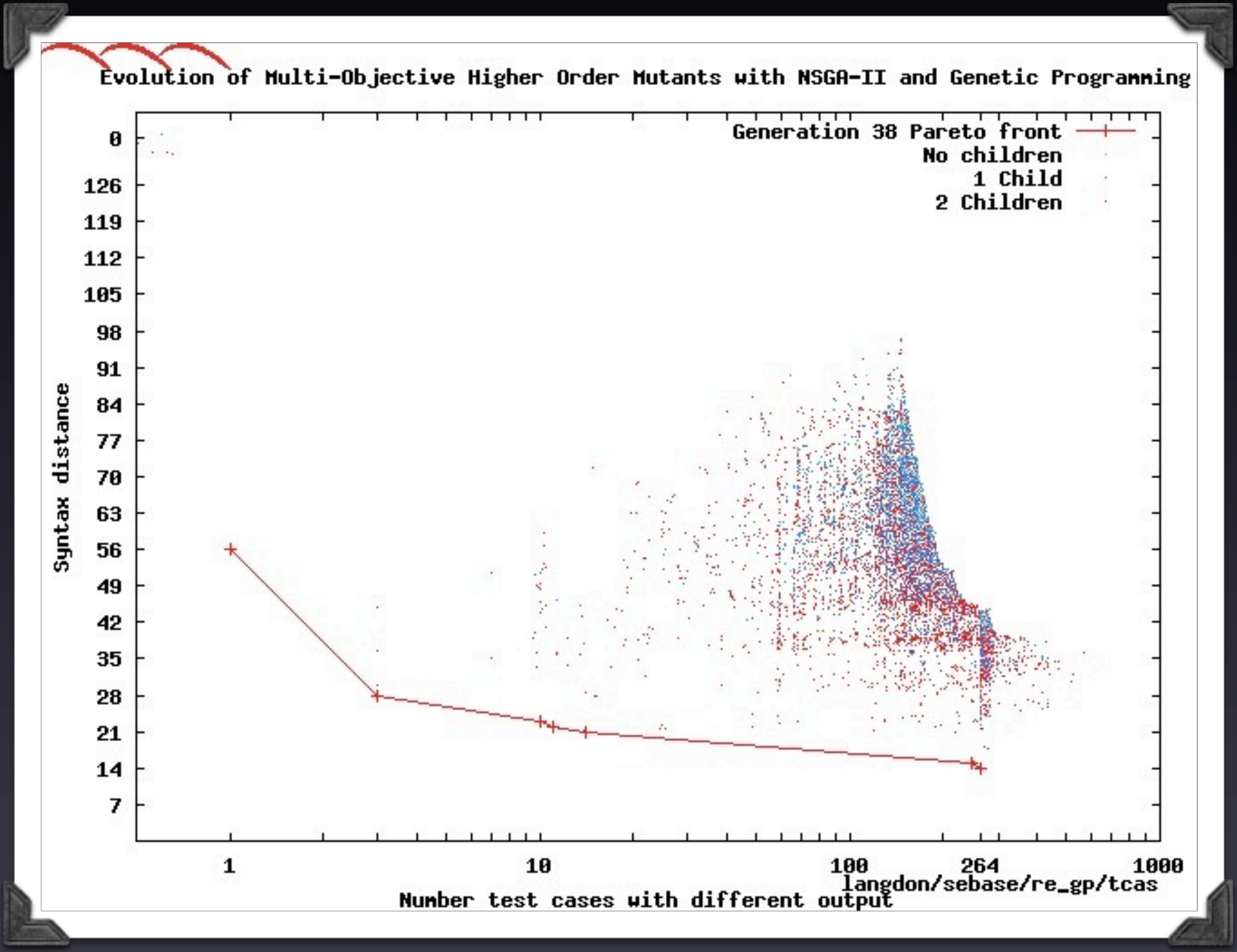
Multi Objective



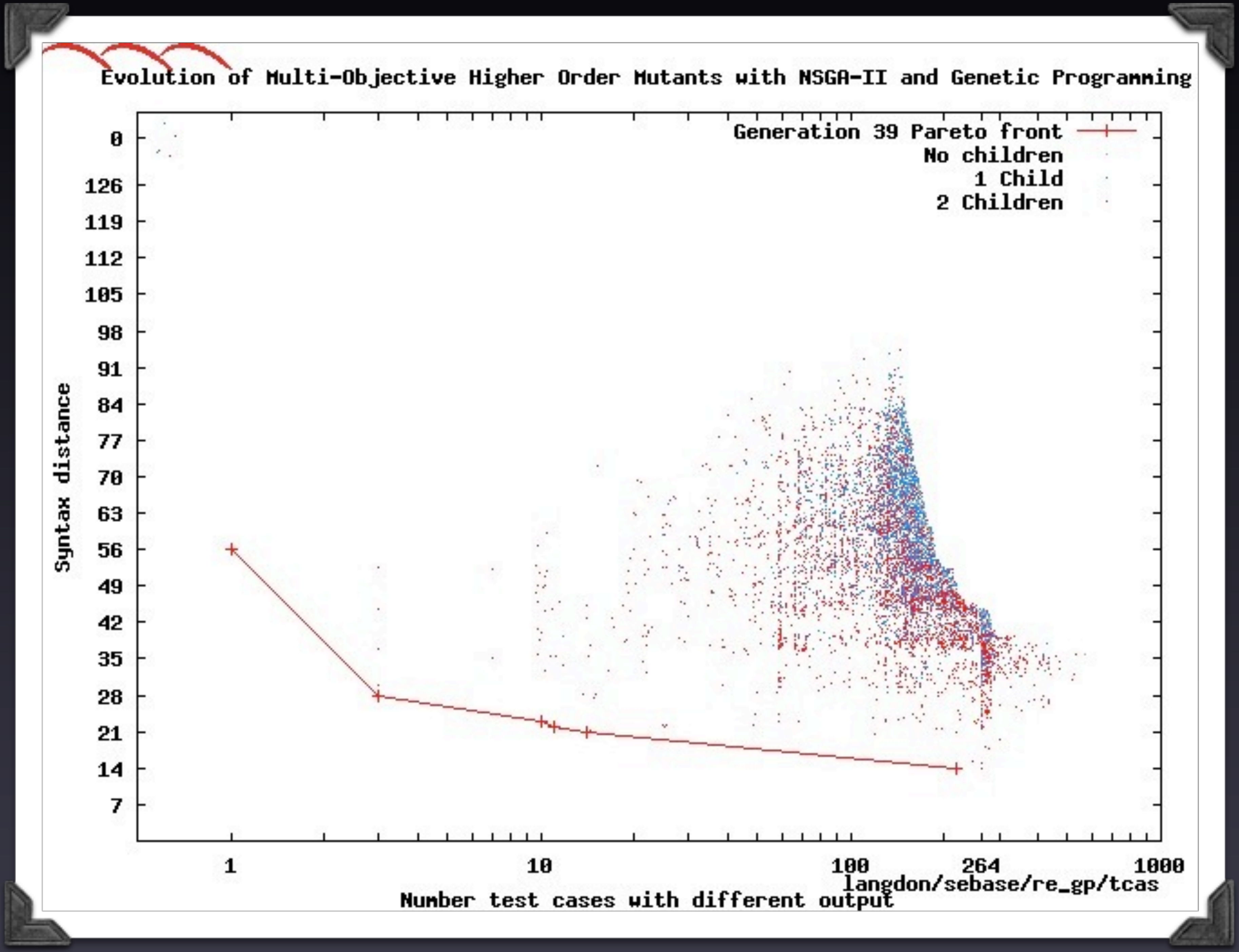
Multi Objective



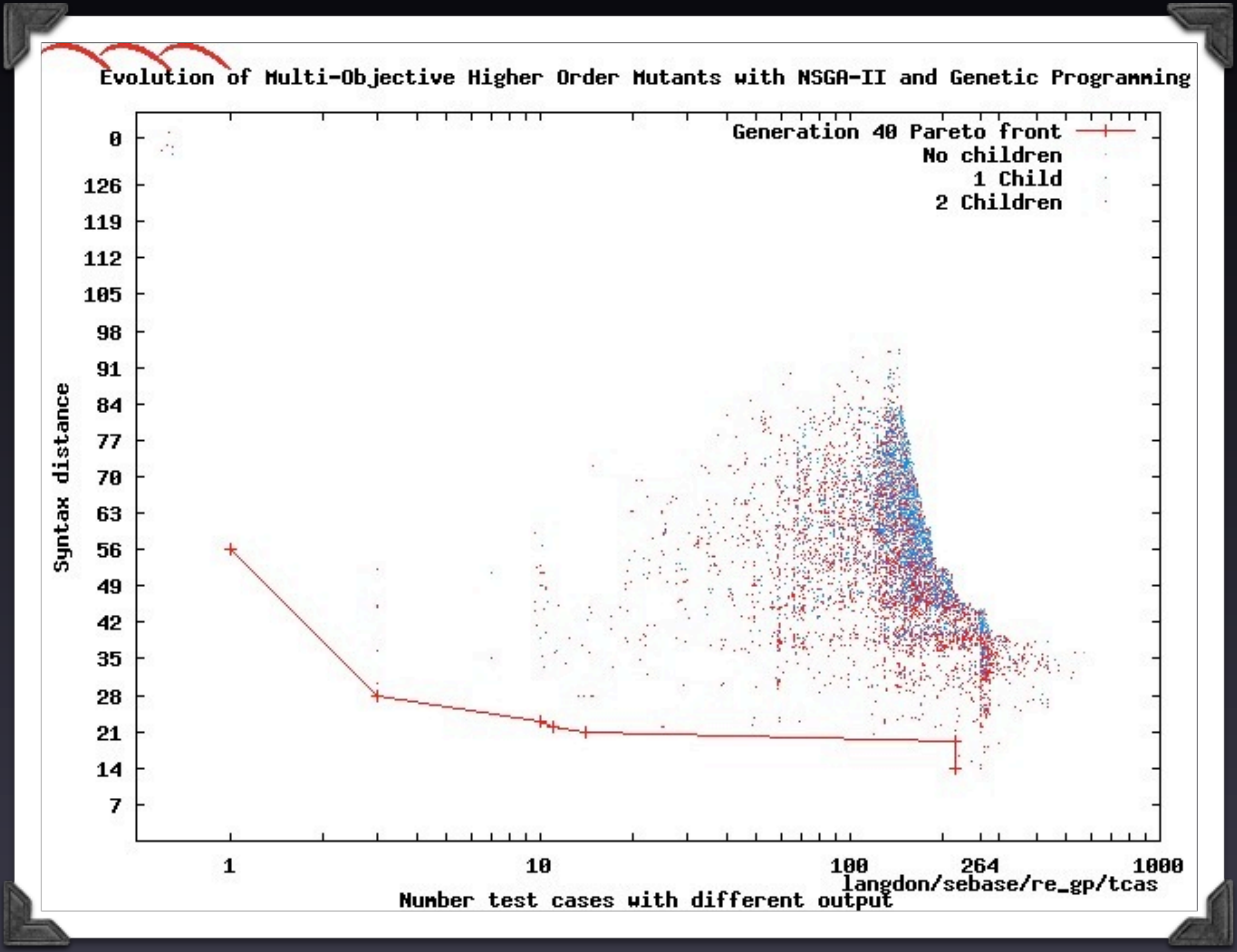
Multi Objective



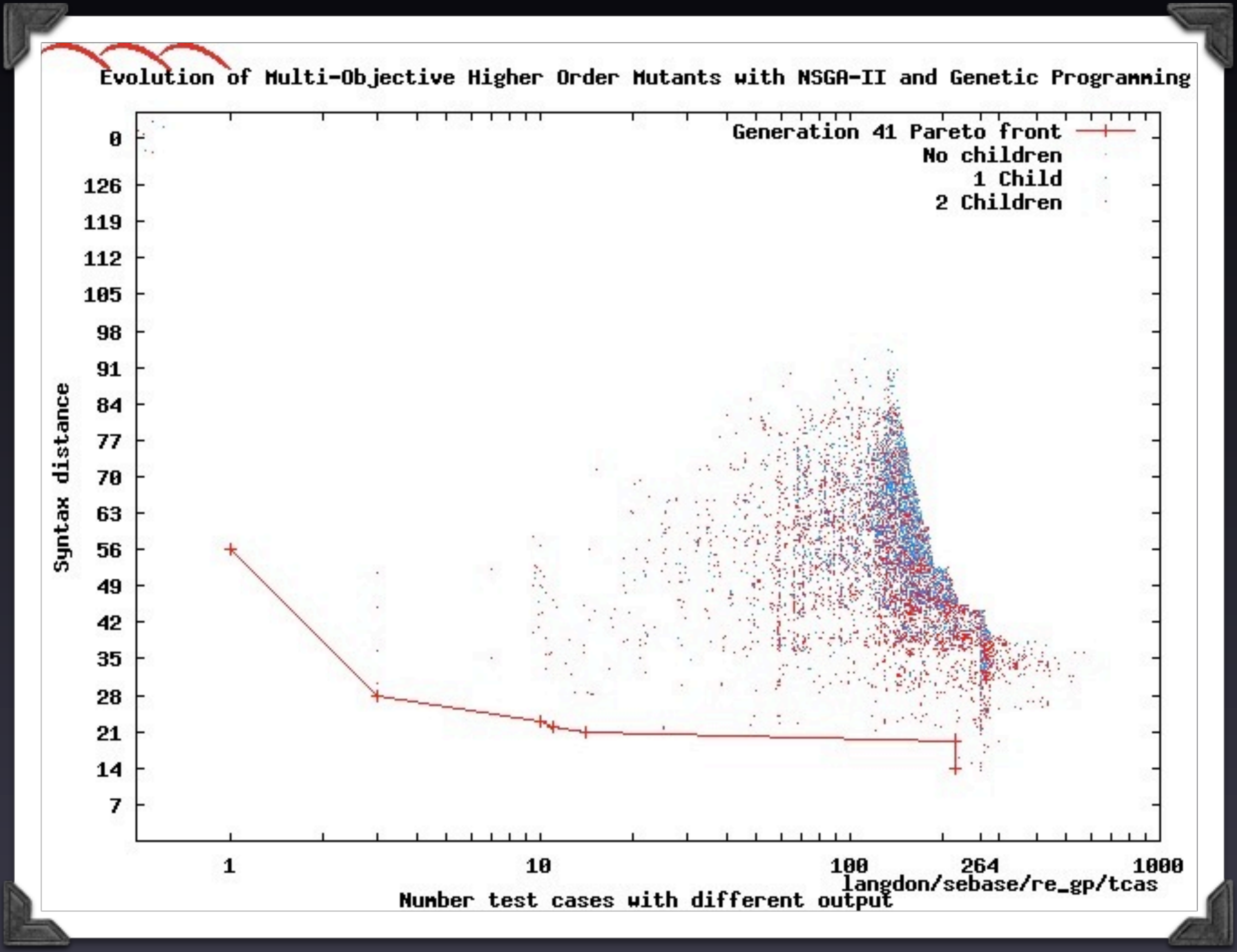
Multi Objective



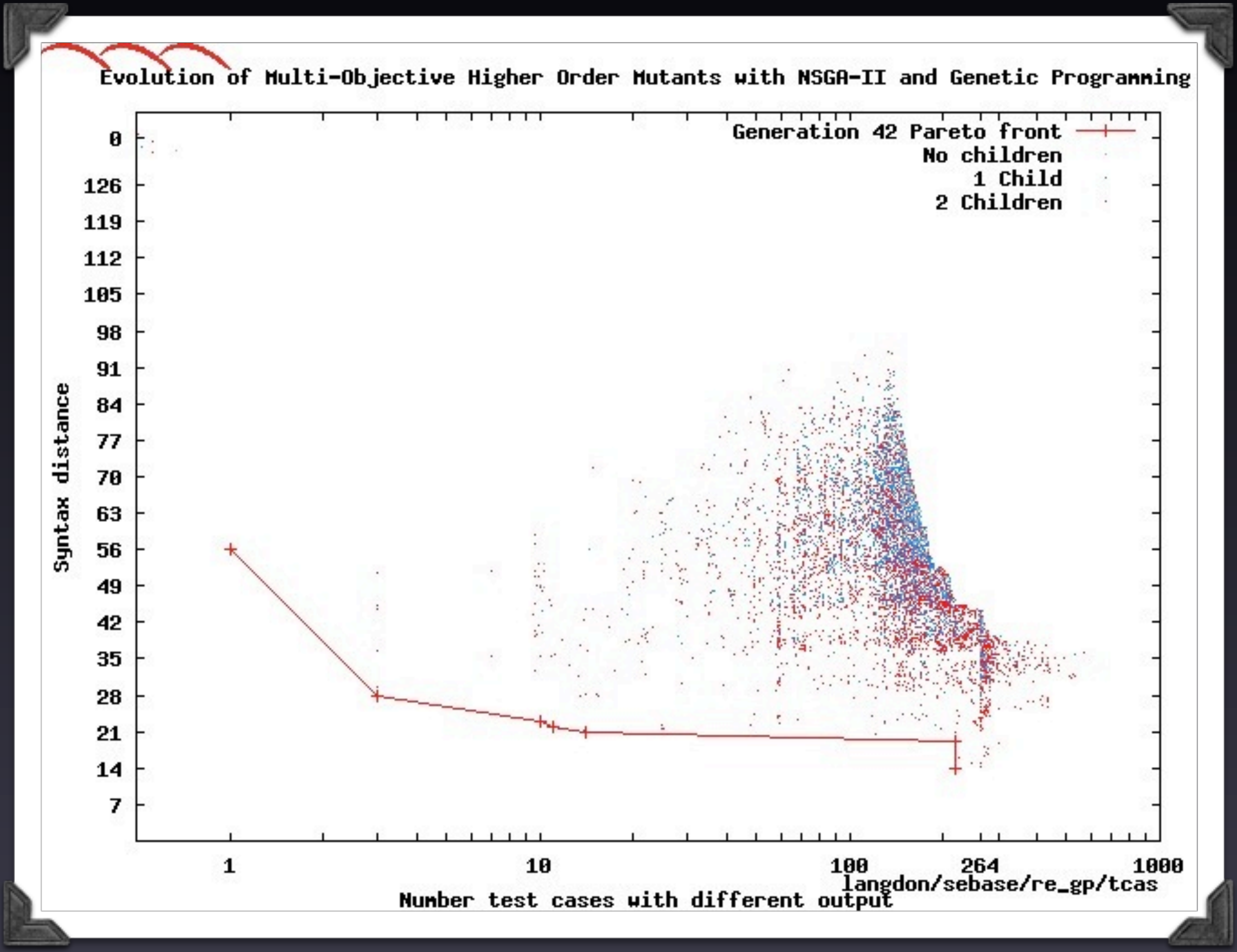
Multi Objective



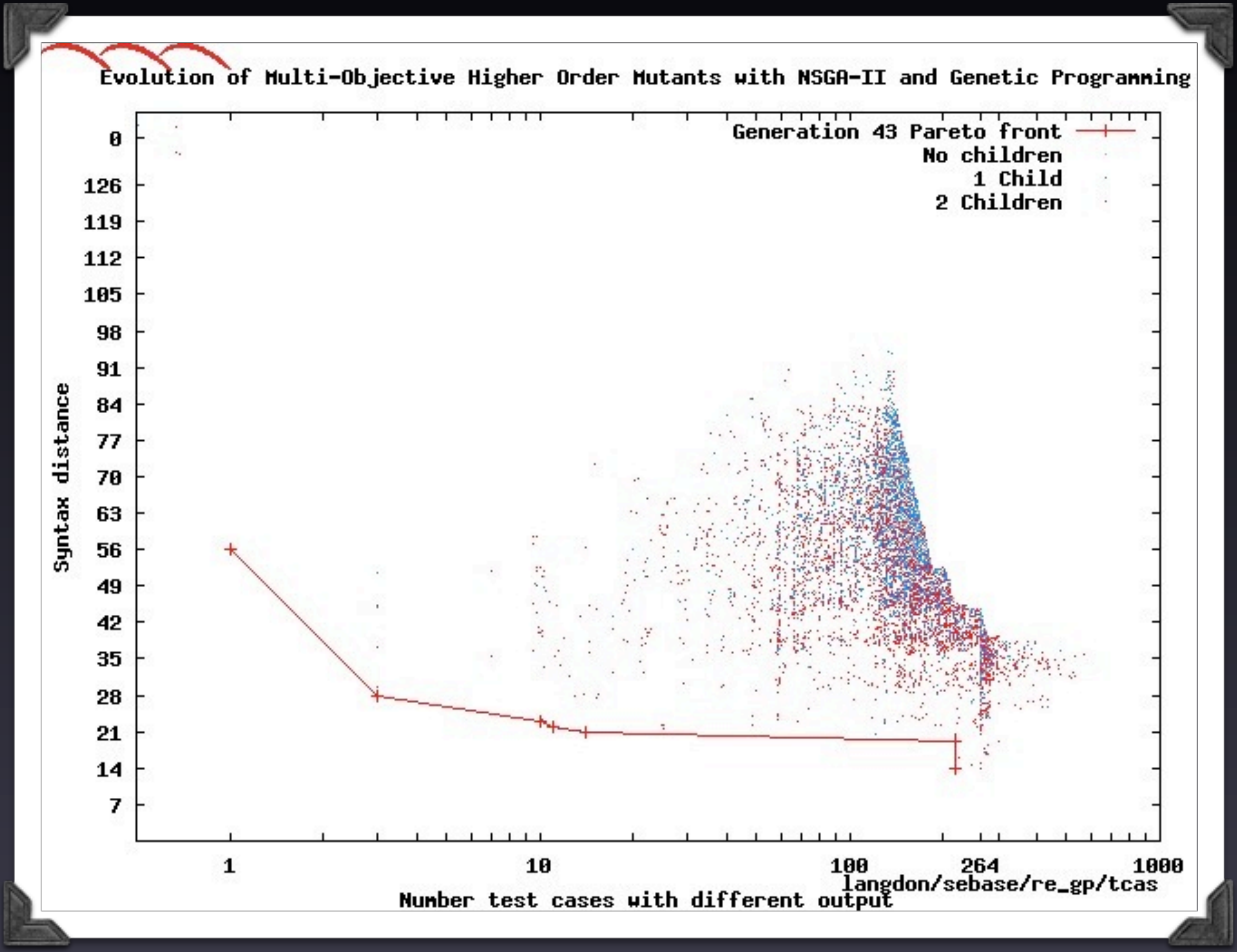
Multi Objective



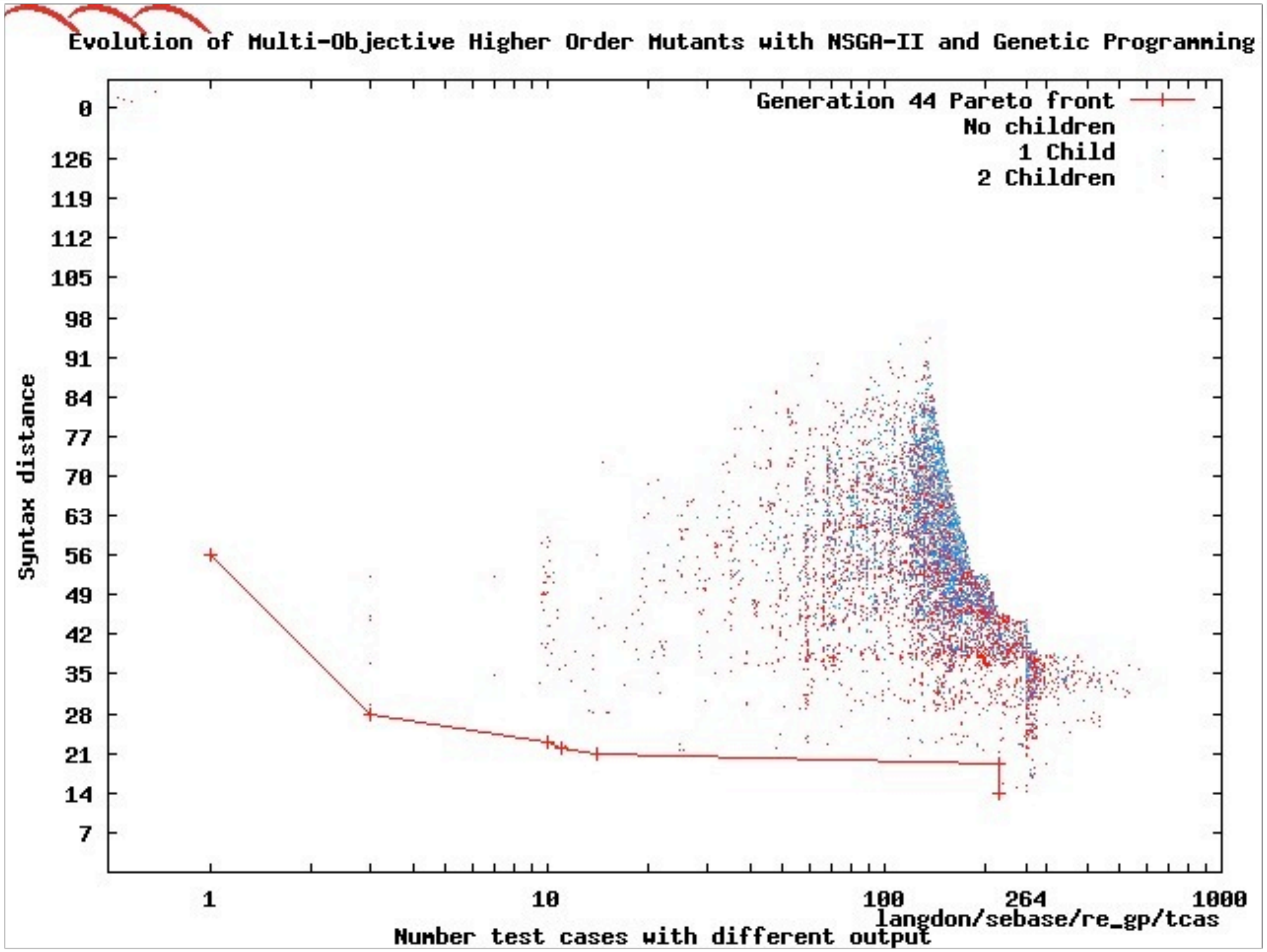
Multi Objective



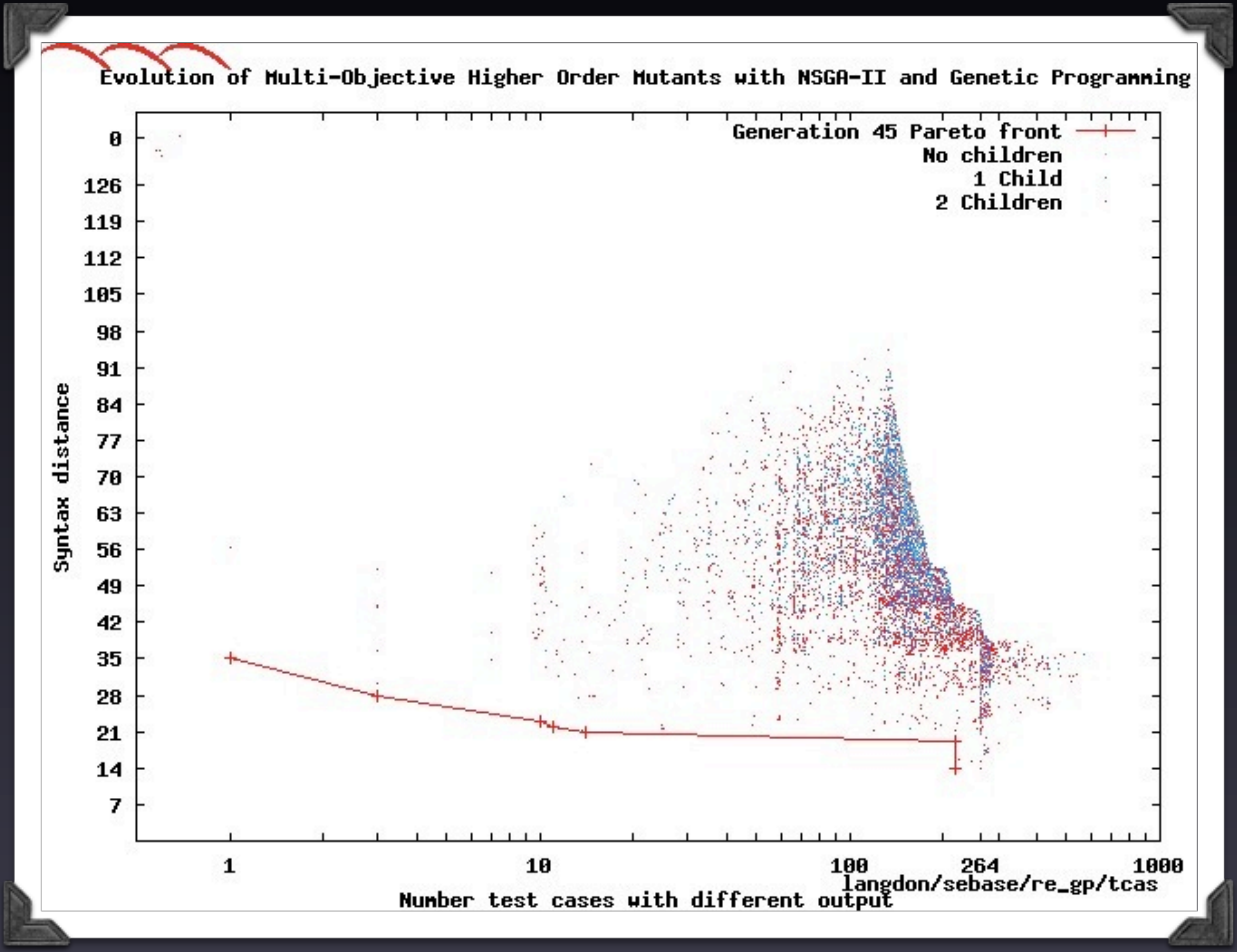
Multi Objective



Multi Objective

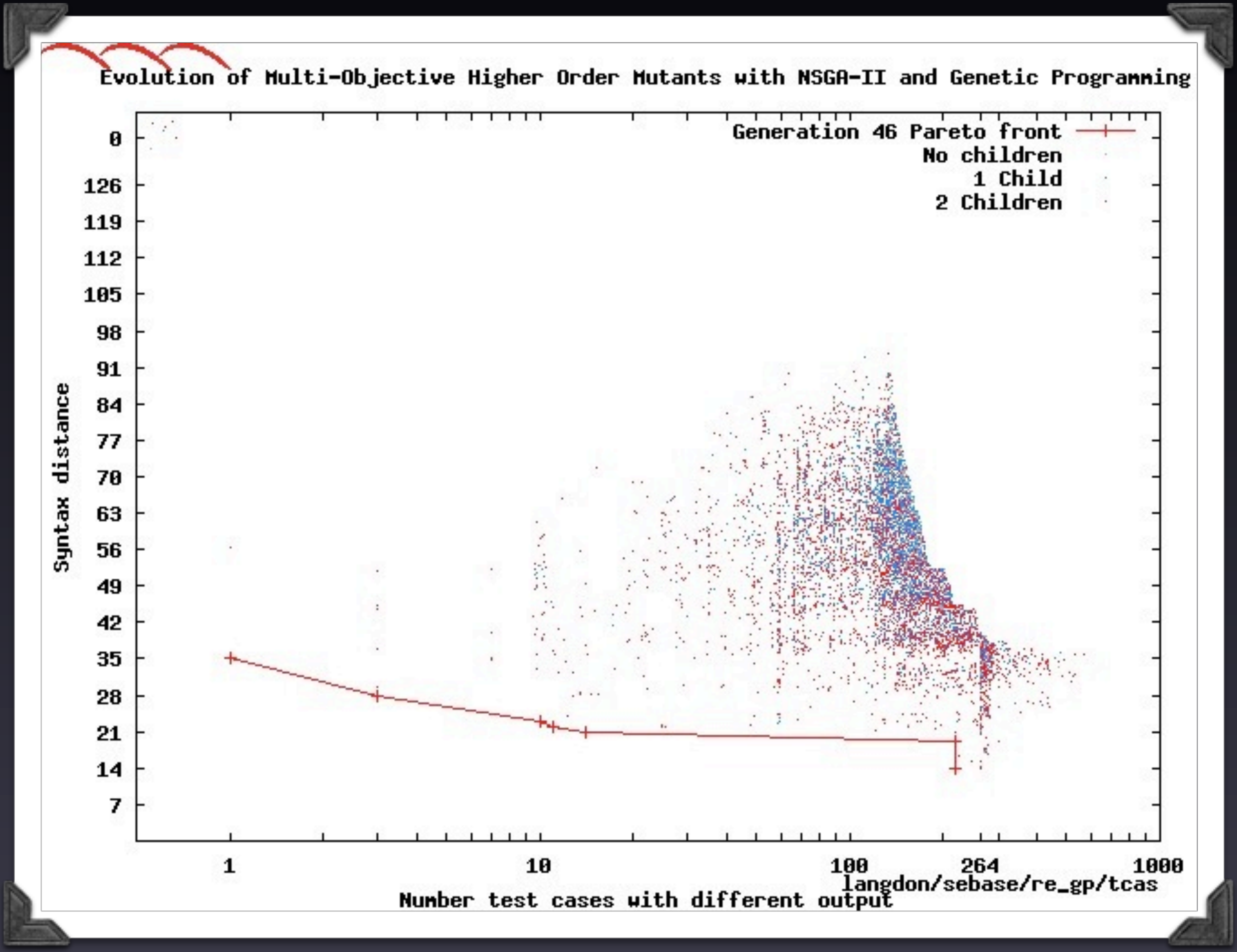


Multi Objective

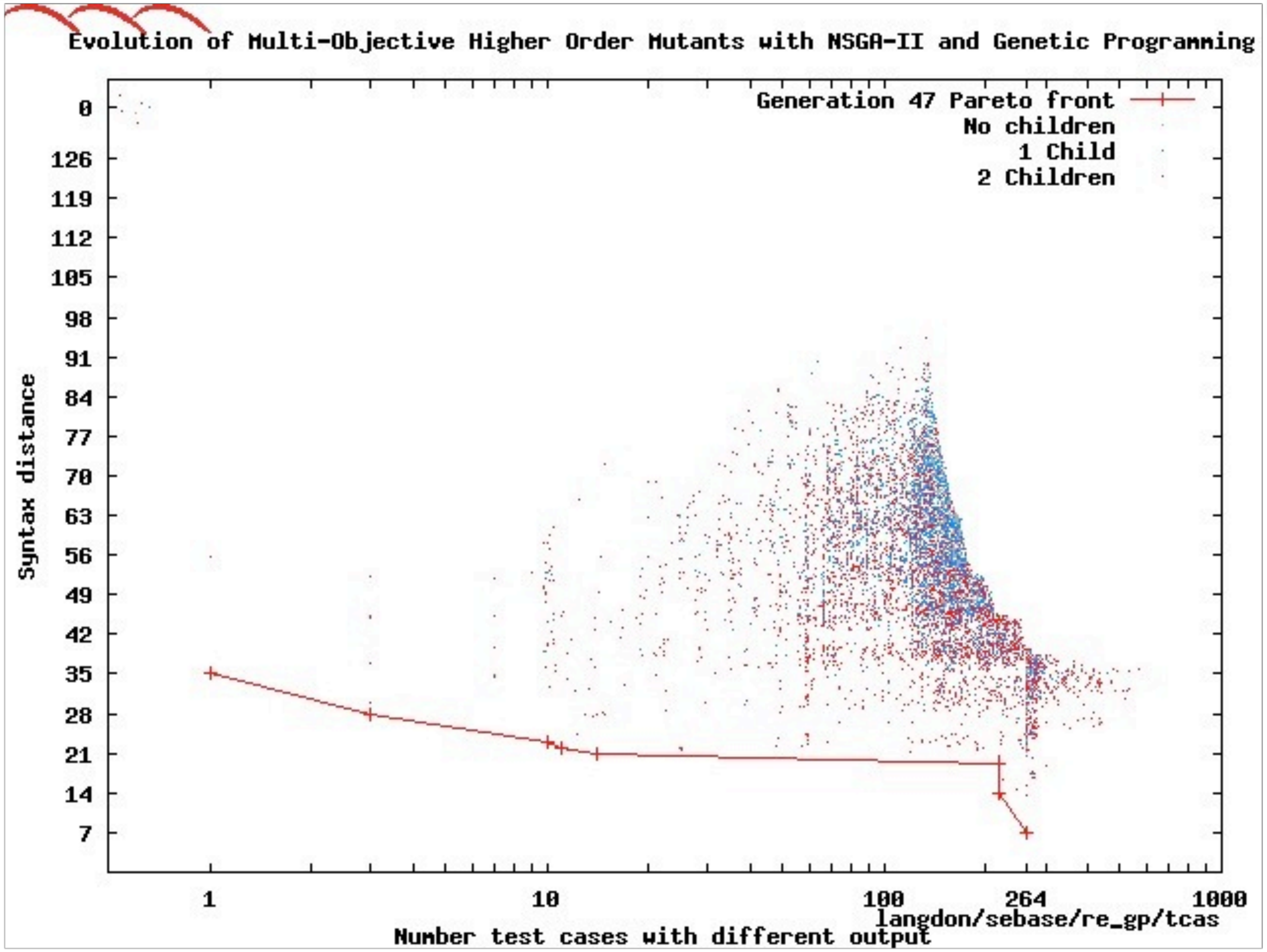


Overview FOM Restriction HOM Testing **Approaches** Classification Future Work

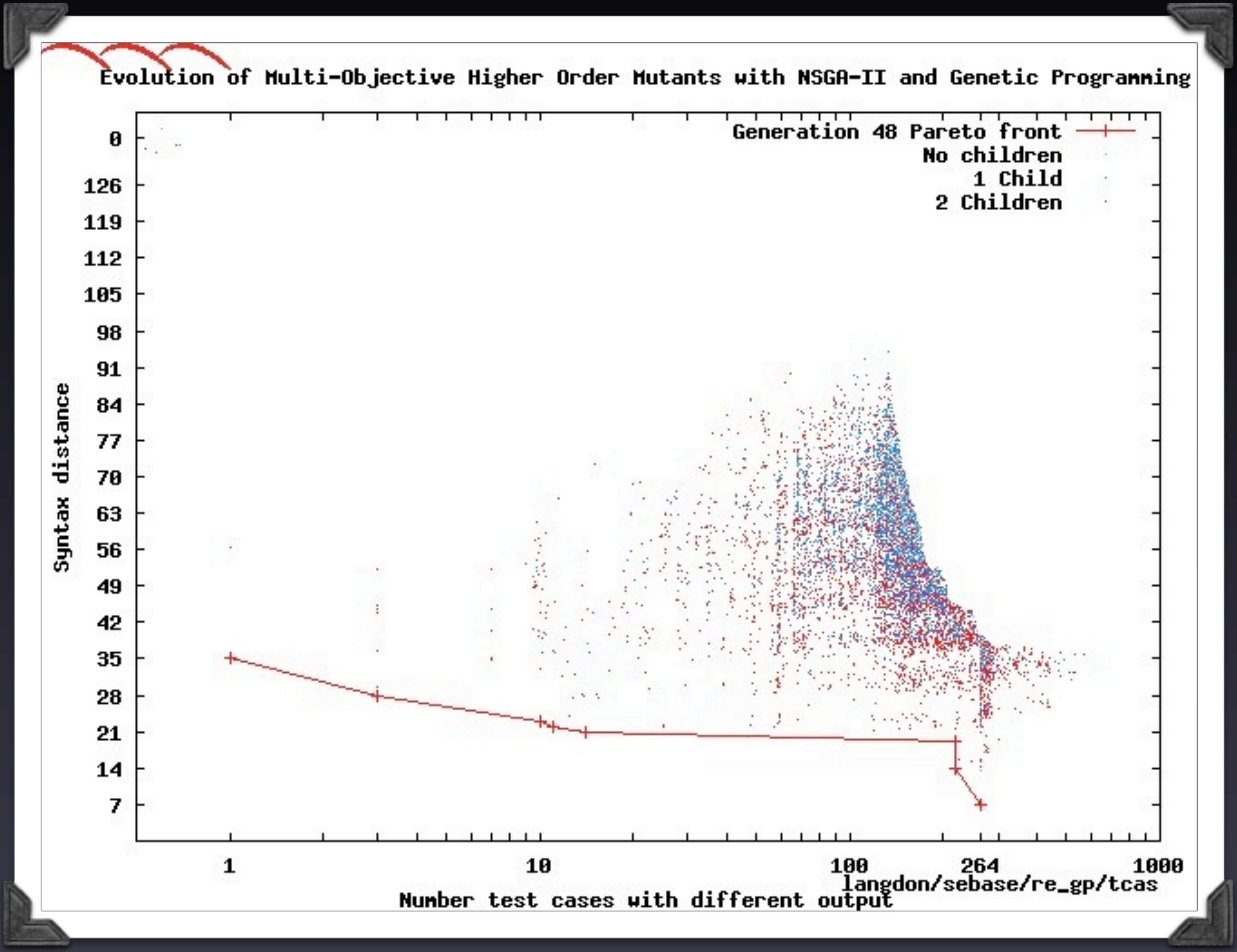
Multi Objective



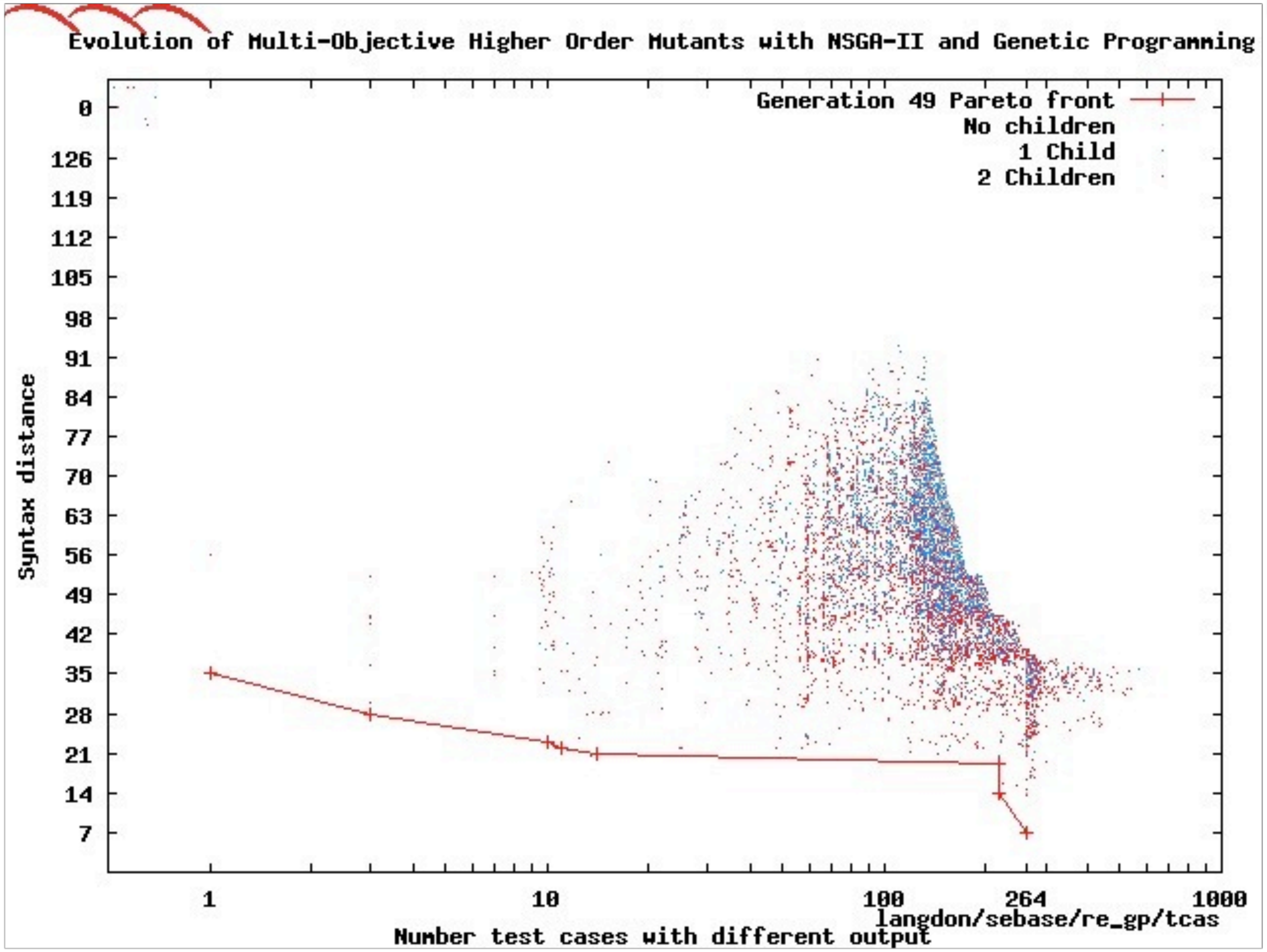
Multi Objective



Multi Objective

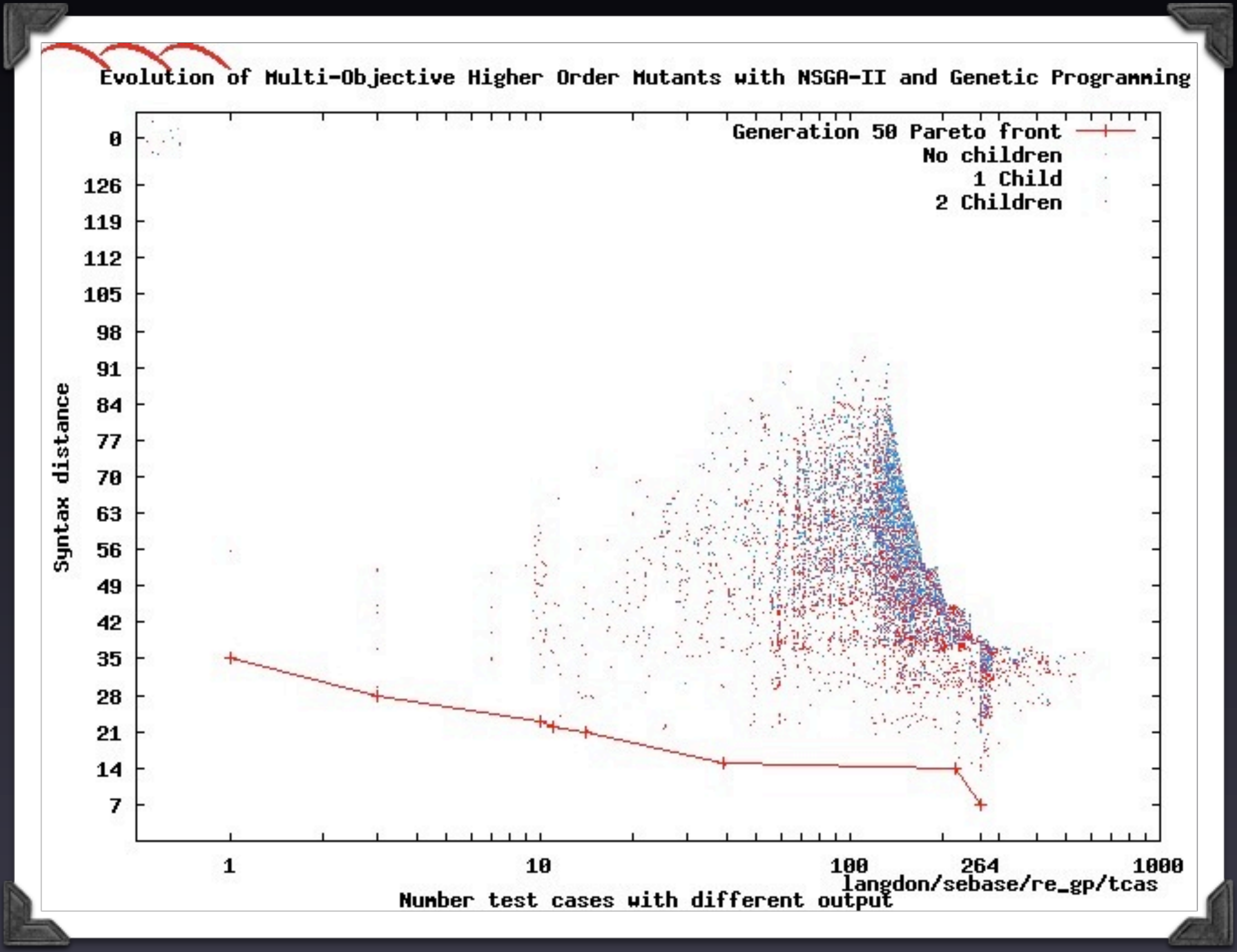


Multi Objective



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

Multi Objective



HOM Classification

HOM classification

Most common case

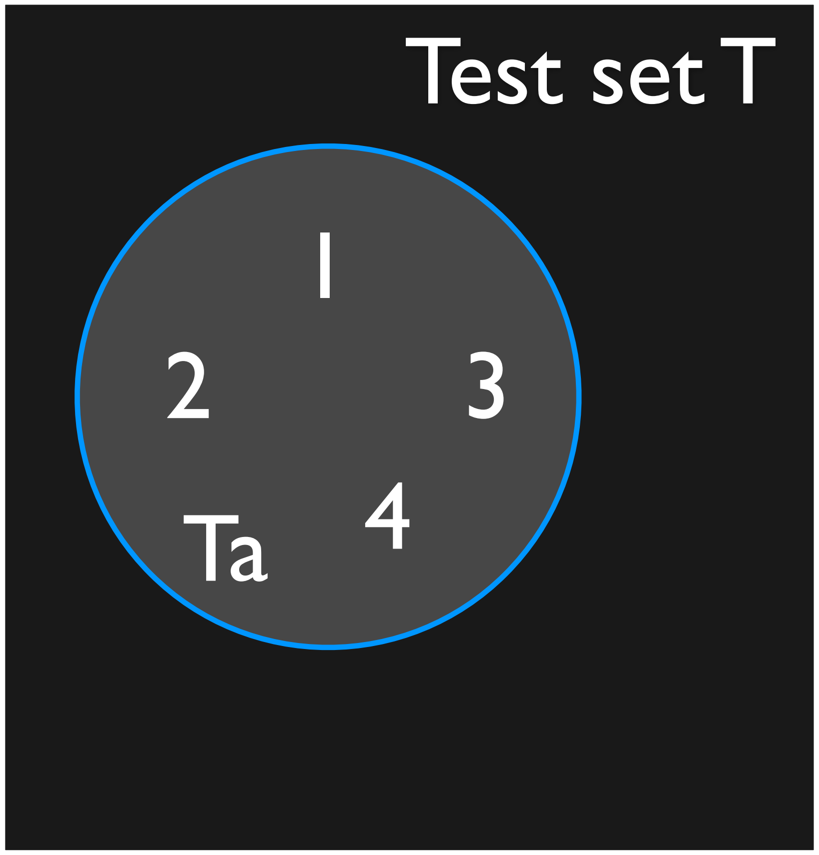
FOM a is killed by
{ 1, 2, 3, 4 }

Test set T

HOM classification

Most common case

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

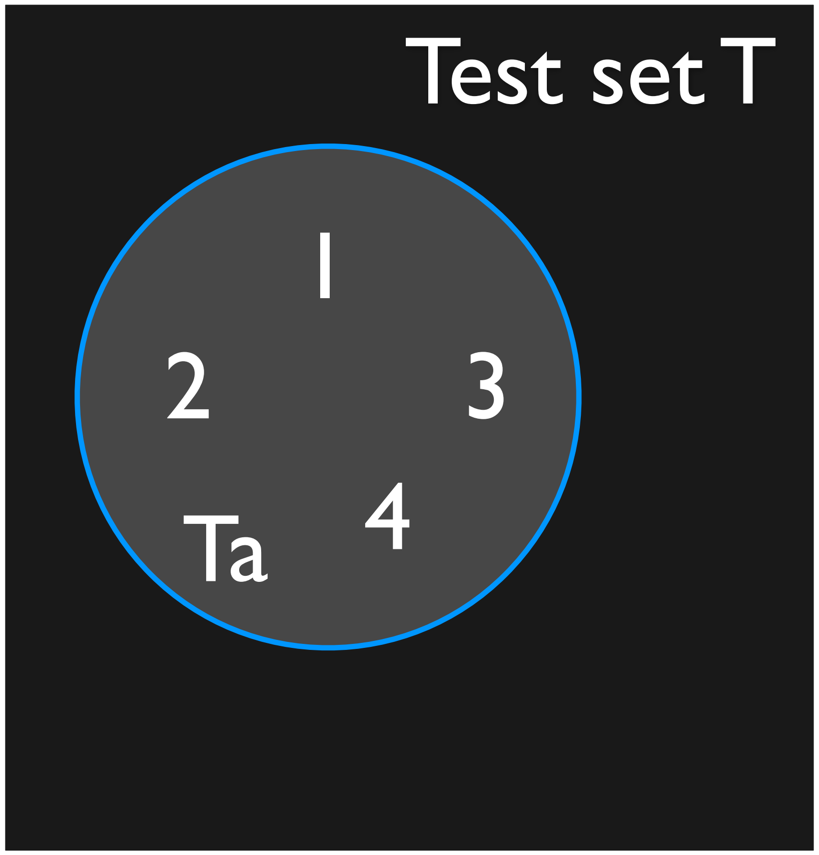


HOM classification

Most common case

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

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

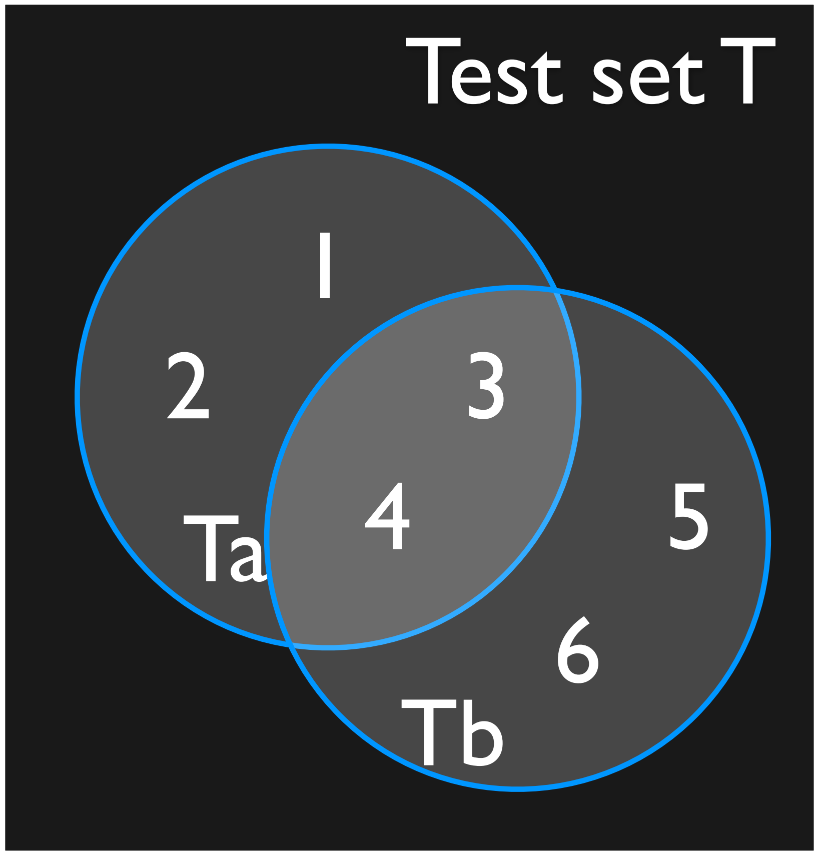


HOM classification

Most common case

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

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



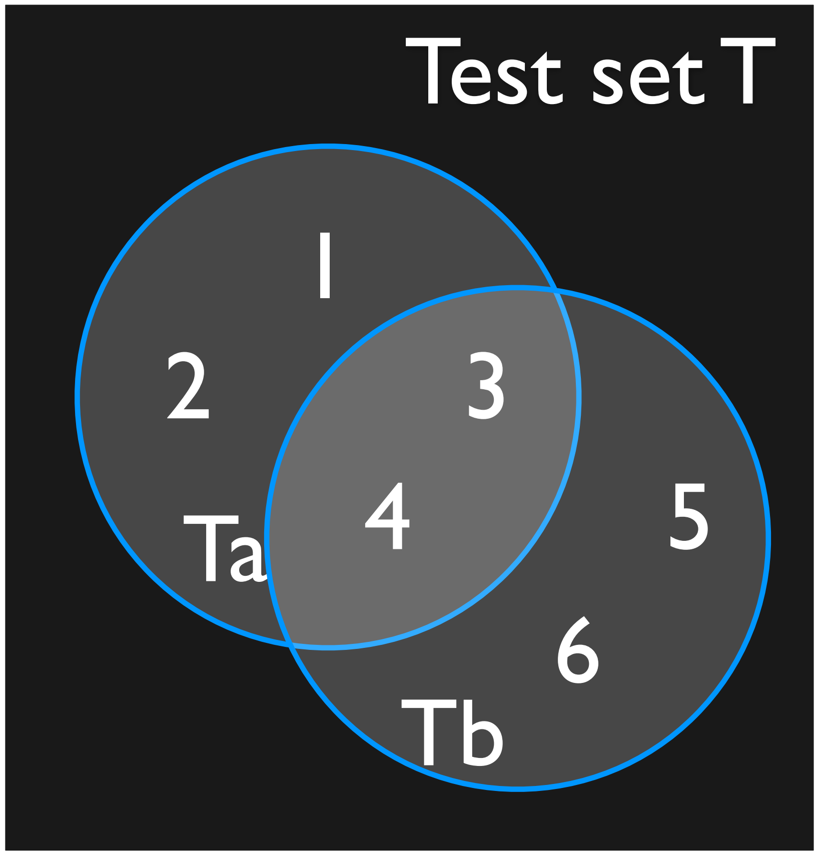
HOM classification

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 }



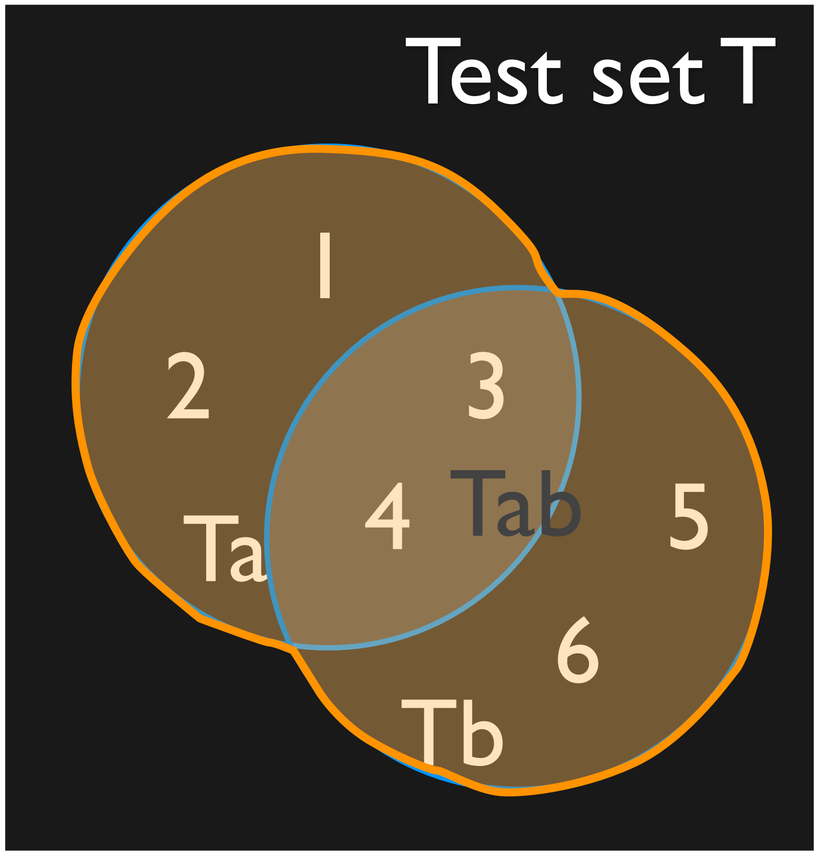
HOM classification

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 }



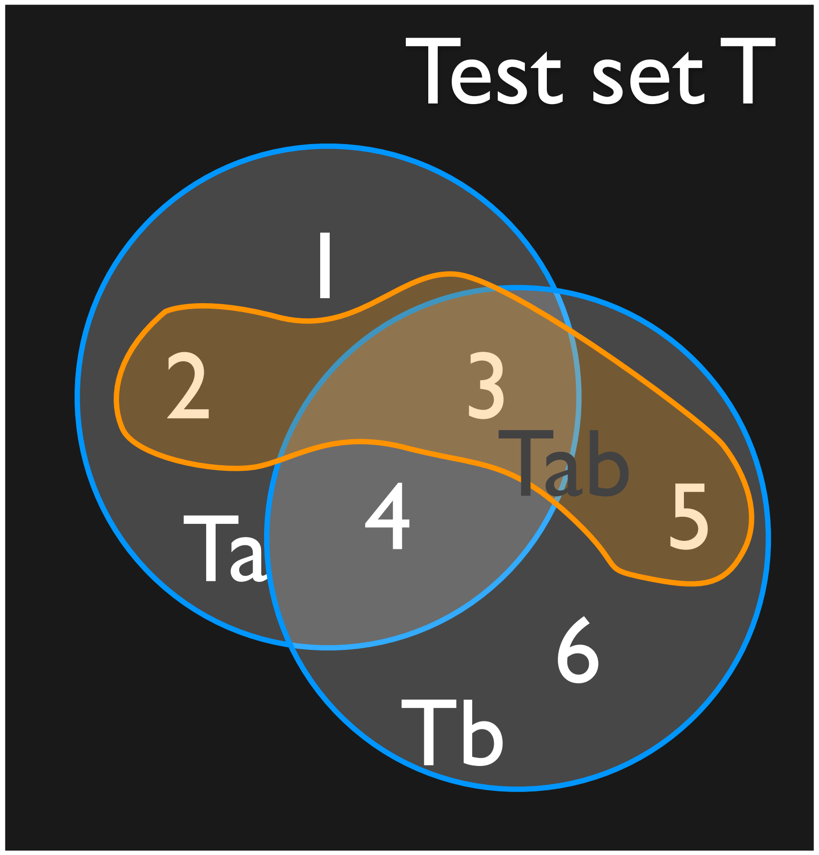
HOM classification

Most common case

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

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

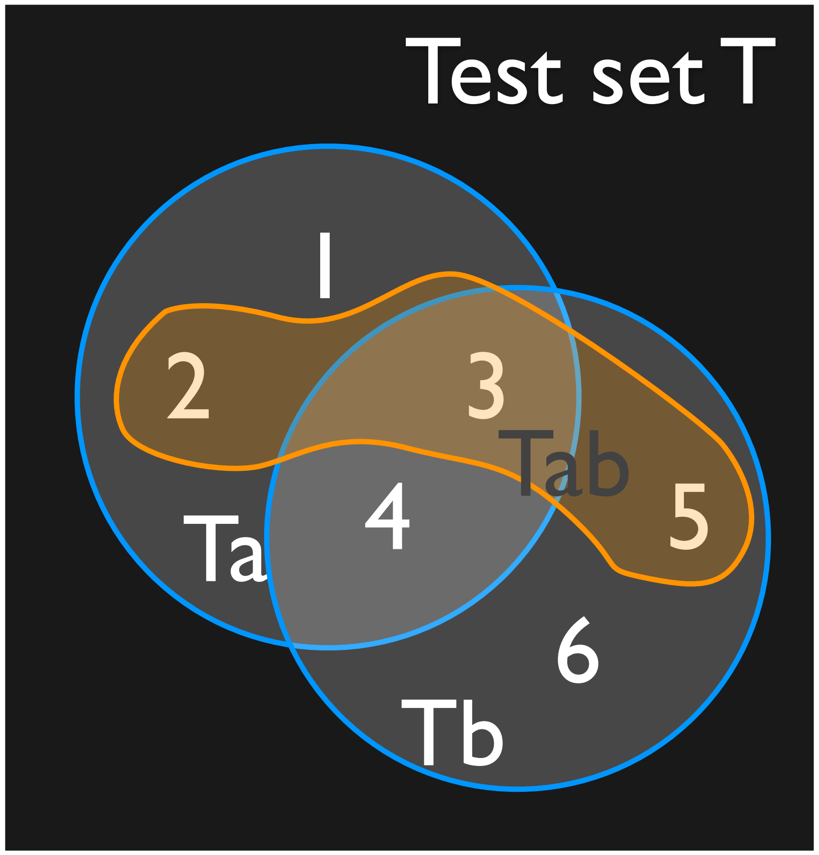
HOM ab is killed by
{ 2, 3, 5 }



HOM classification

Many types of HOM

Subsuming HOM

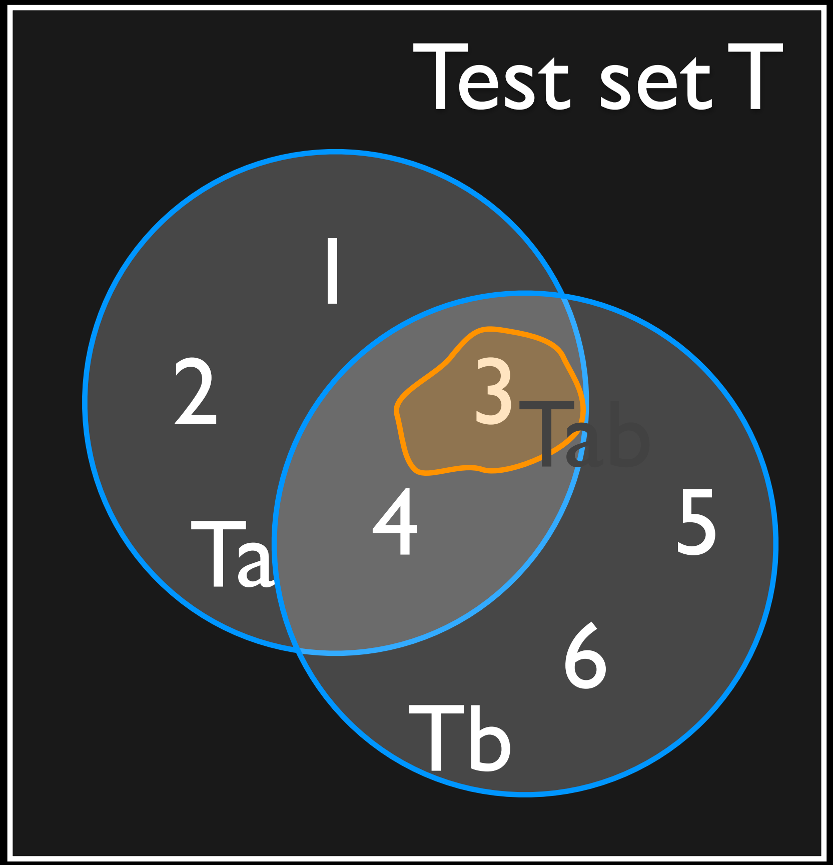


HOM classification

Many types of HOM

Subsuming HOM

Strongly Subsuming



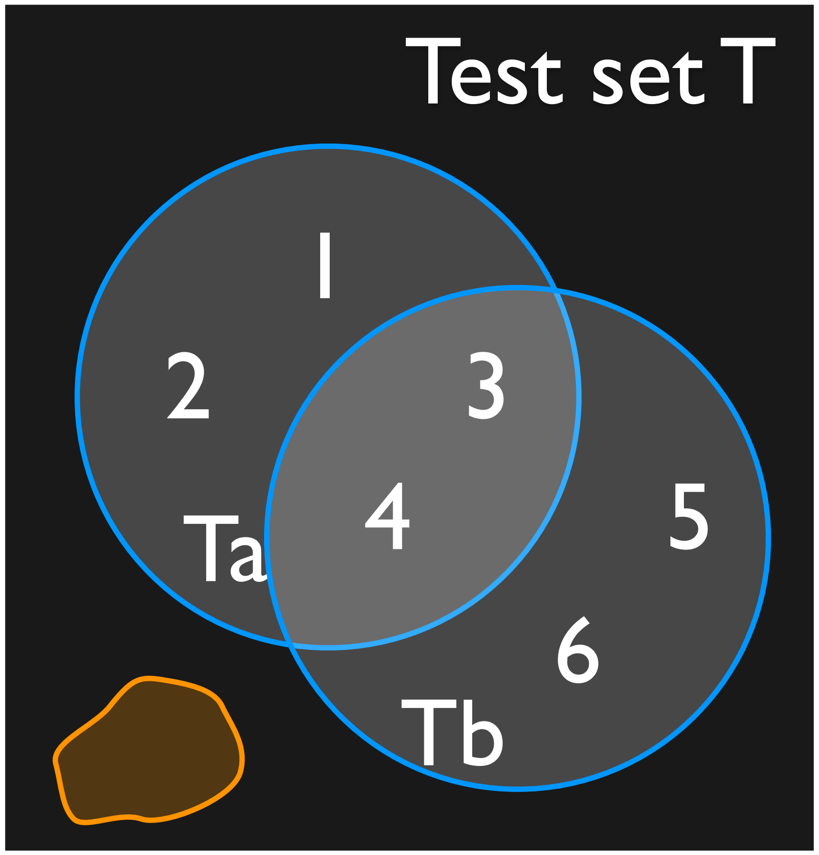
HOM classification

Many types of HOM

Subsuming HOM

Strongly Subsuming

Anti-Coupling Effect



HOM classification

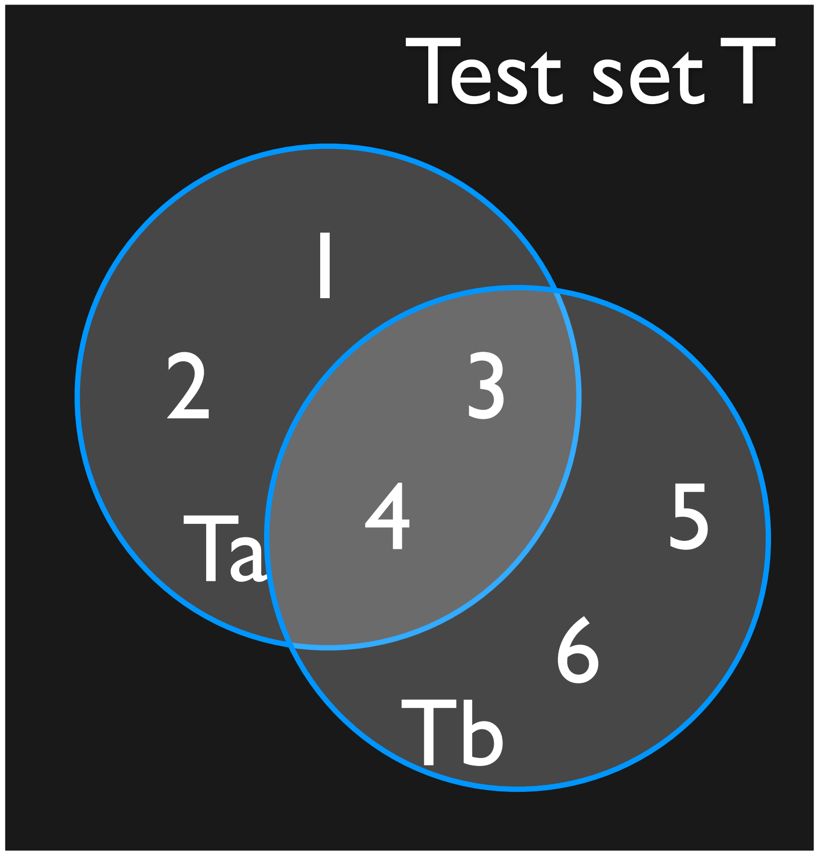
Many types of HOM

Subsuming HOM

Strongly Subsuming

Anti-Coupling Effect

Equivalent

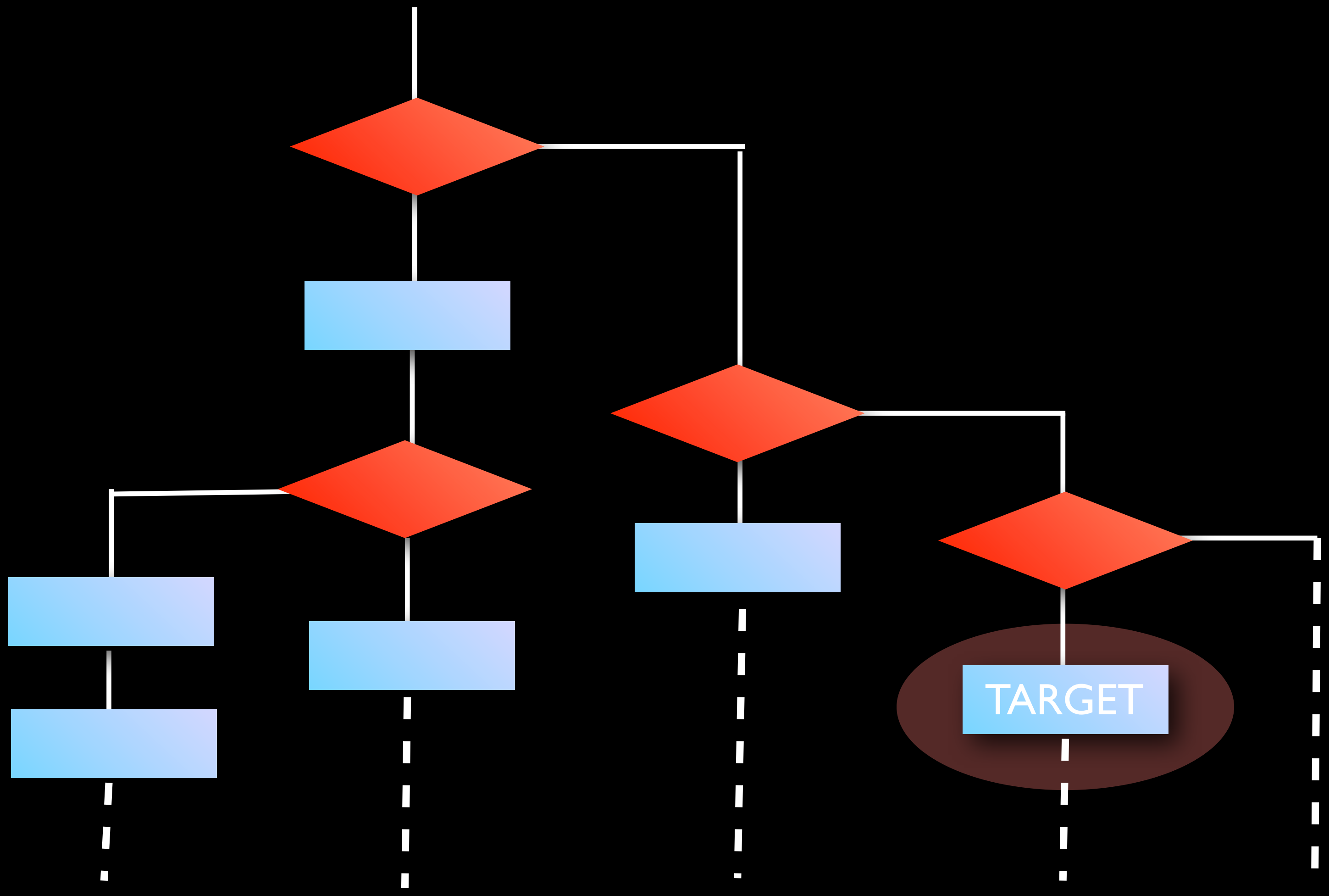


Single Objective Results

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

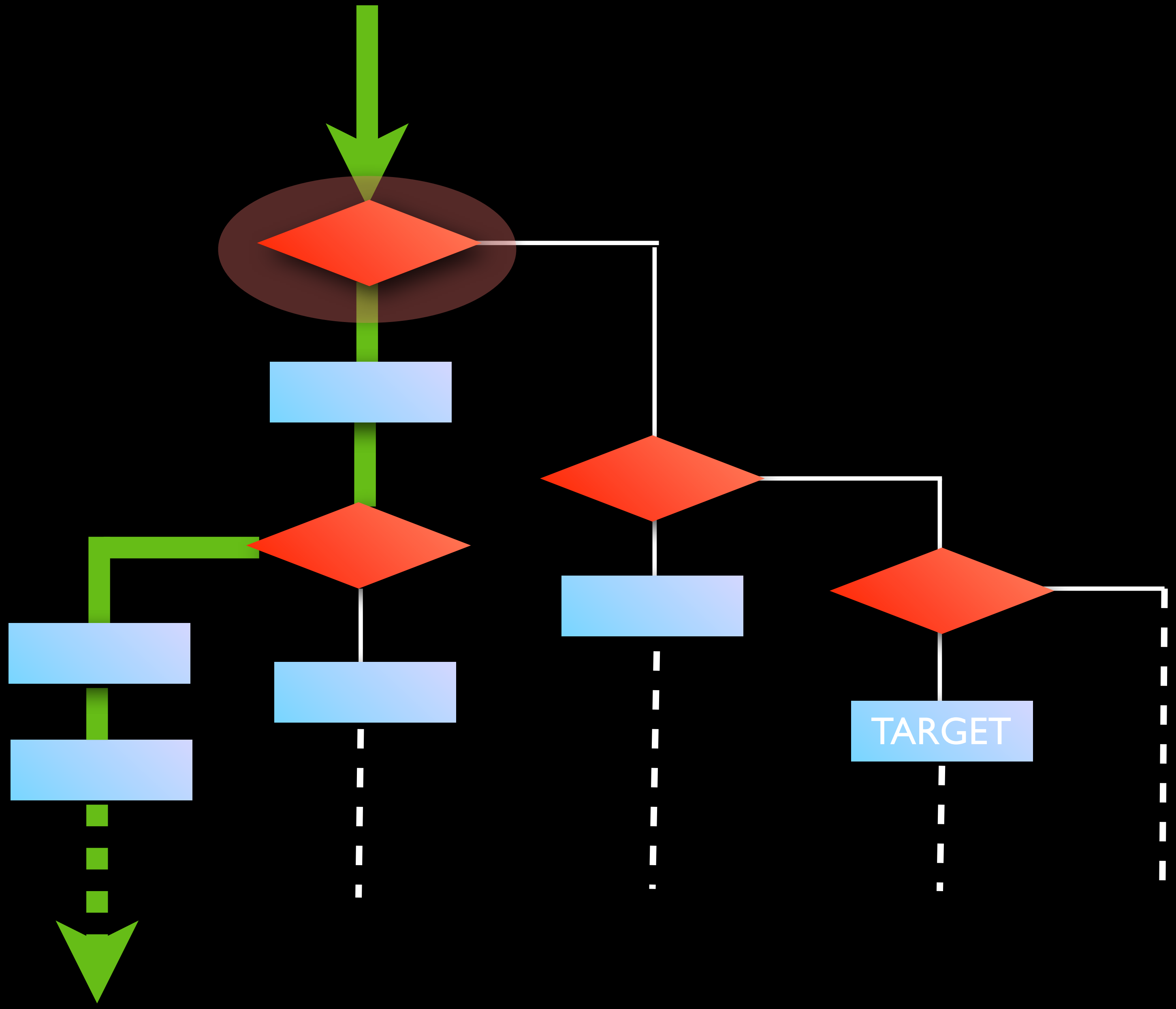
Optimising
Structural Testing

Program Coverage

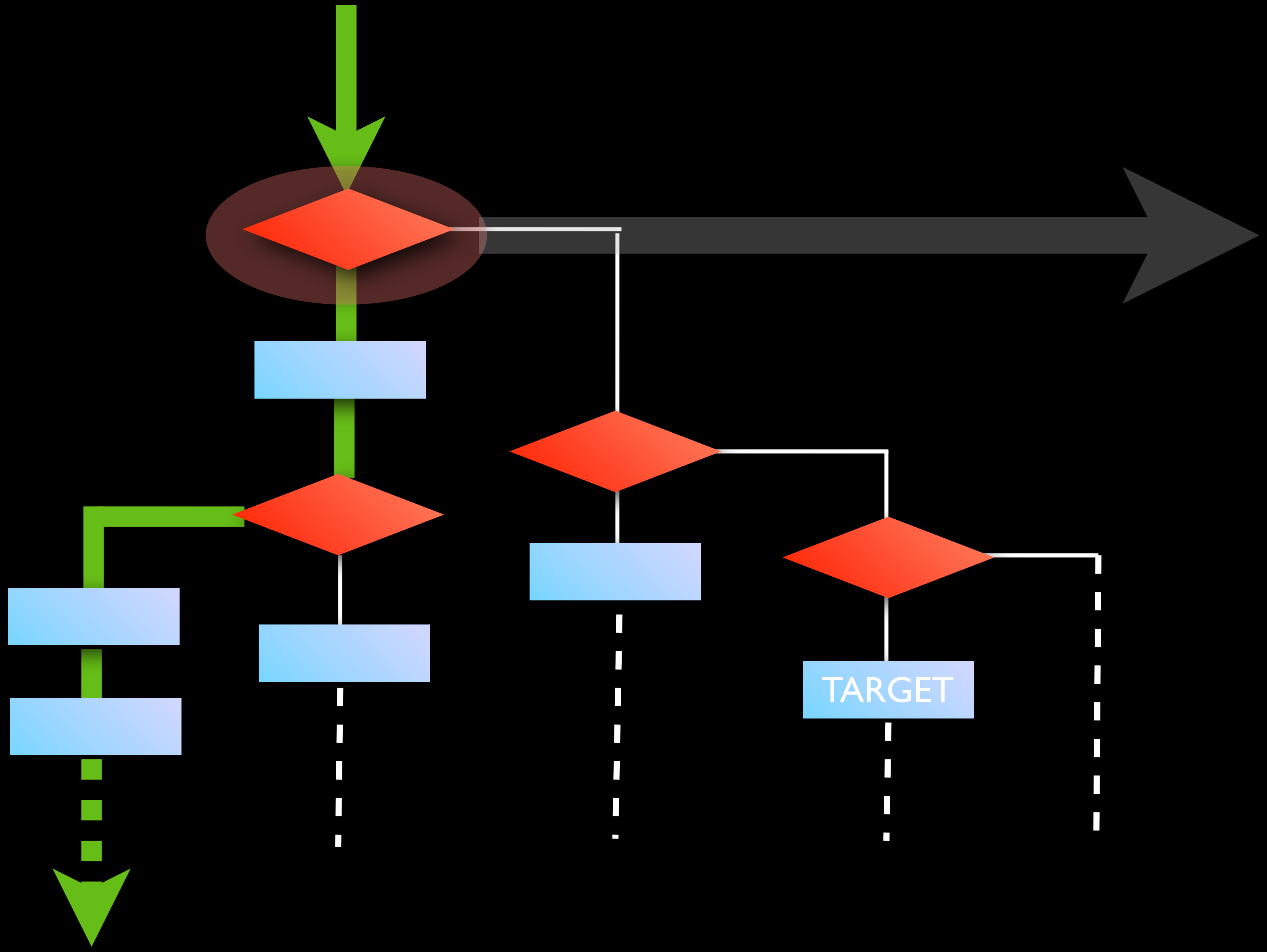


control flow graph of a program
containing a structure we want to cover

Fitness evaluation

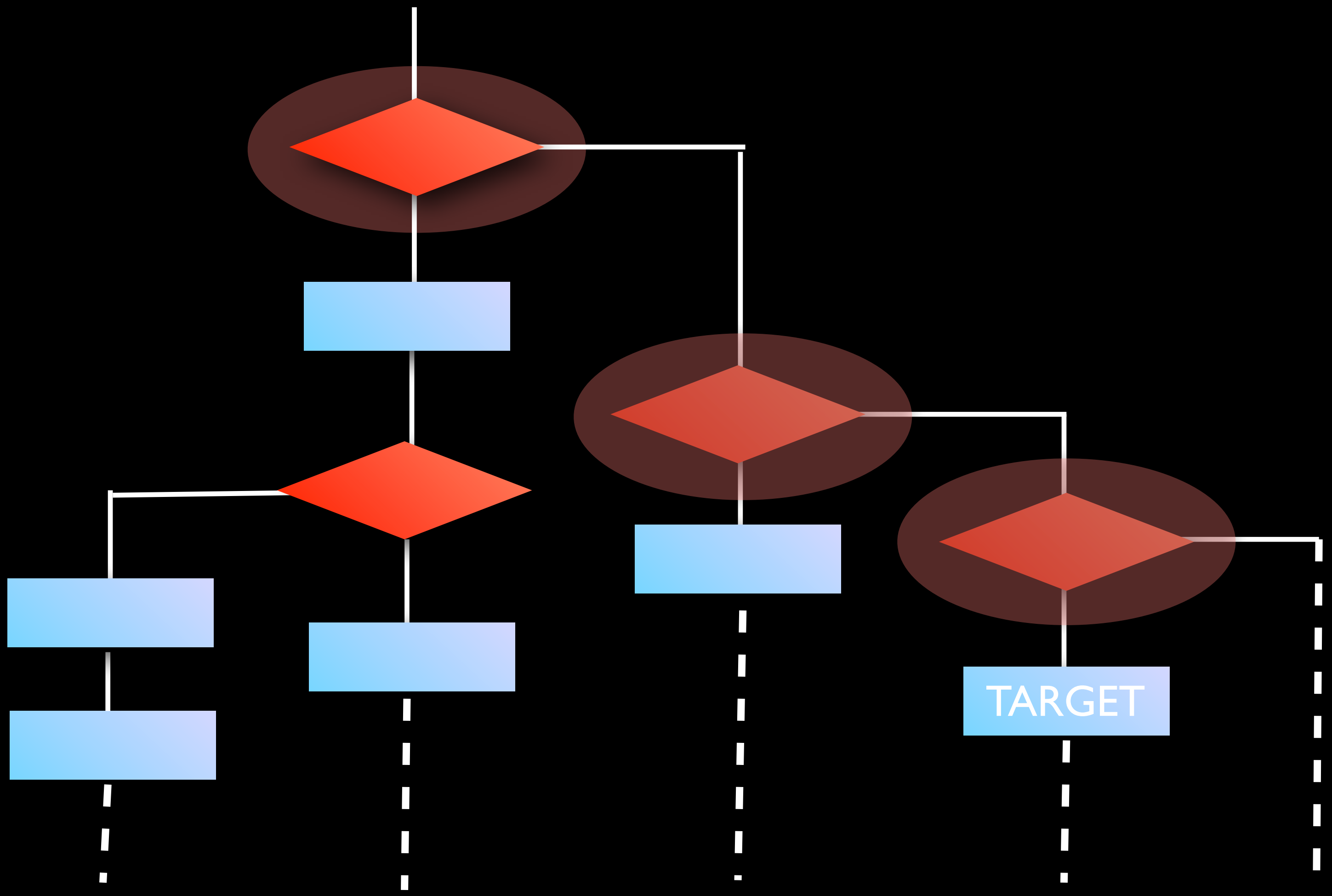


Fitness evaluation

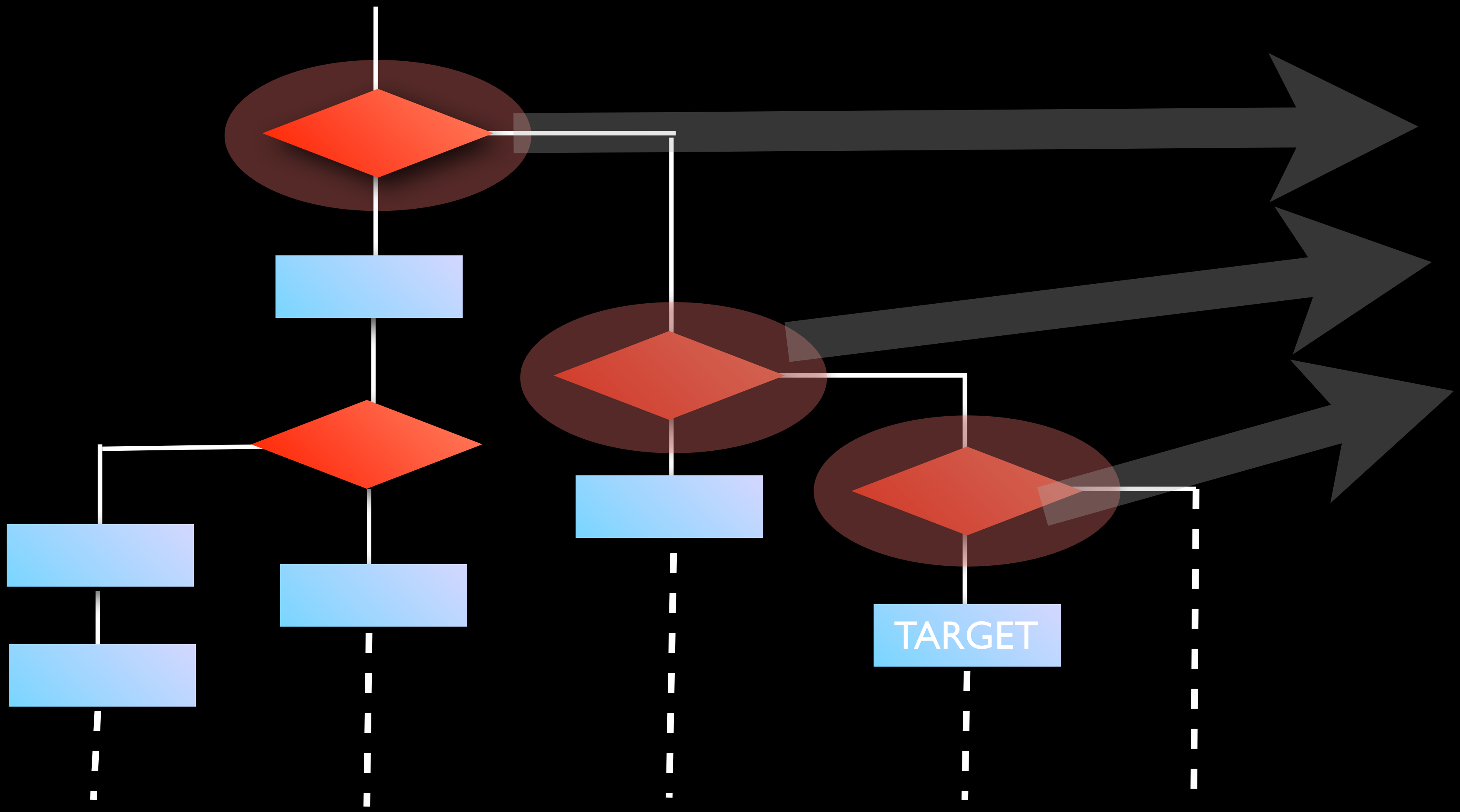


The test data executes the 'wrong' path

Analysing control flow



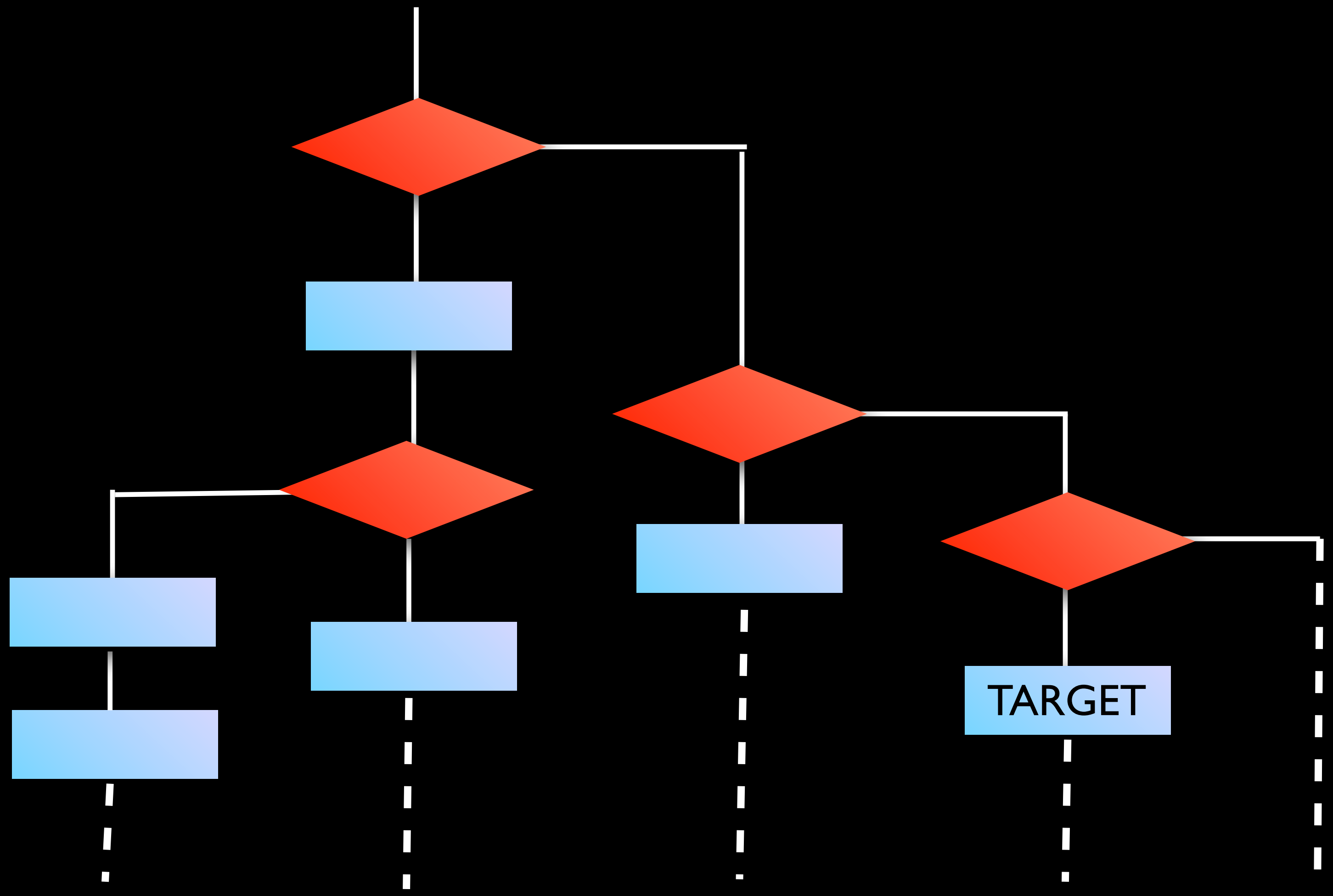
Analysing control flow



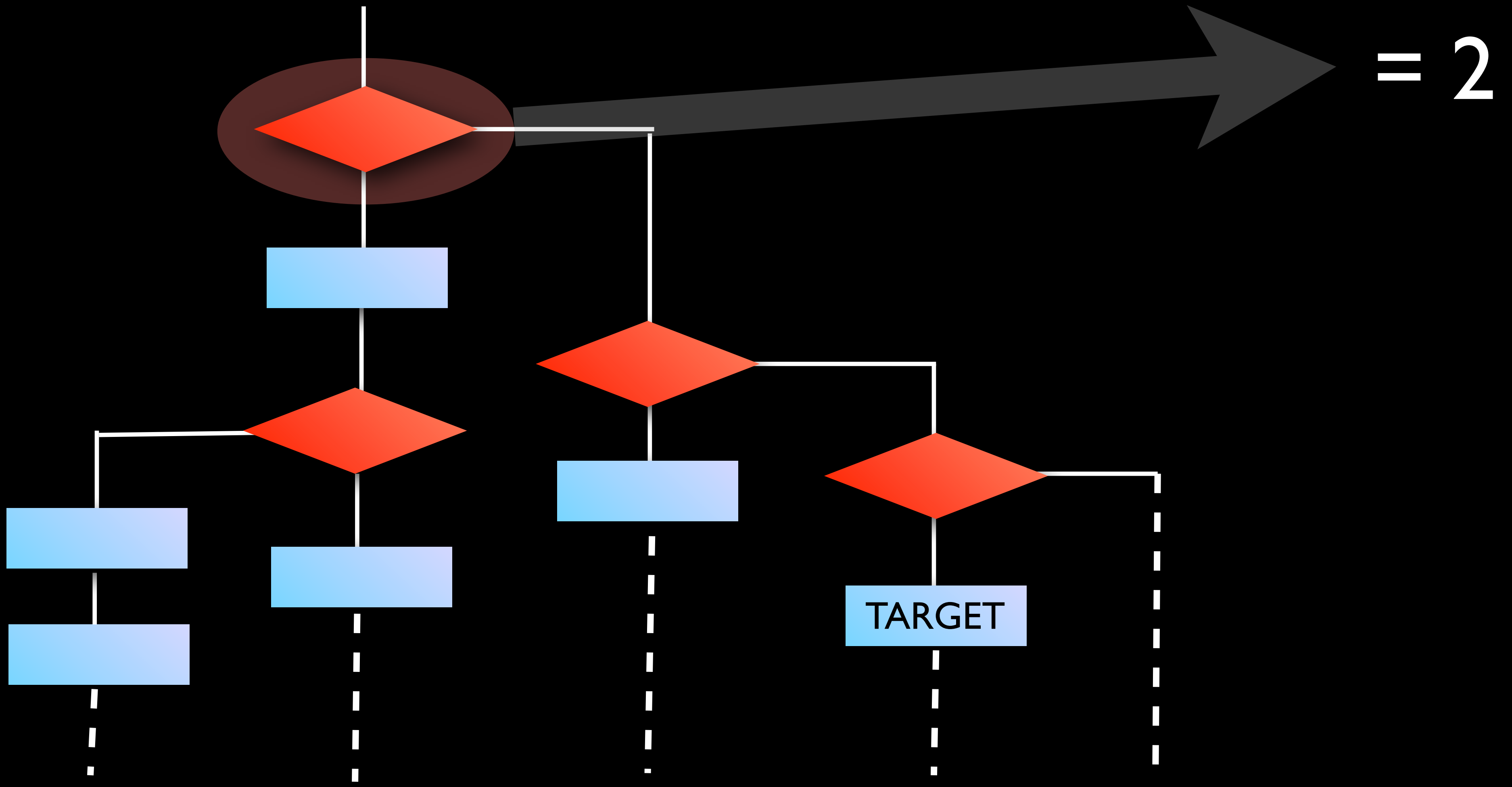
The outcomes at key decision statements matter.

These are the decisions on which the target is control dependent

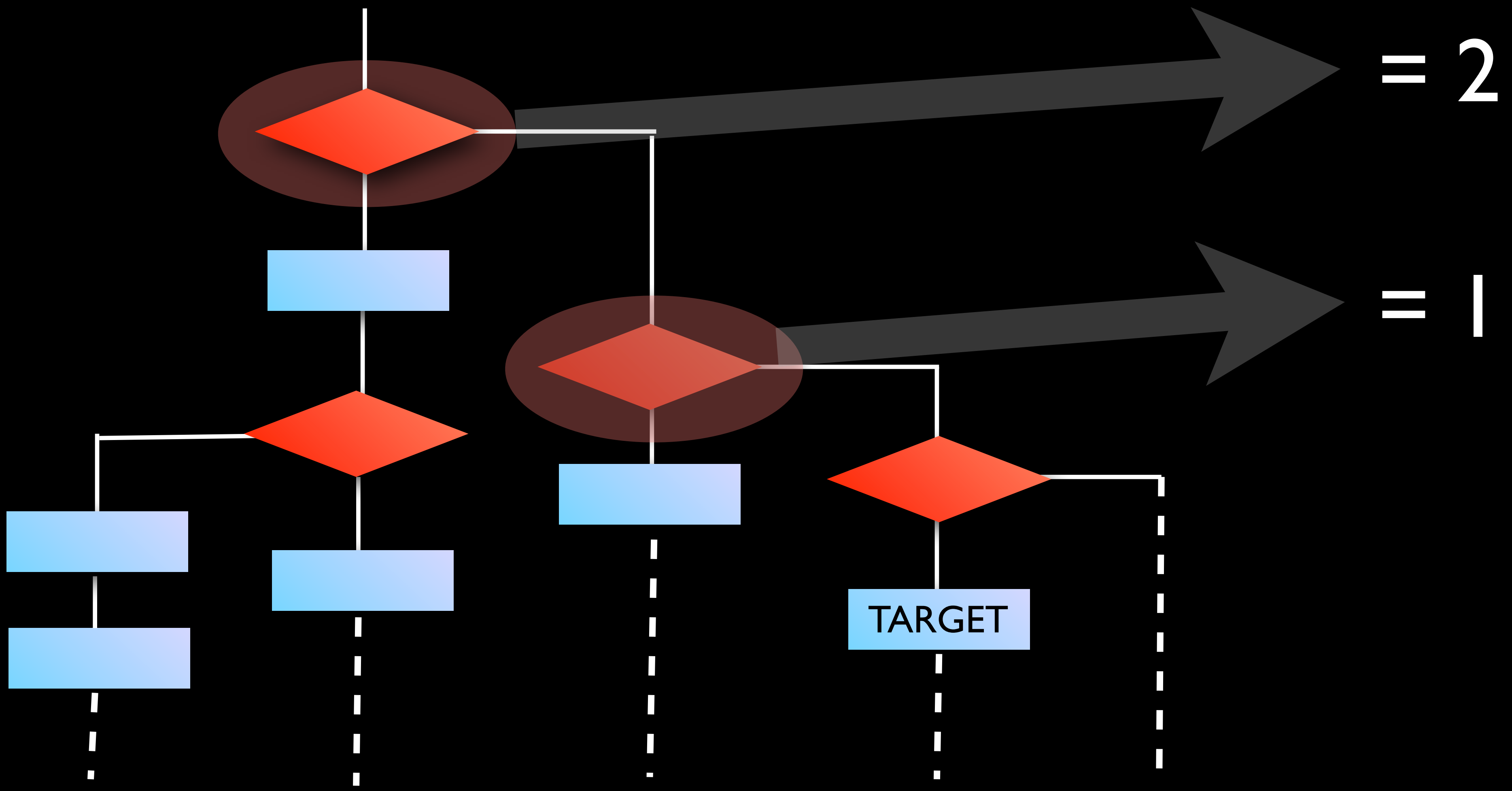
Approach Level



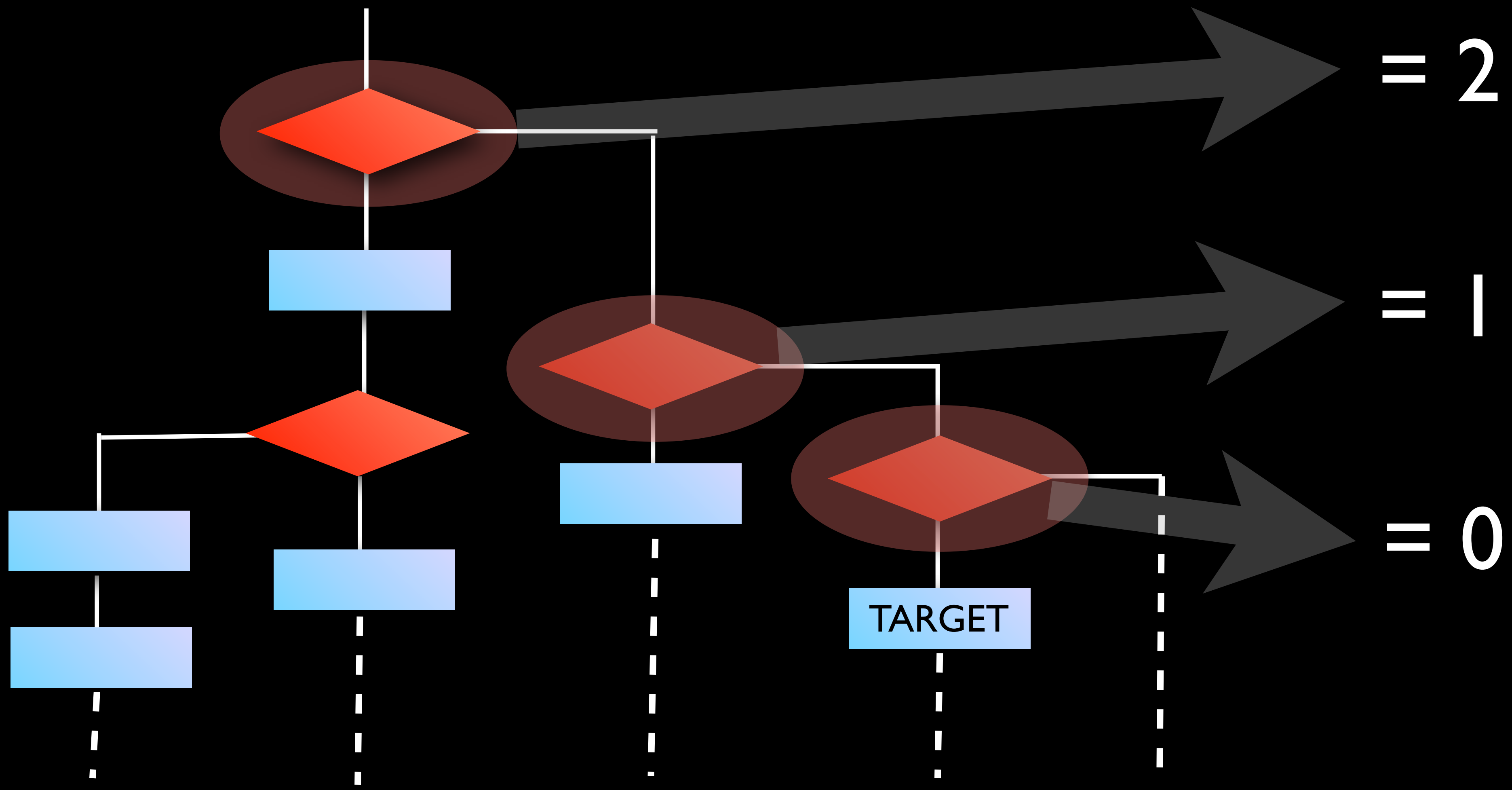
Approach Level



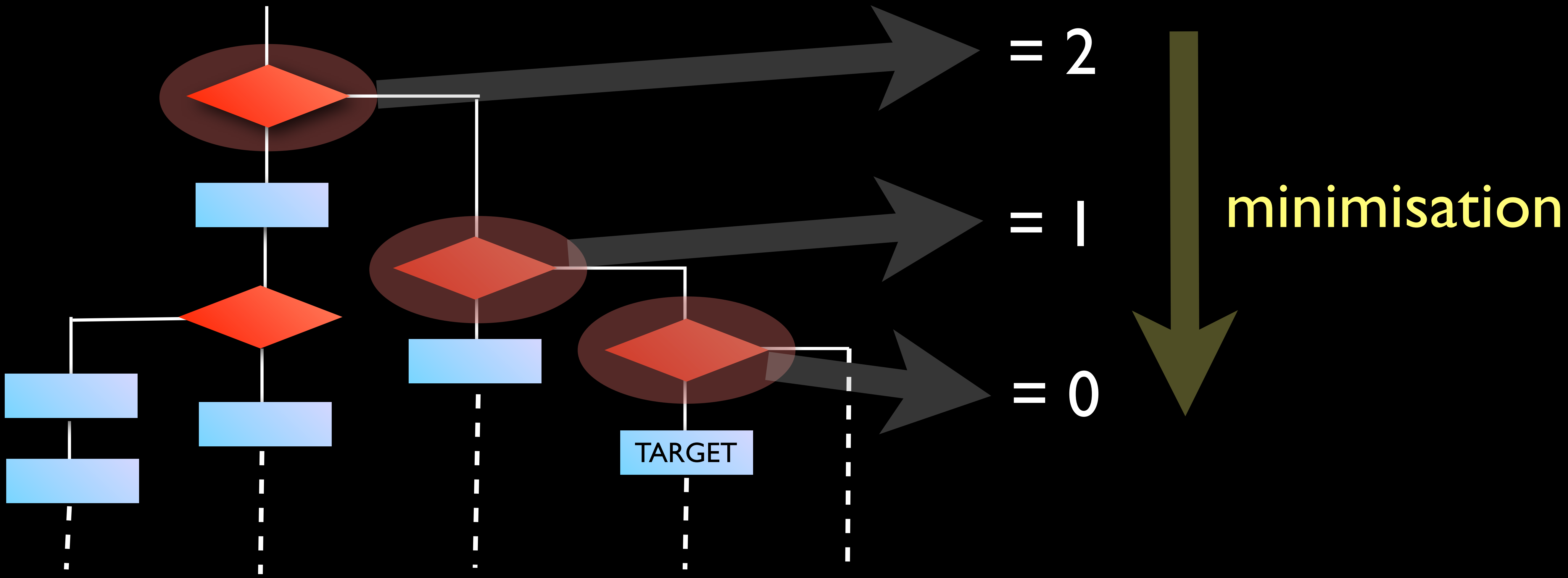
Approach Level



Approach Level

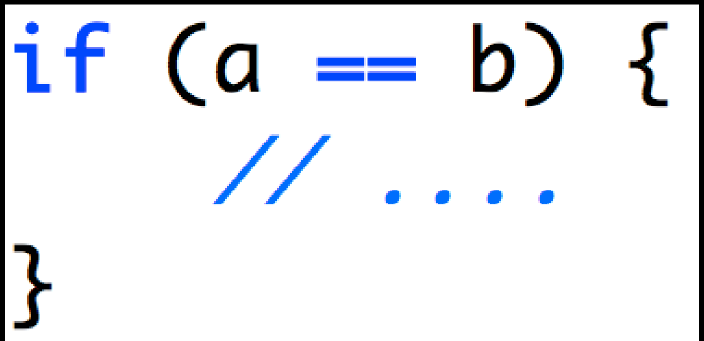


Approach Level



Analysing predicates

Approach level alone gives us coarse values

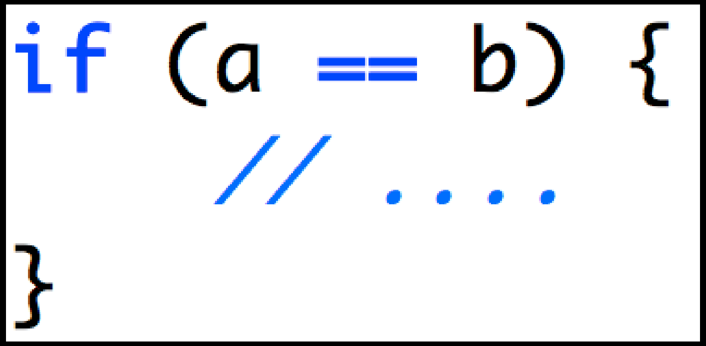


```
if (a == b) {
    // ....
}
```

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

Analysing predicates

Approach level alone gives us coarse values



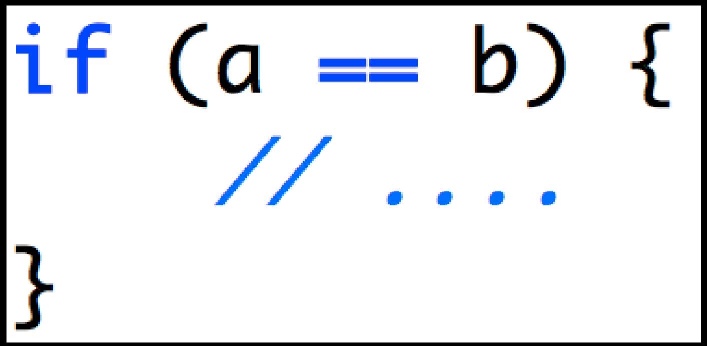
```
if (a == b) {
    // ....
}
```

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

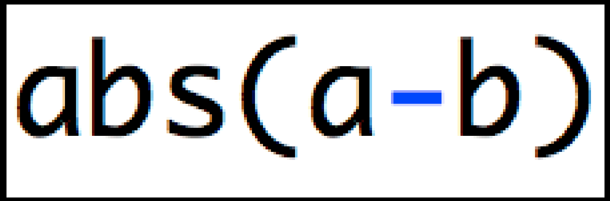
getting ‘closer’
to being true

Branch distance

Associate a distance formula with different relational predicates



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



```
abs(a-b)
```

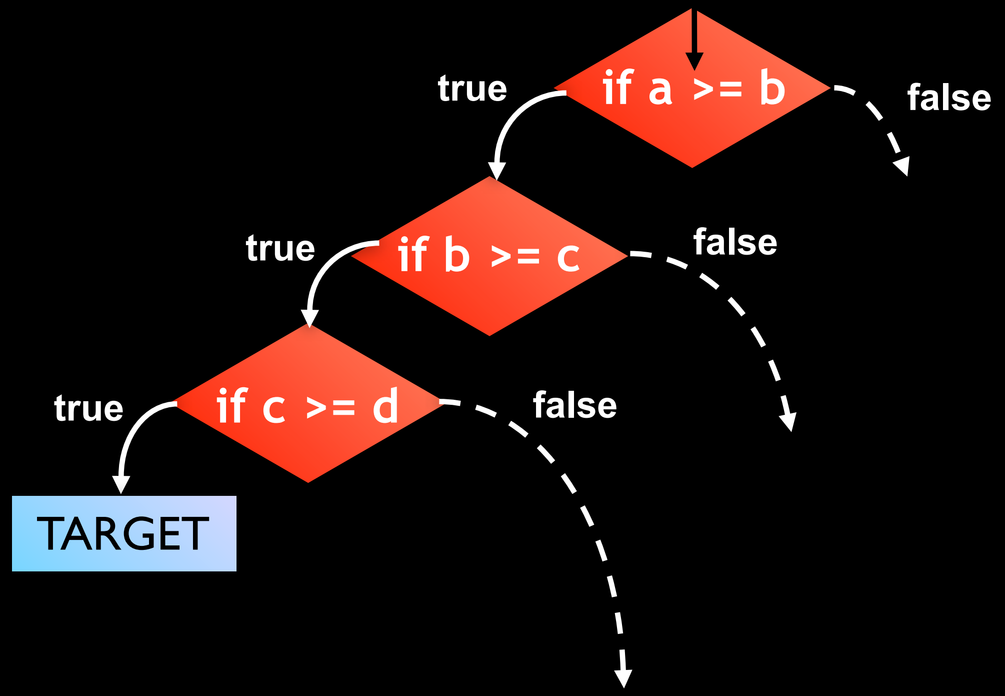
a = 50, b = 0	branch distance = 50
a = 45, b = 5	branch distance = 40
a = 40, b = 10	branch distance = 30
a = 35, b = 15	branch distance = 20
a = 30, b = 20	branch distance = 10
a = 25, b = 25	branch distance = 0

getting ‘closer’ to being true

Putting it all together

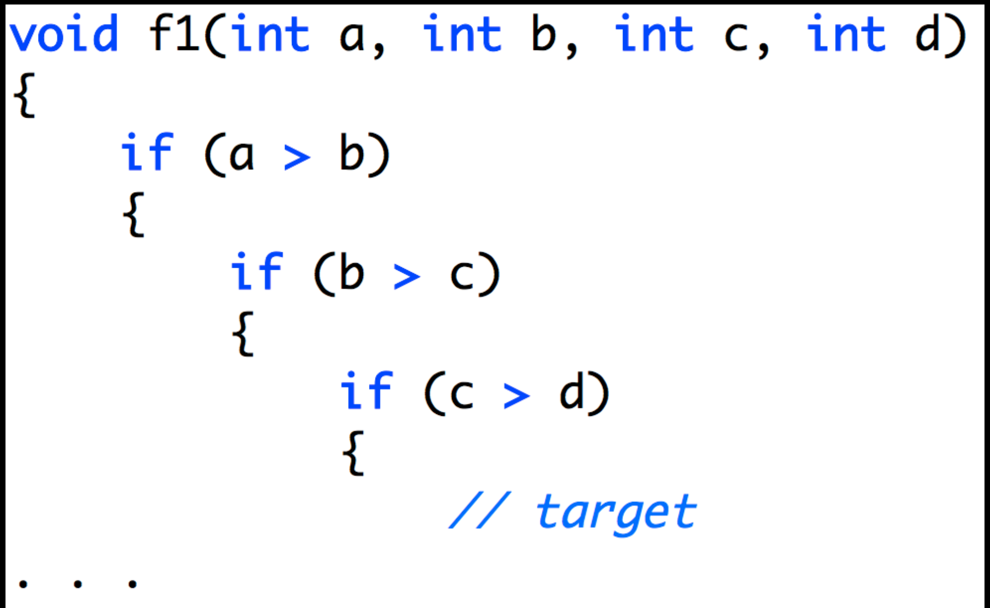
Fitness = approach Level + *normalised* branch distance

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

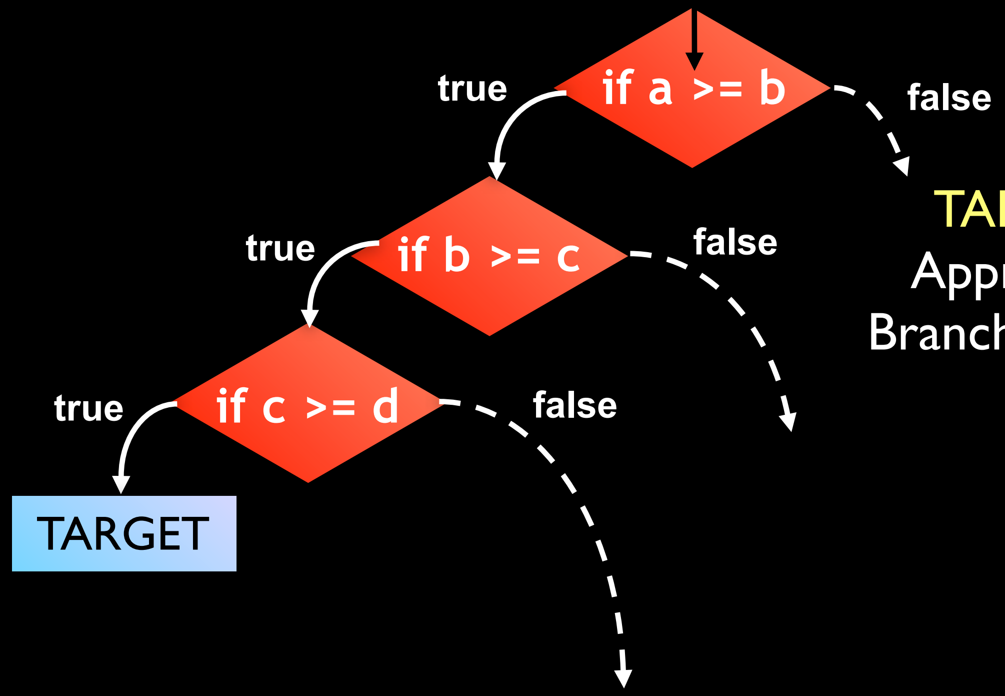


Putting it all together

Fitness = approach Level + *normalised* branch distance



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

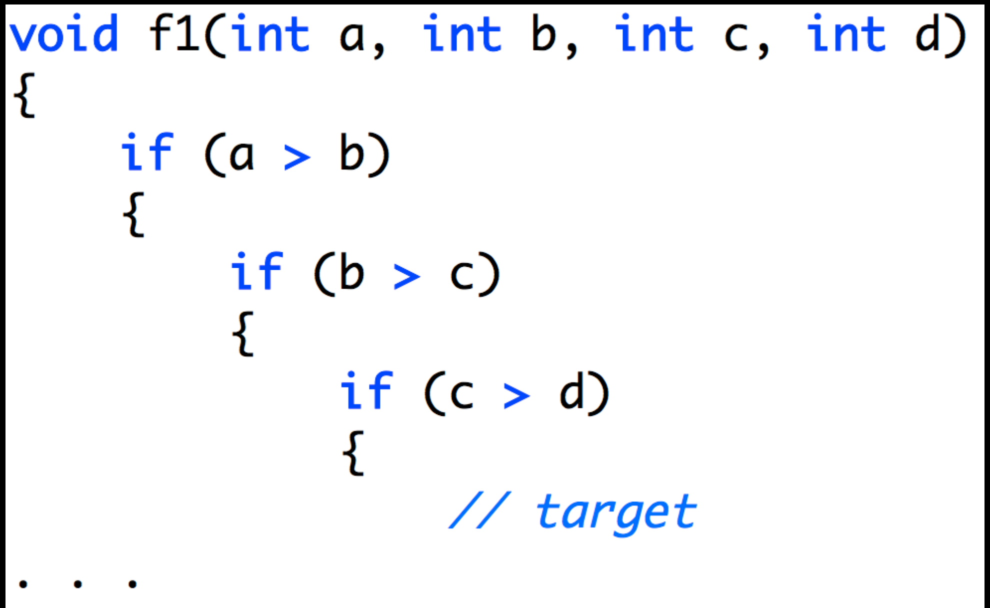


TARGET MISSED
Approach Level = 2
Branch Distance = b - a

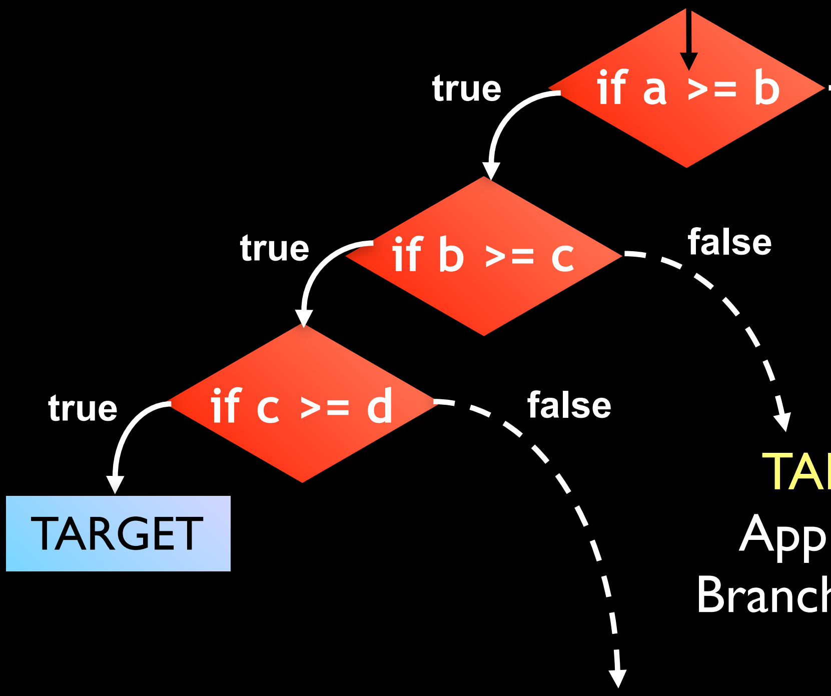
normalised branch distance between 0 and 1
indicates how close approach level is to being penetrated

Putting it all together

Fitness = approach Level + *normalised* branch distance



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



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

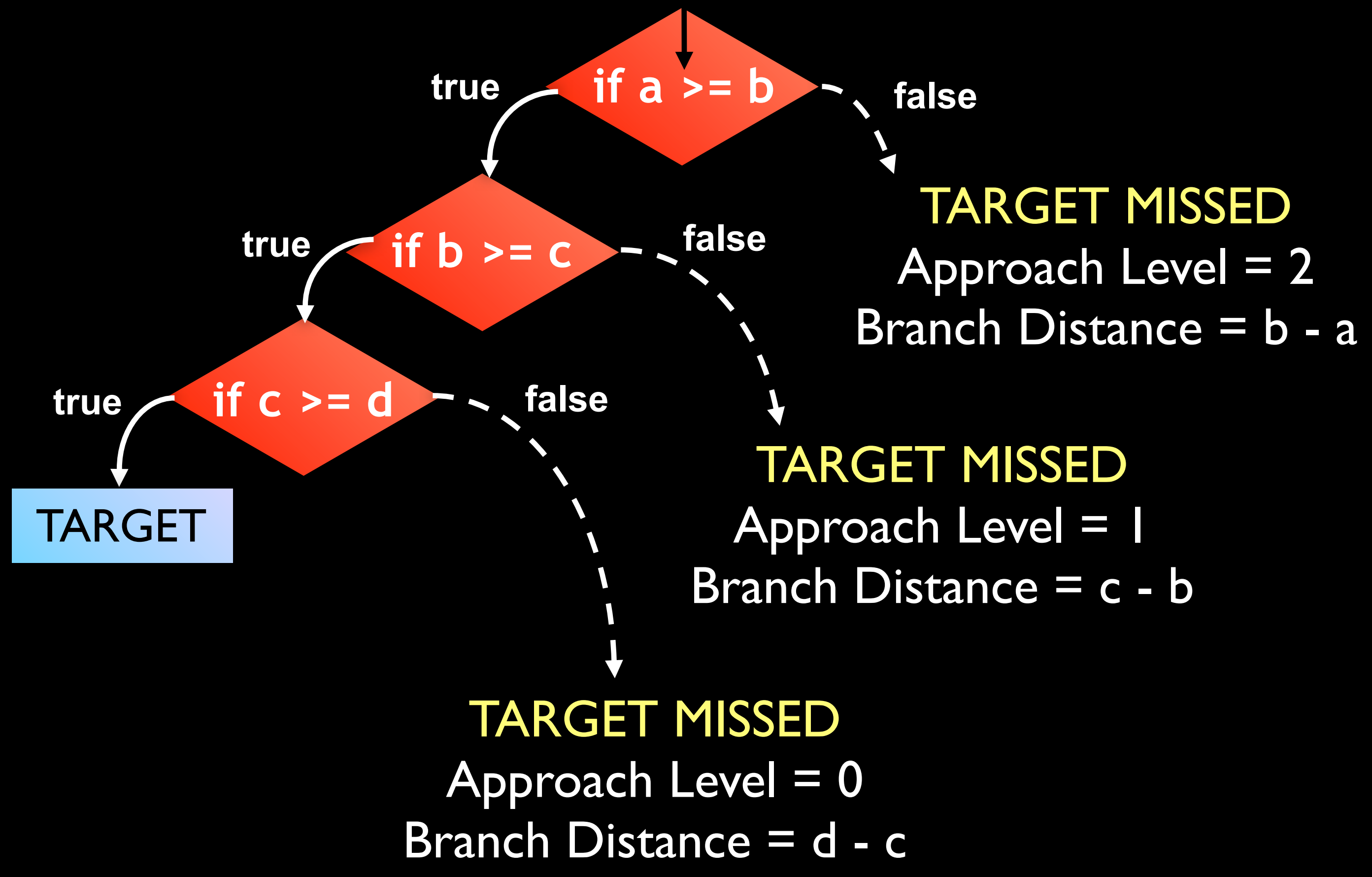
TARGET MISSED
Approach Level = 1
Branch Distance = c - b

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

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



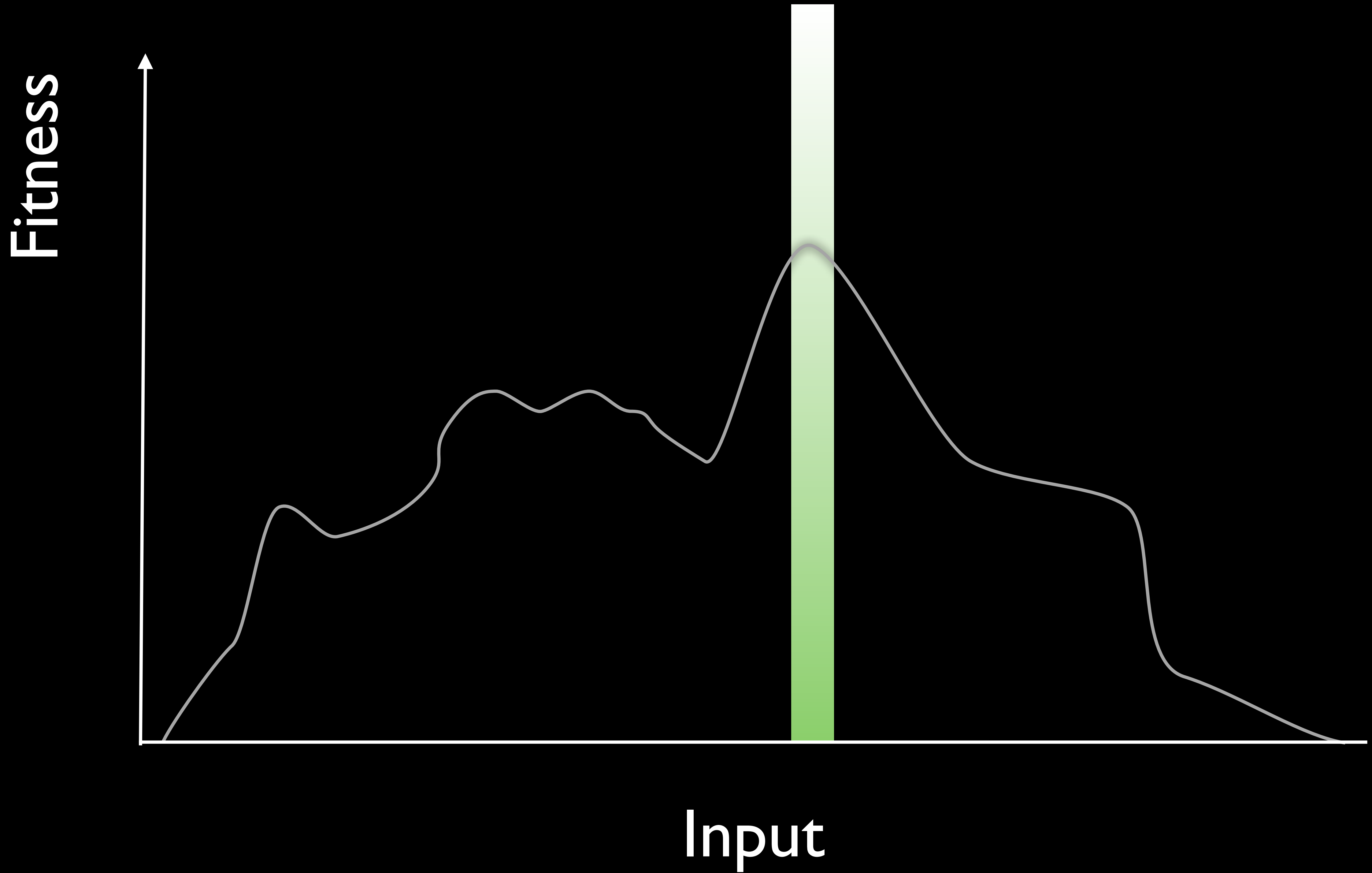
TARGET MISSED
Approach Level = 2
Branch Distance = b - a

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

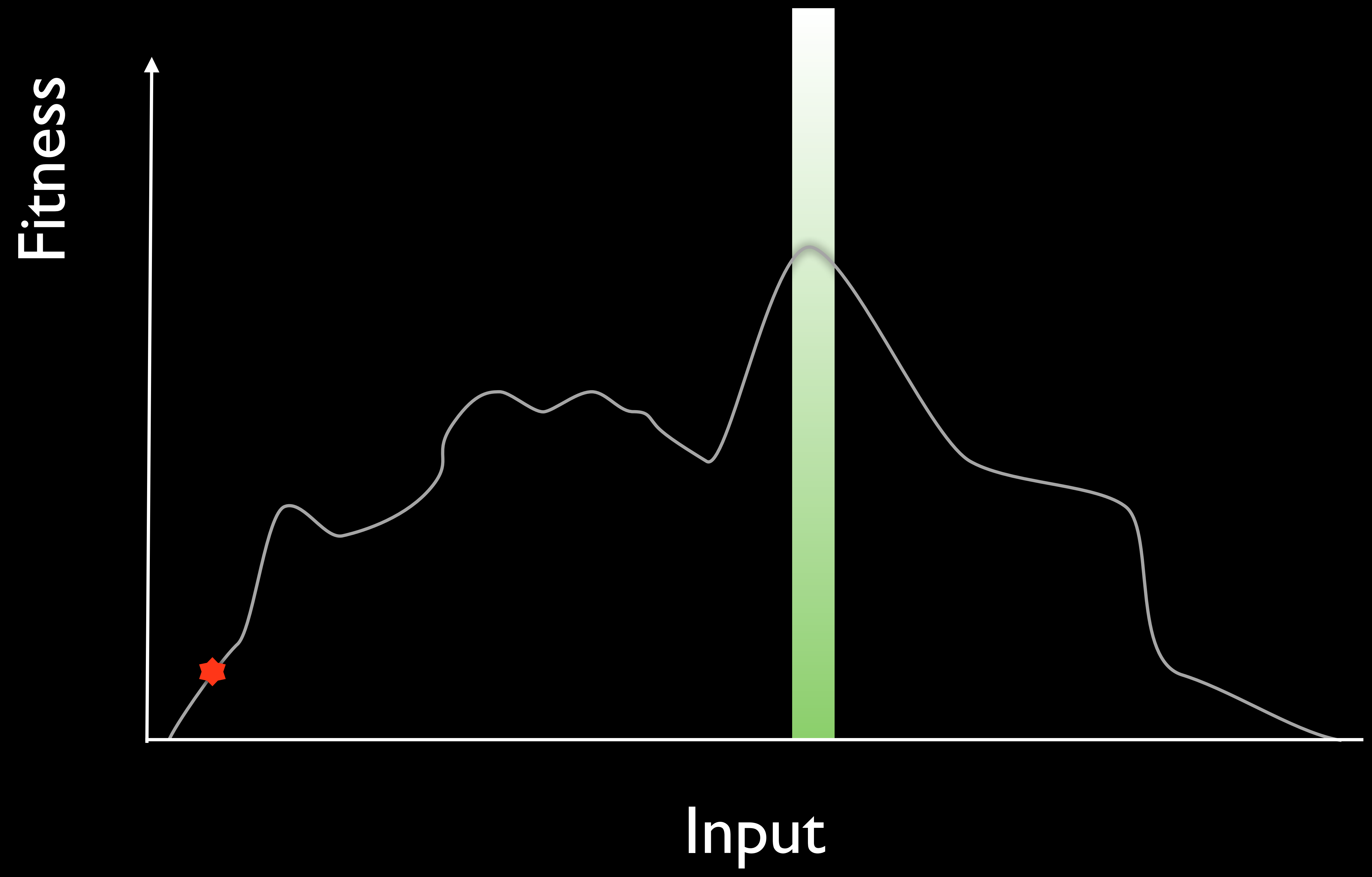
TARGET MISSED
Approach Level = 0
Branch Distance = d - c

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

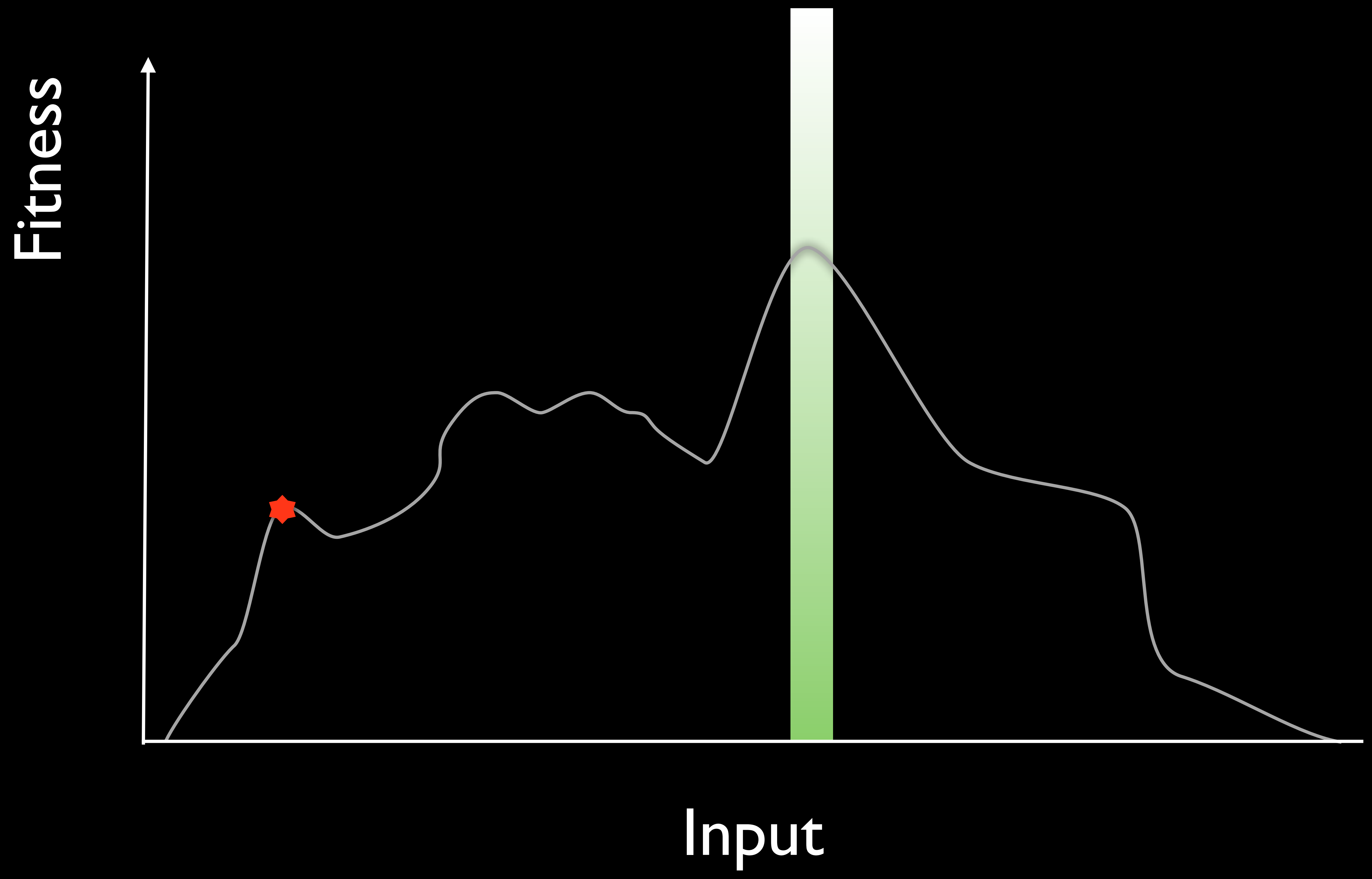
Hill Climbing



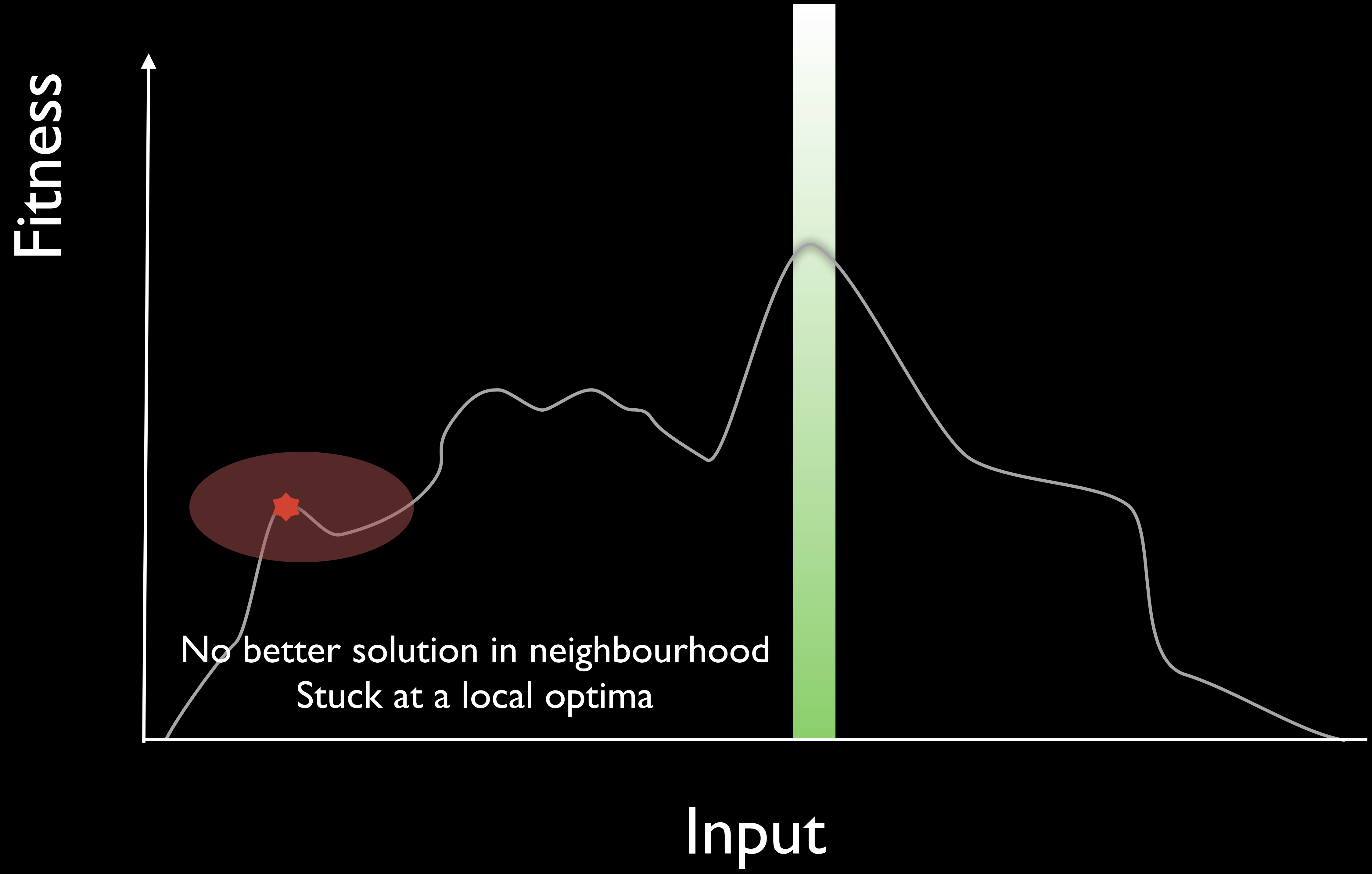
Hill Climbing



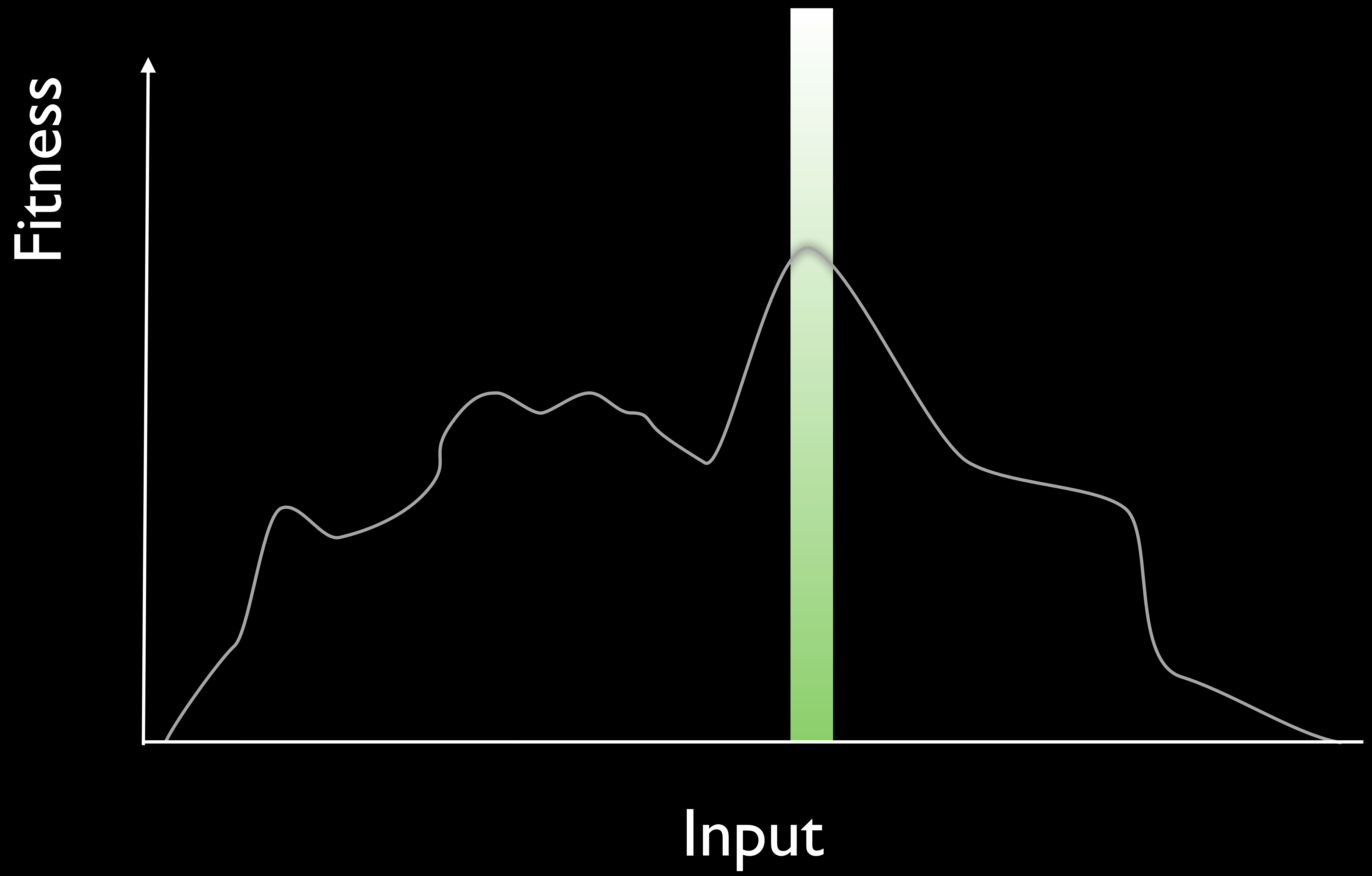
Hill Climbing



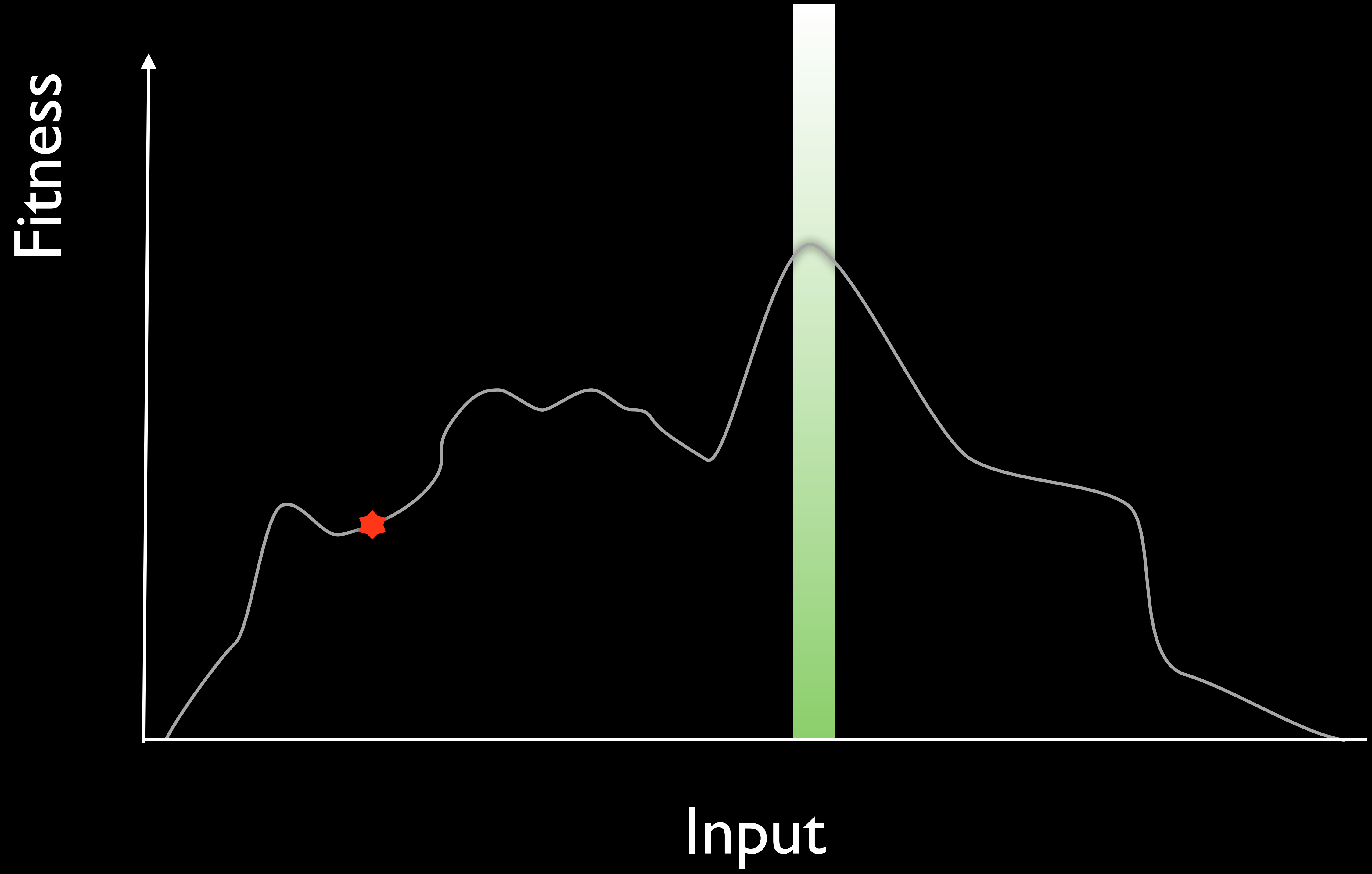
Hill Climbing



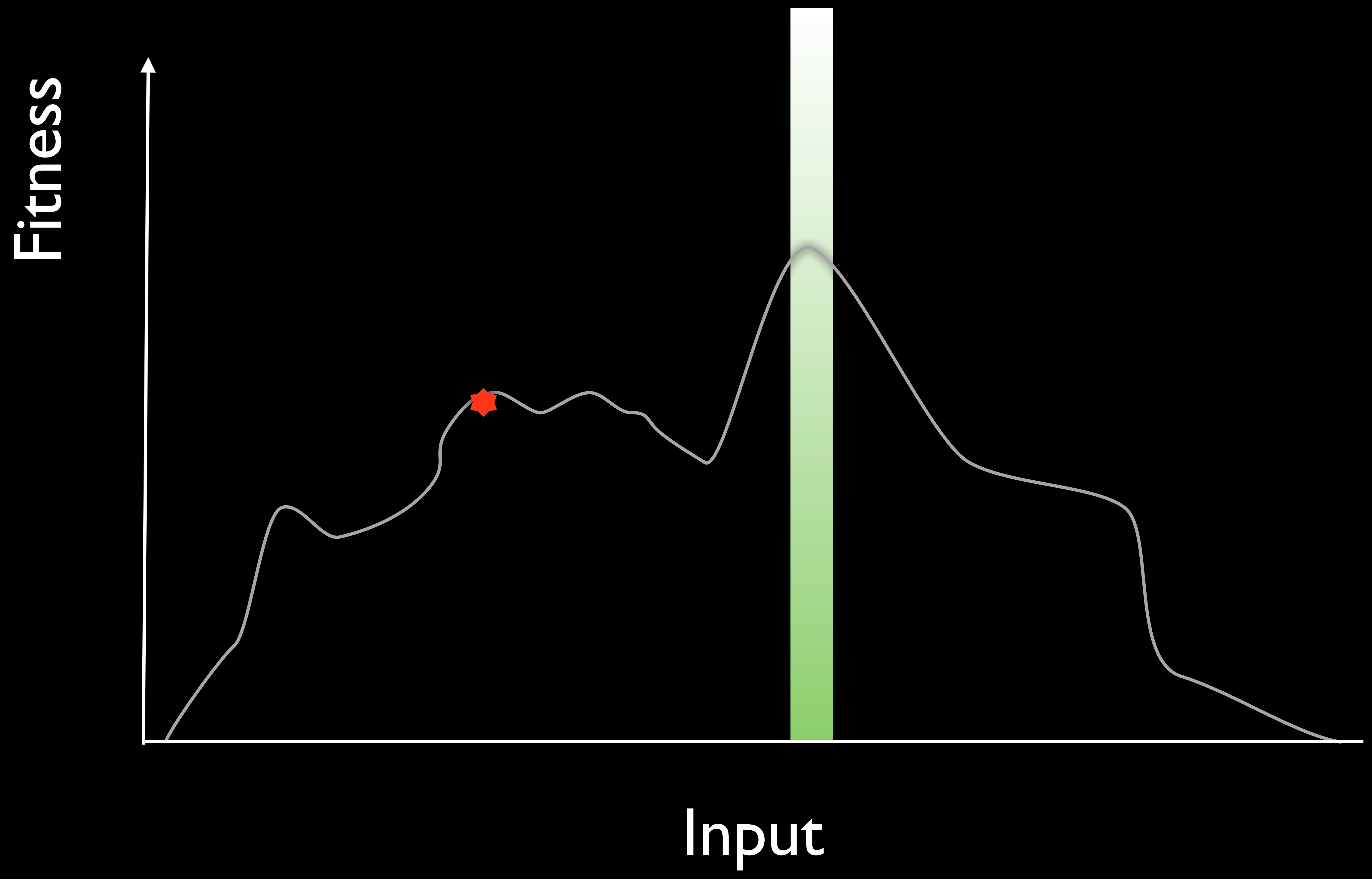
Hill Climbing - Restarts



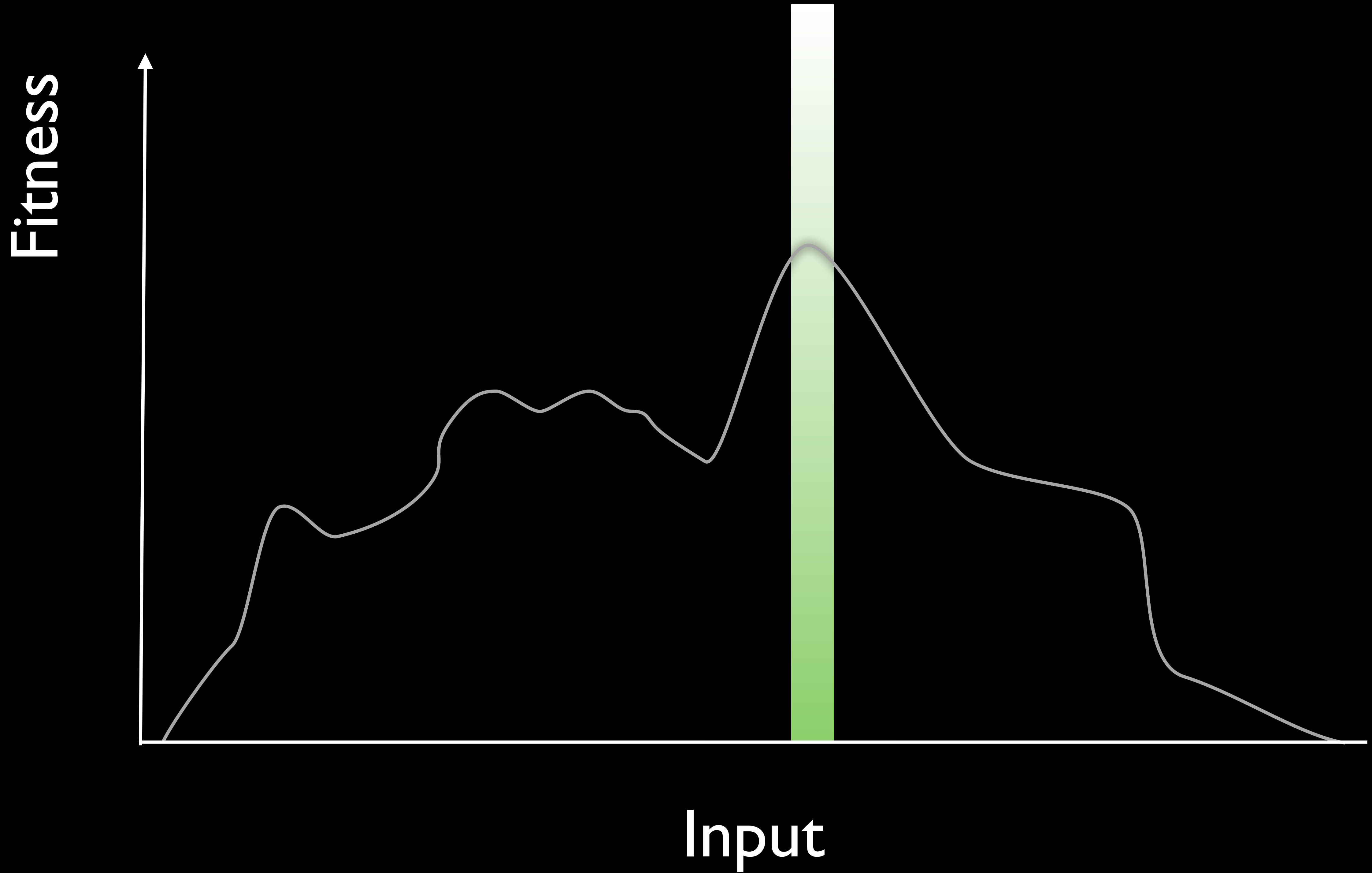
Hill Climbing - Restarts



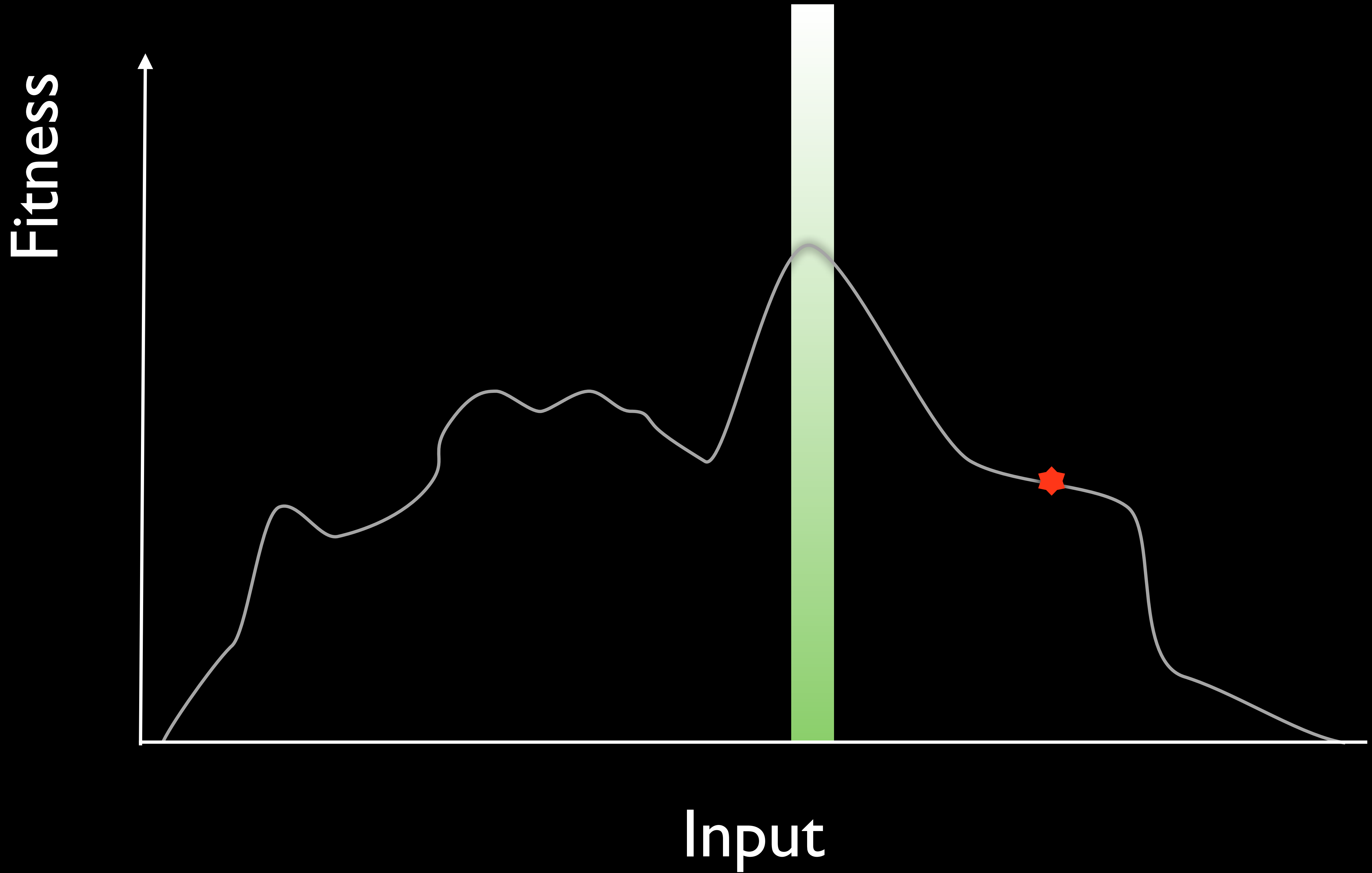
Hill Climbing - Restarts



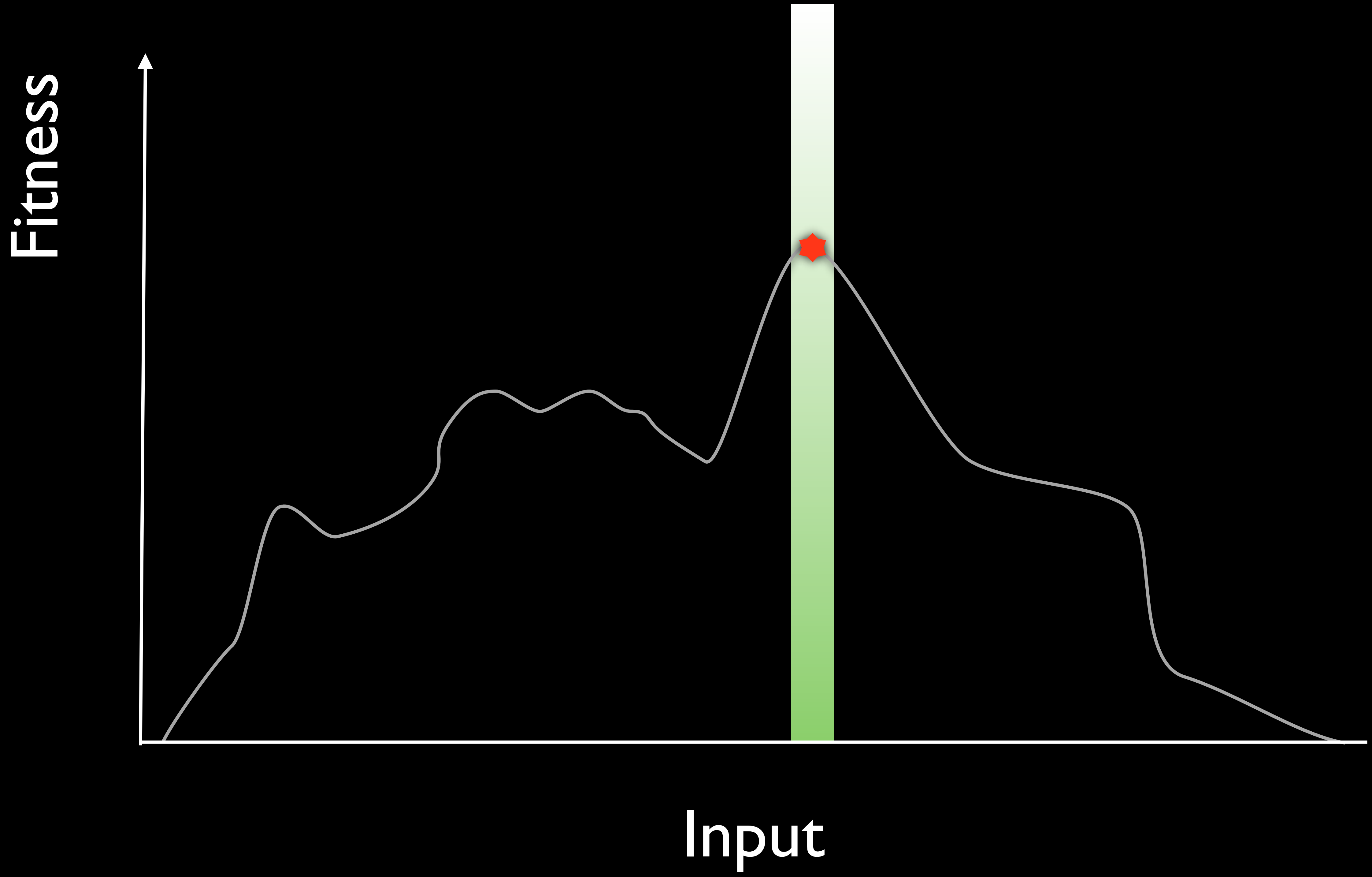
Hill Climbing - Restarts



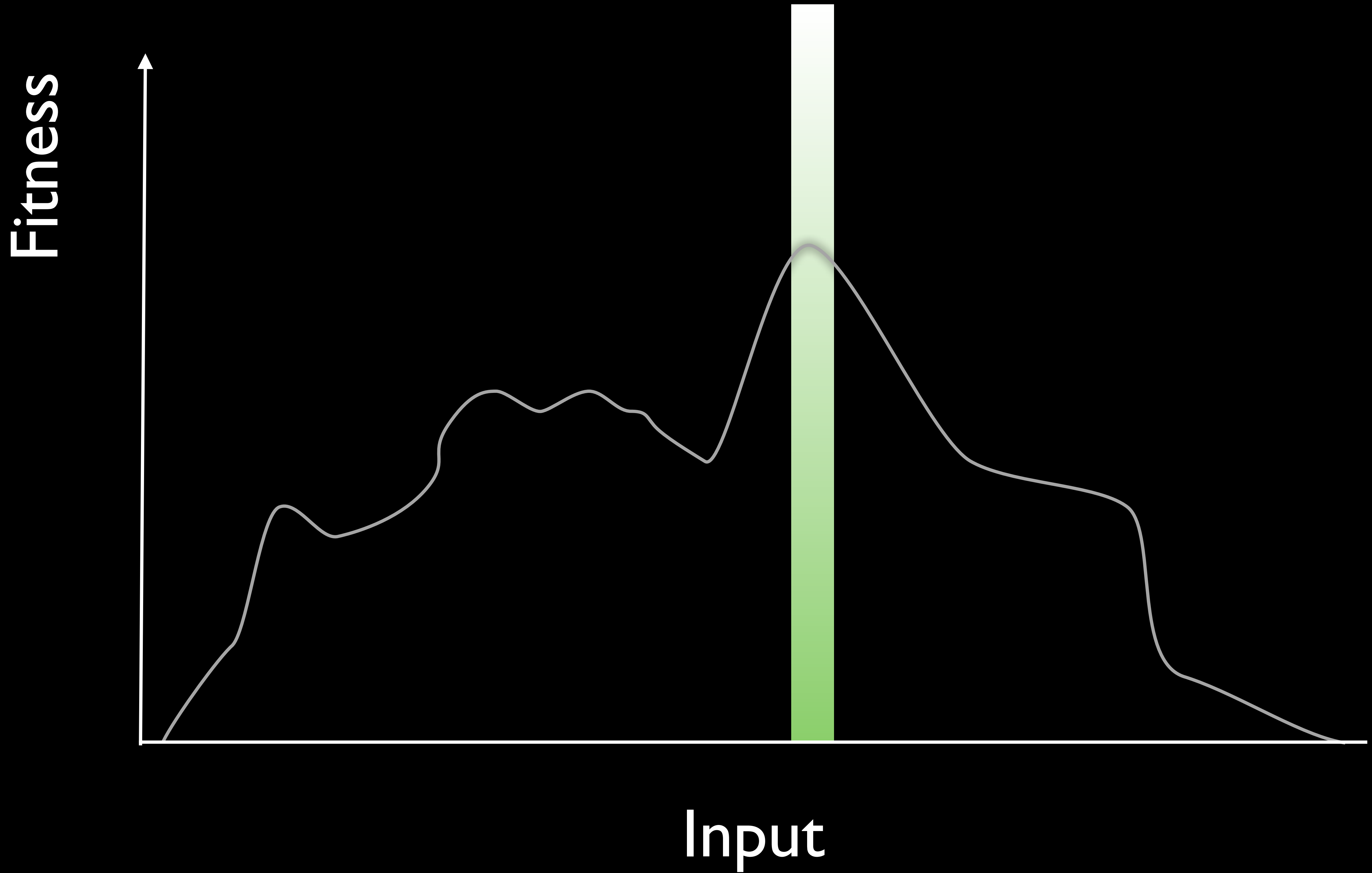
Hill Climbing - Restarts



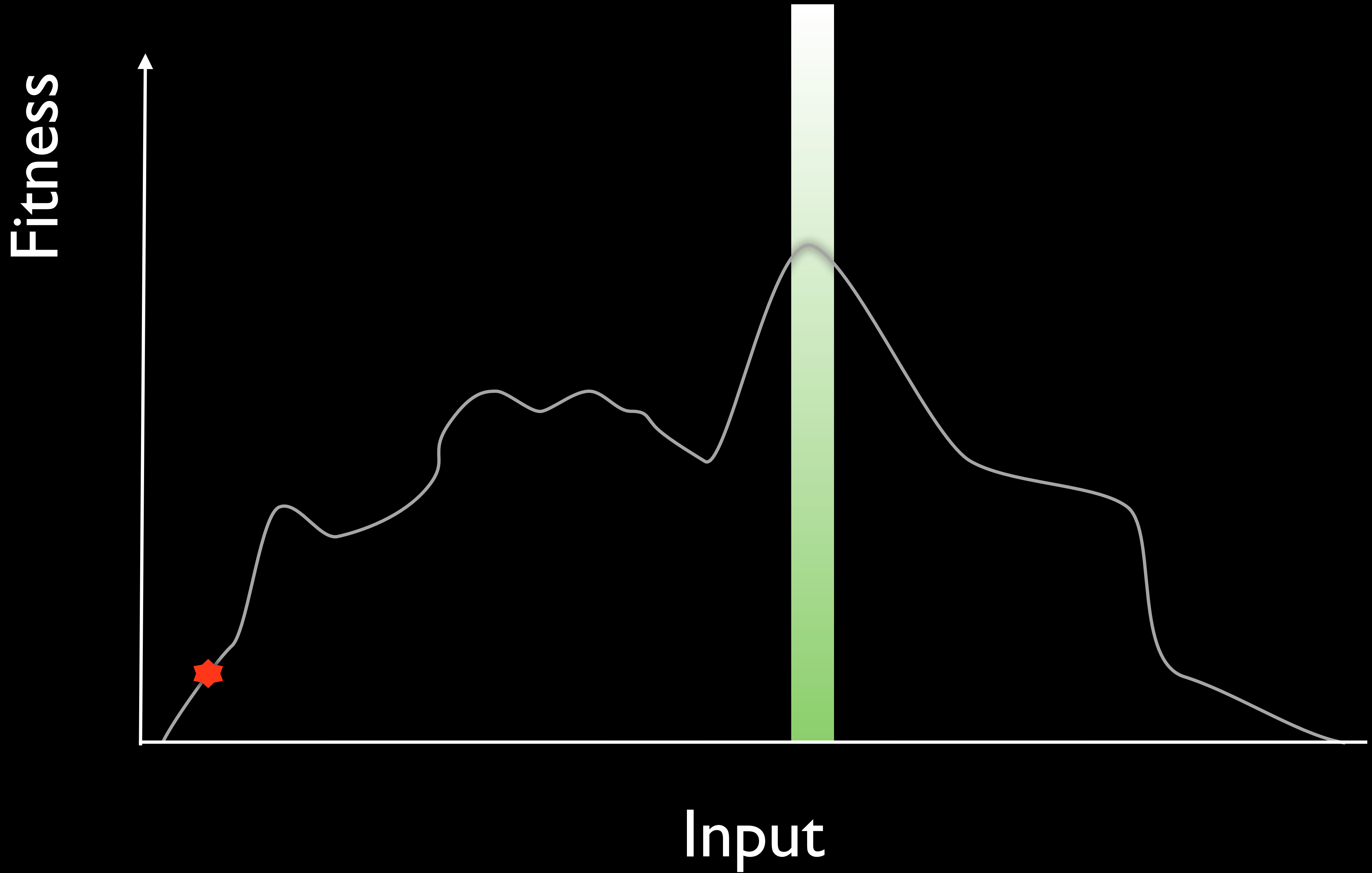
Hill Climbing - Restarts



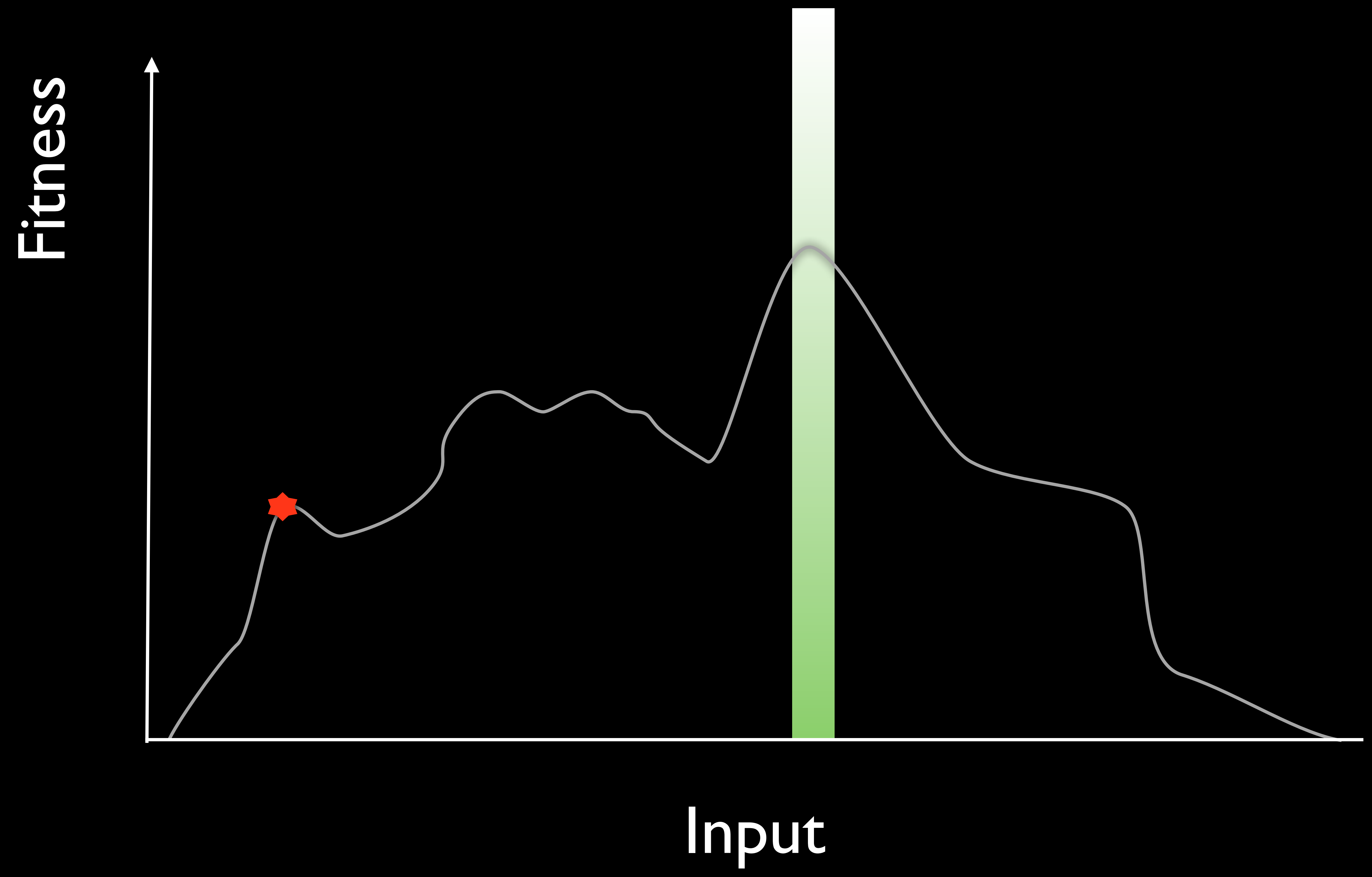
Simulated Annealing



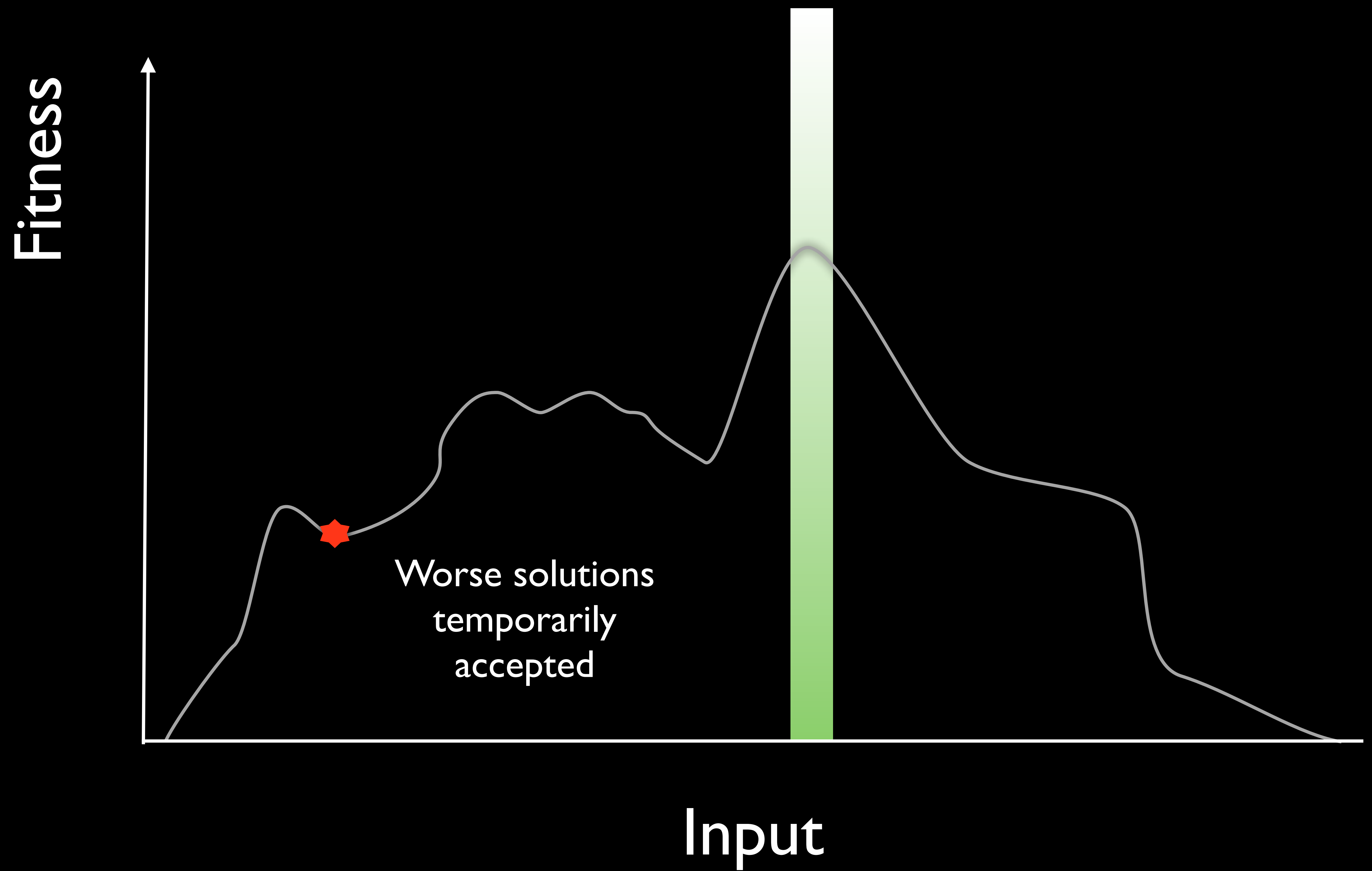
Simulated Annealing



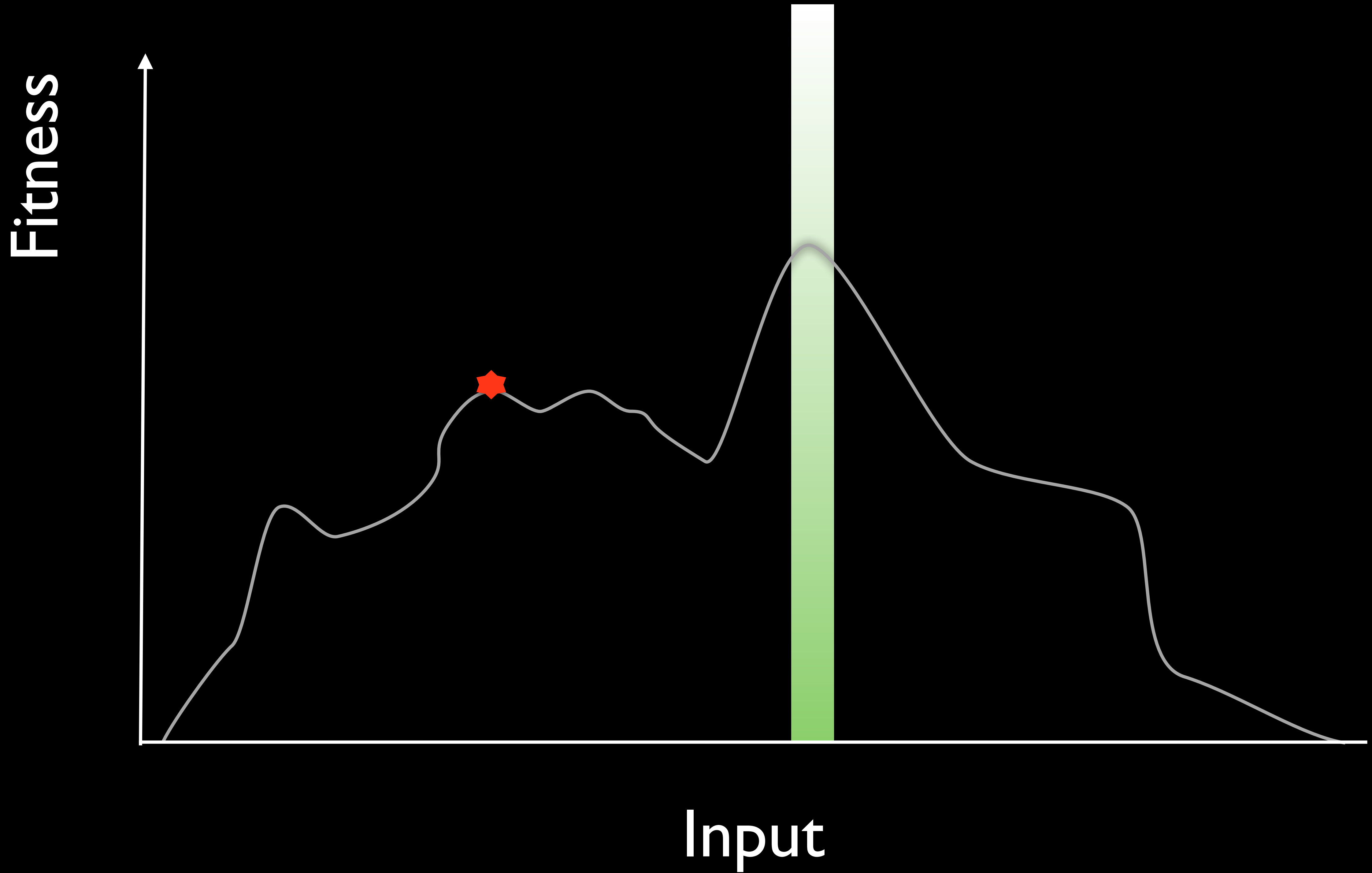
Simulated Annealing



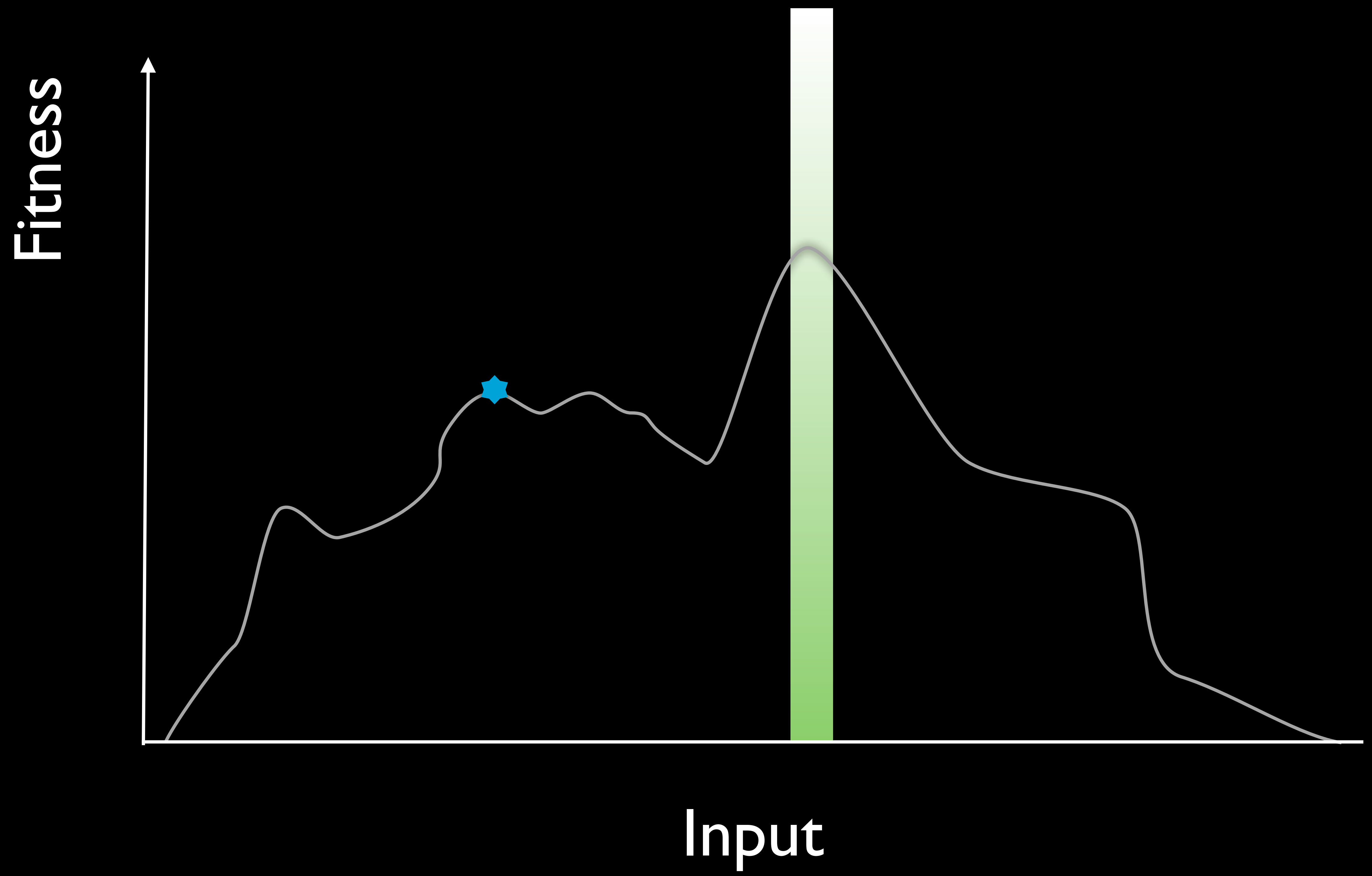
Simulated Annealing



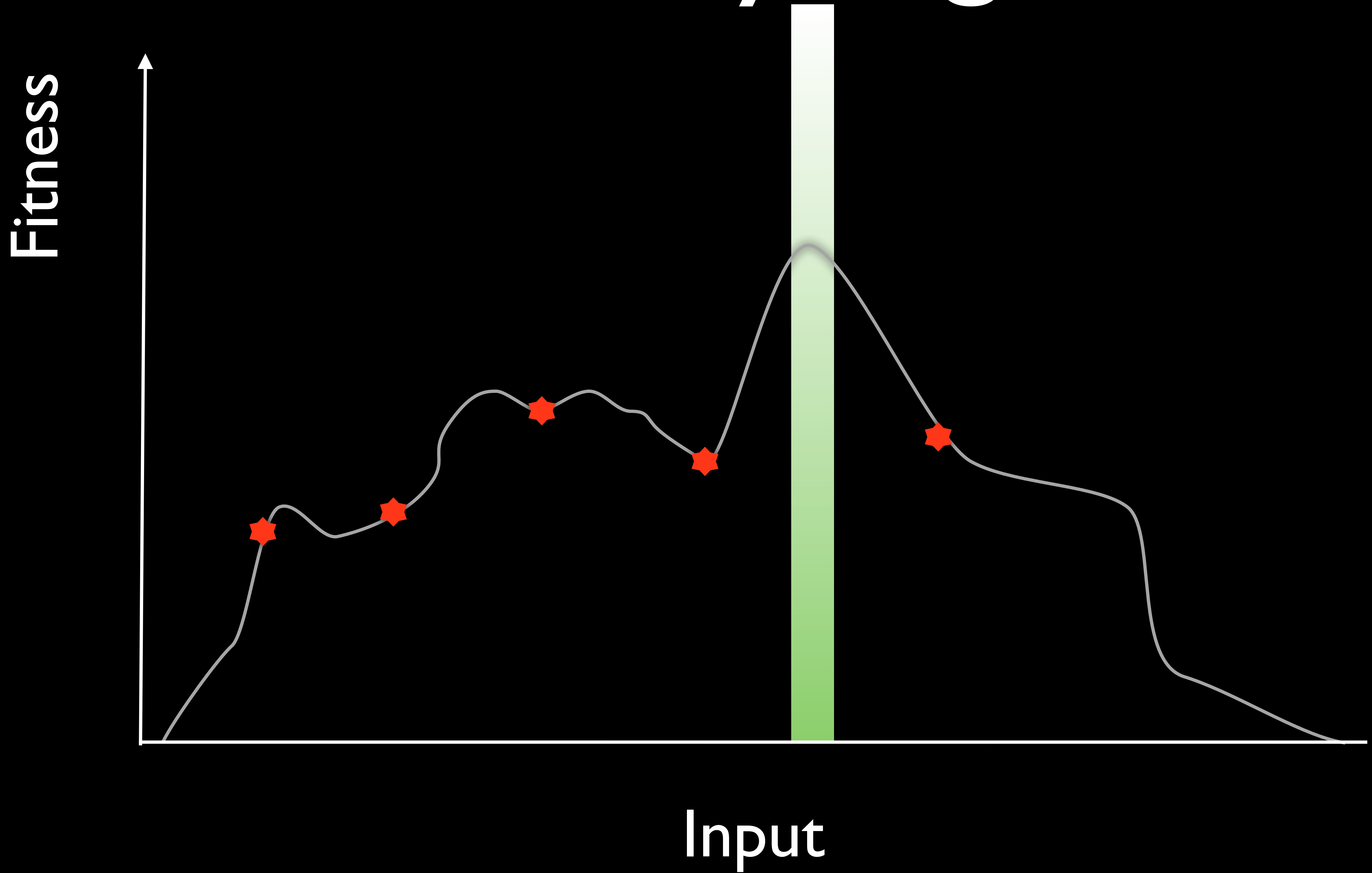
Simulated Annealing



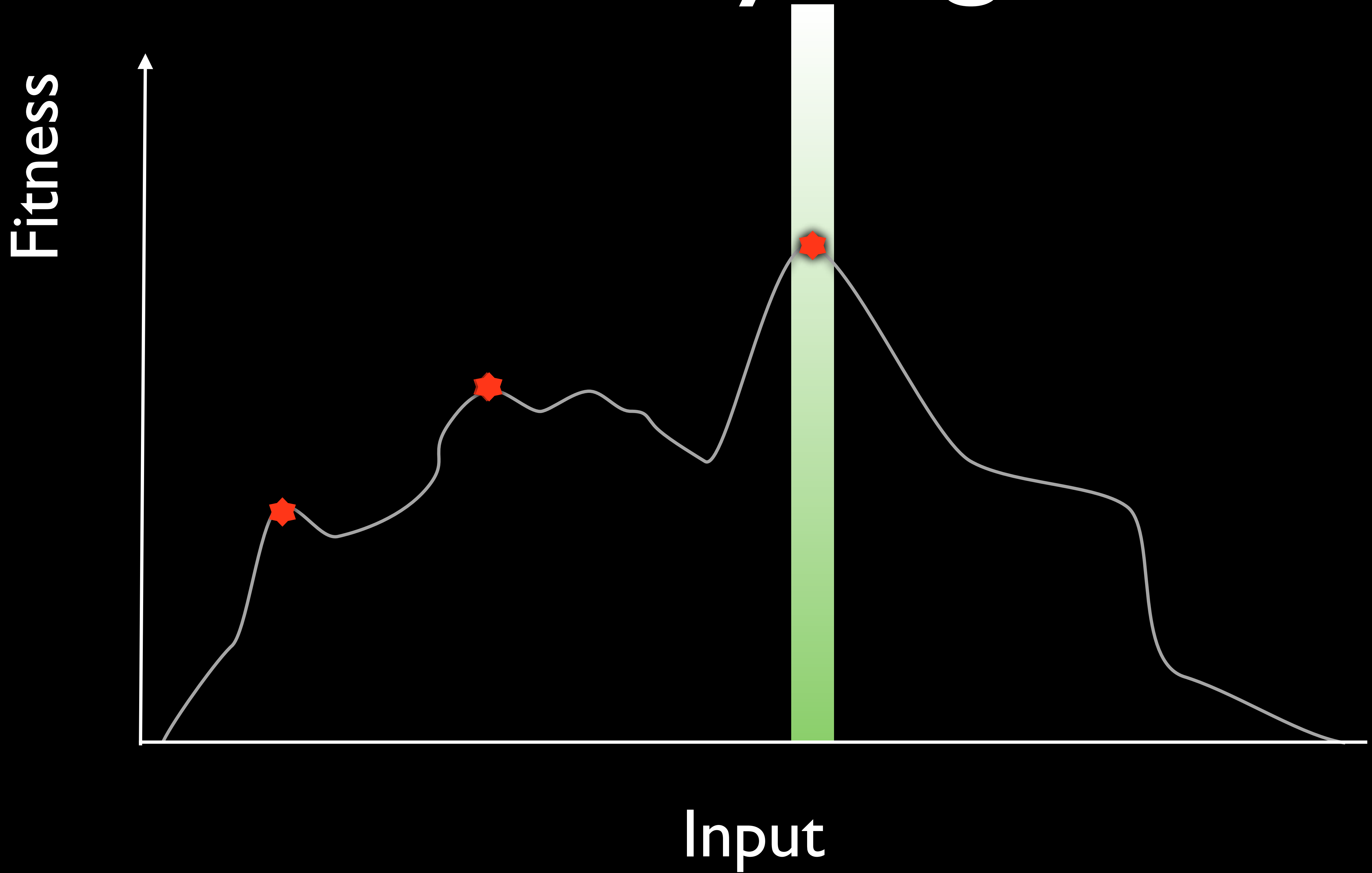
Simulated Annealing



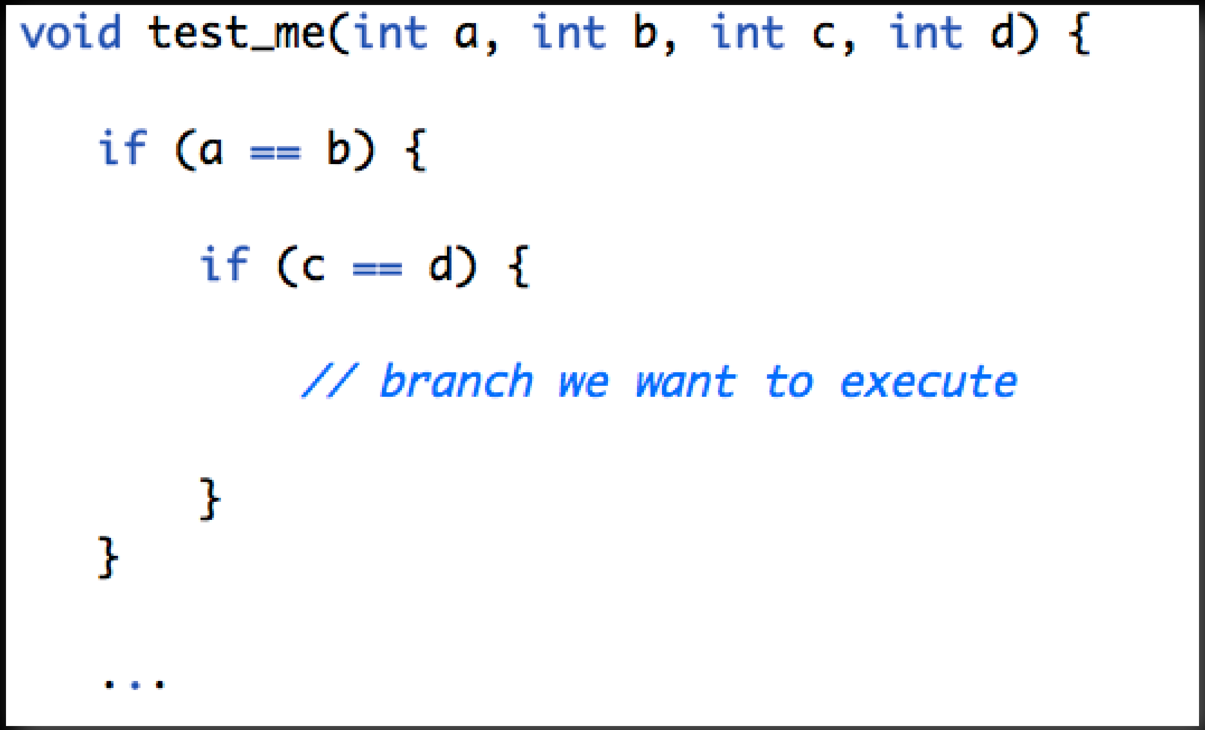
Evolutionary Algorithm



Evolutionary Algorithm



Crossover

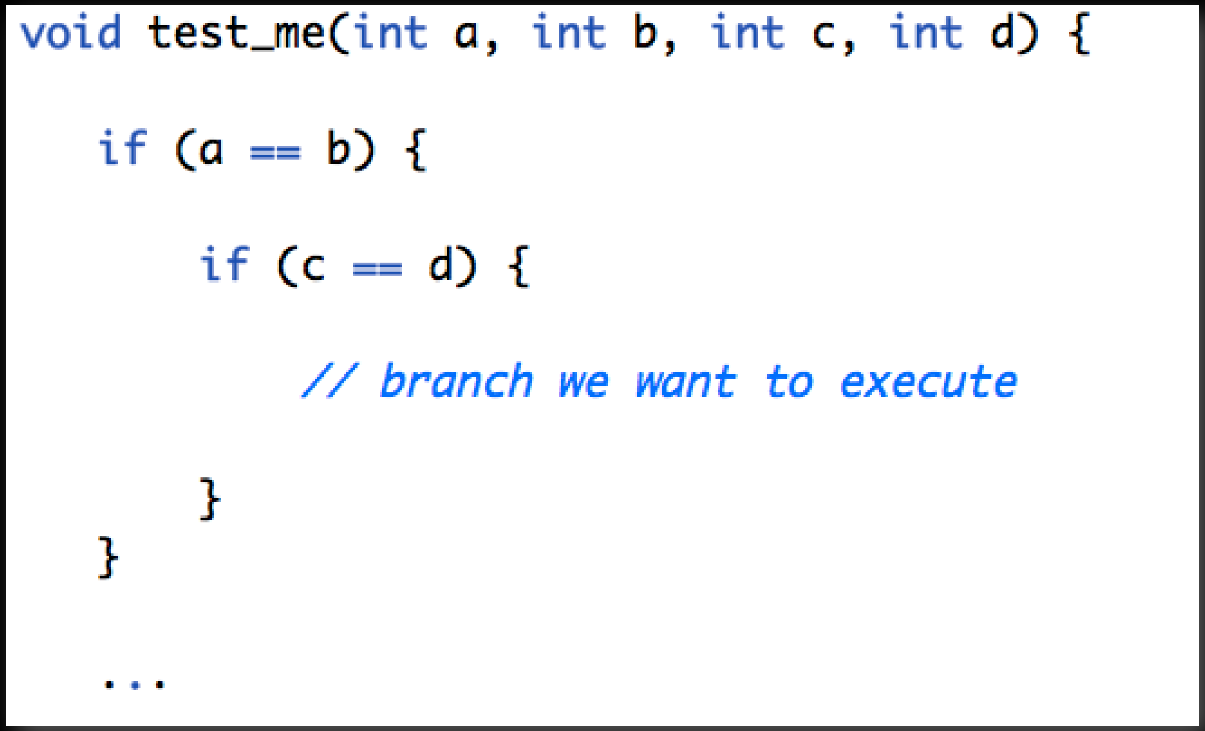


```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

       if (c == d) {

           // branch we want to execute

       }
   }

   ...
```

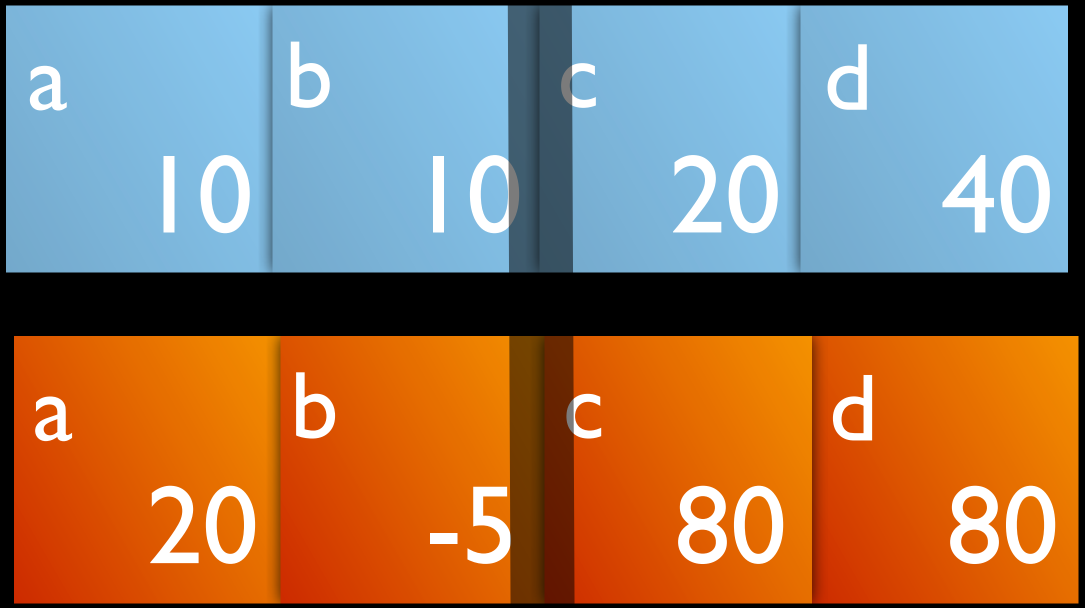
a	b	c	d
10	10	20	40

a	b	c	d
20	-5	80	80

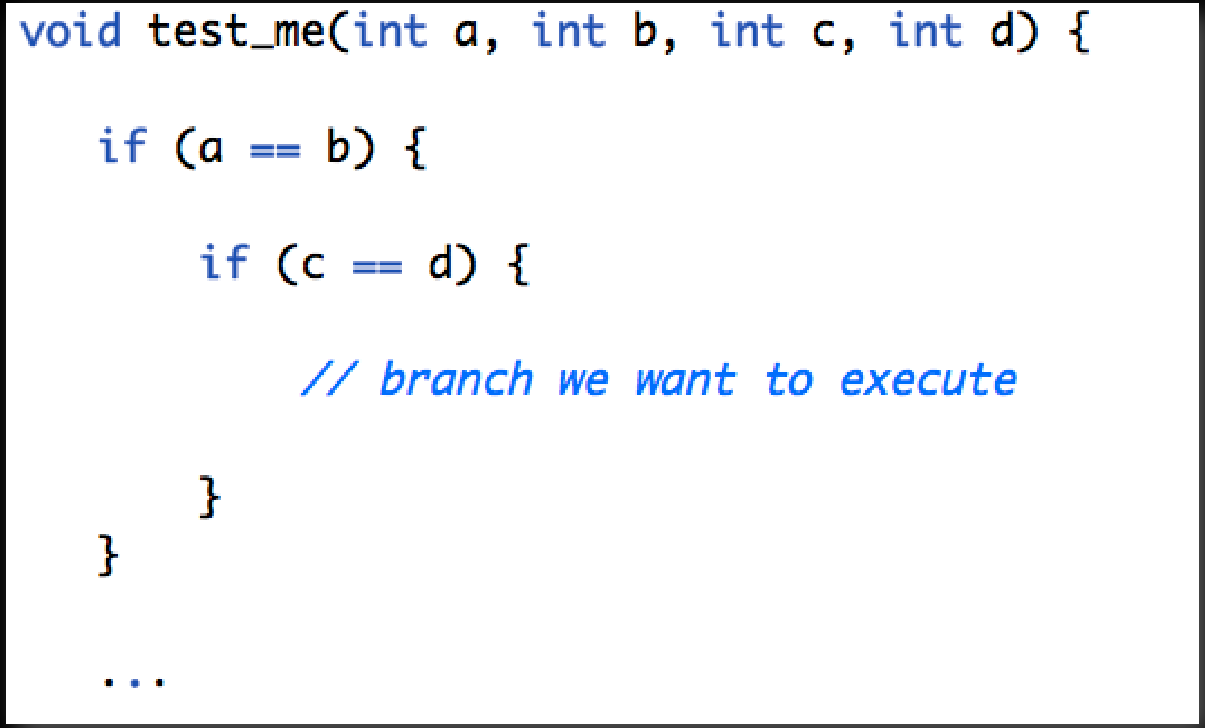
Crossover



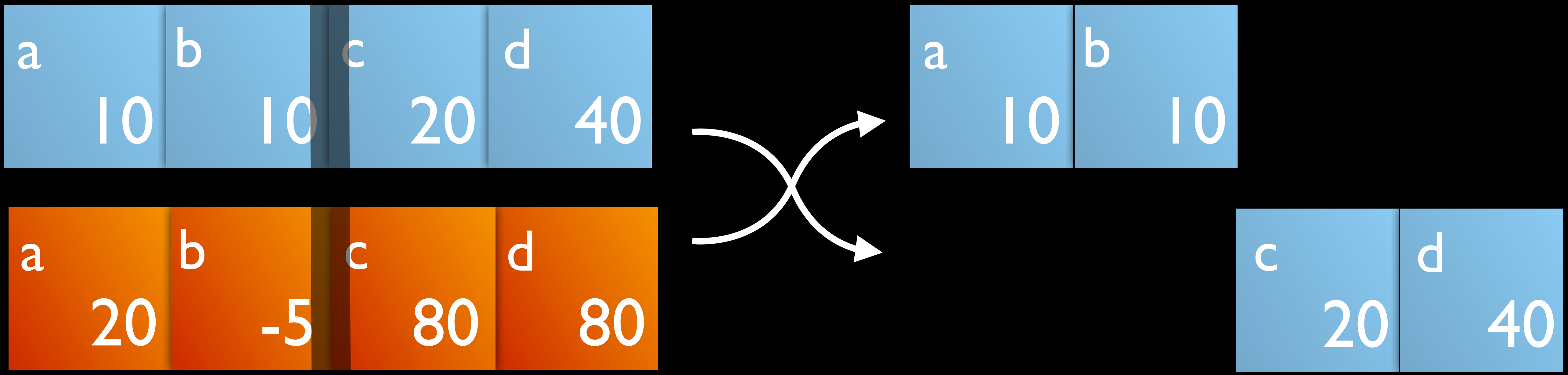
```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

       if (c == d) {

           // branch we want to execute

       }
   }

   ...
```



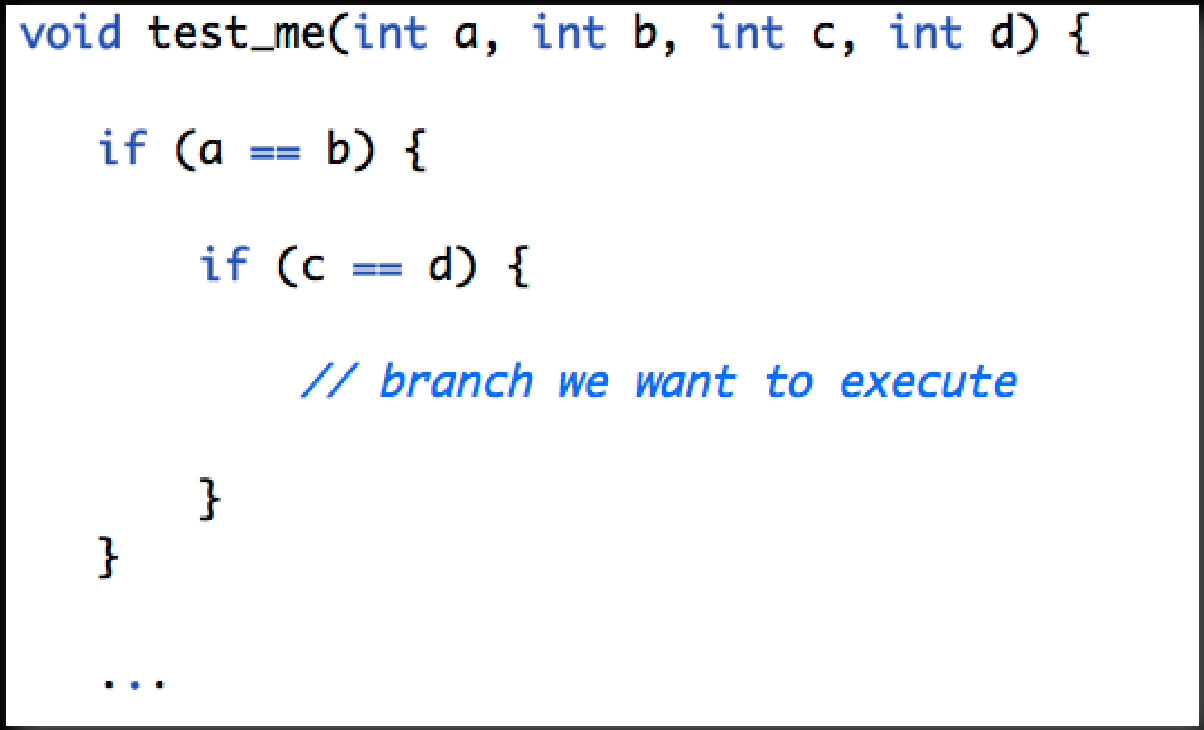
Crossover



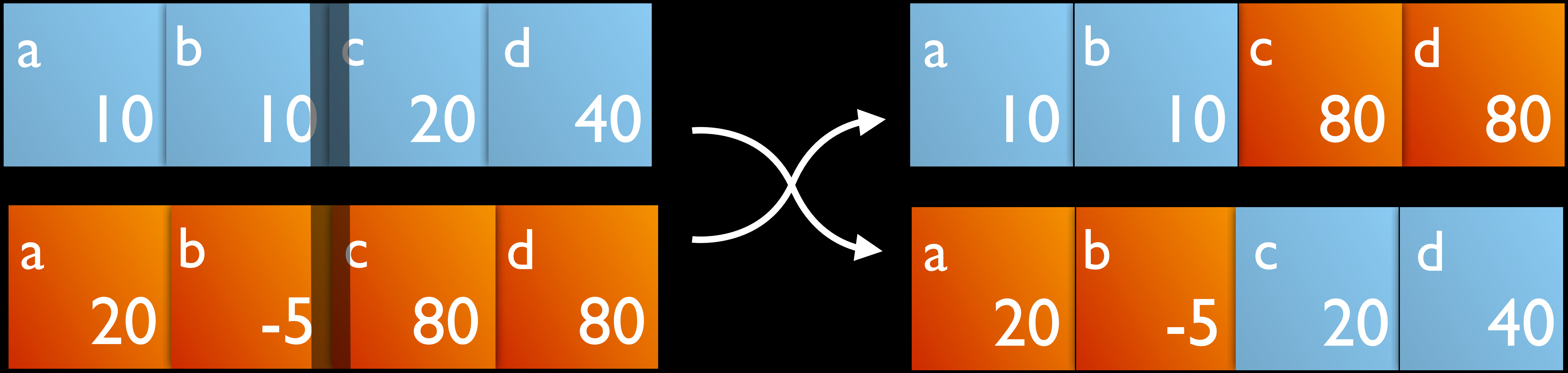
```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

       if (c == d) {

           // branch we want to execute

       }
   }

   ...
```



Crossover



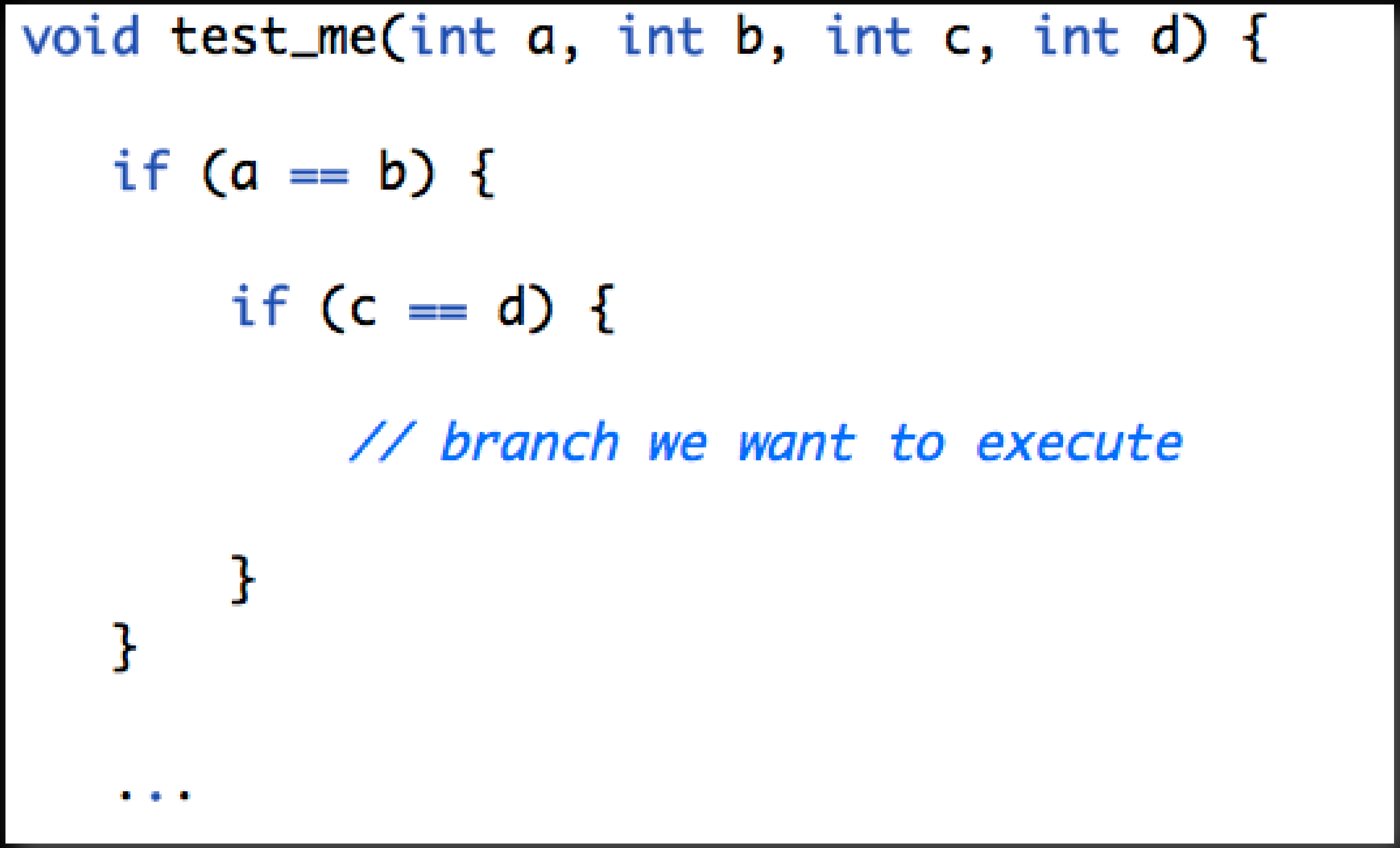
```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

       if (c == d) {

           // branch we want to execute

       }
   }

   ...
```



a	b	c	d
10	10	20	40
20	-5	80	80

a	b	c	d
10	10	80	80
20	-5	20	40

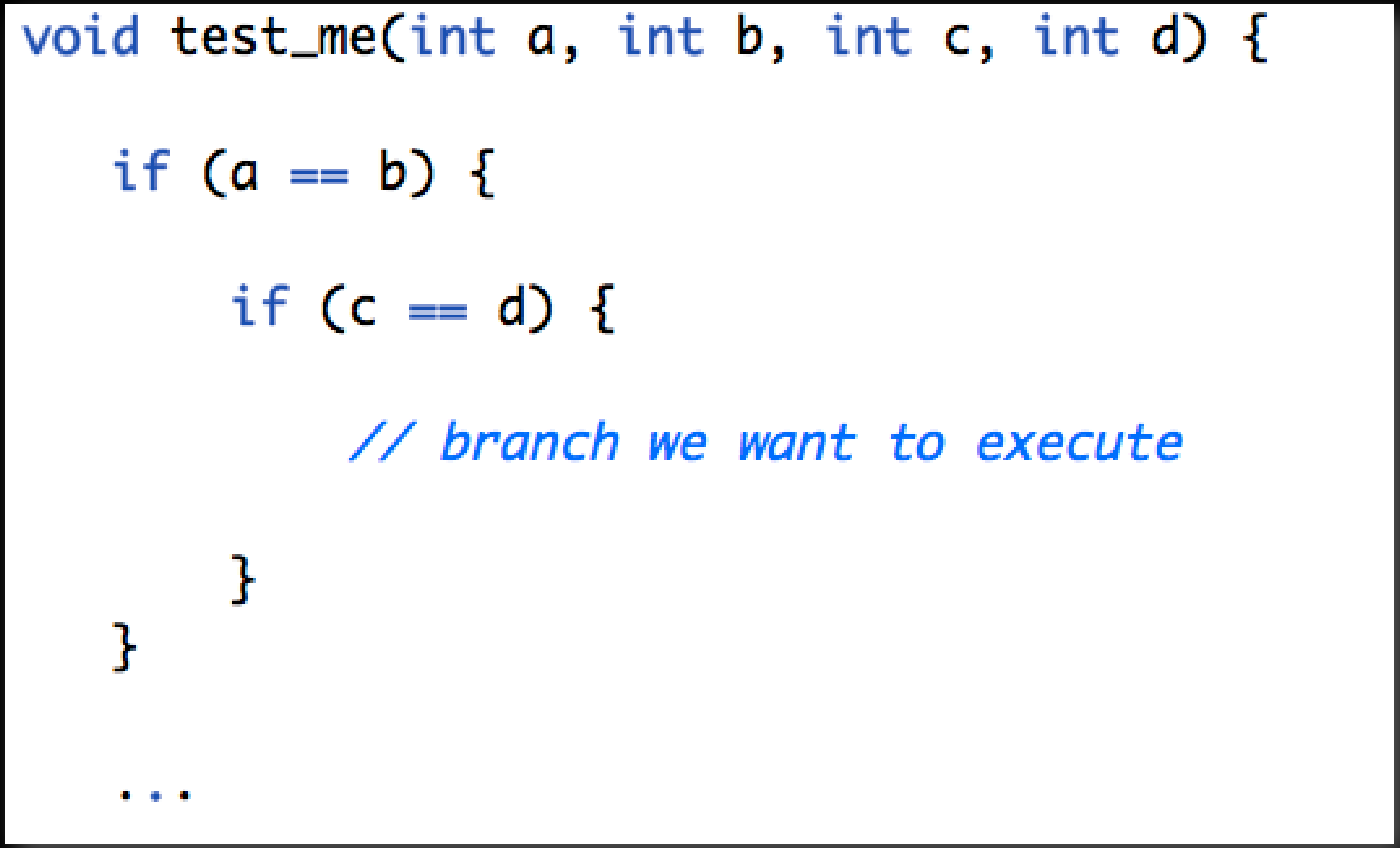
Mutation



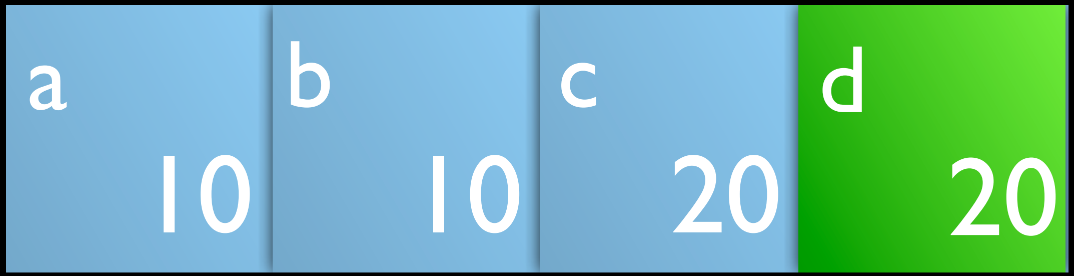
```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

      if (c == d) {

         // branch we want to execute

      }
   }

   ...
```

a	b	c	d
10	10	20	40

Mutation

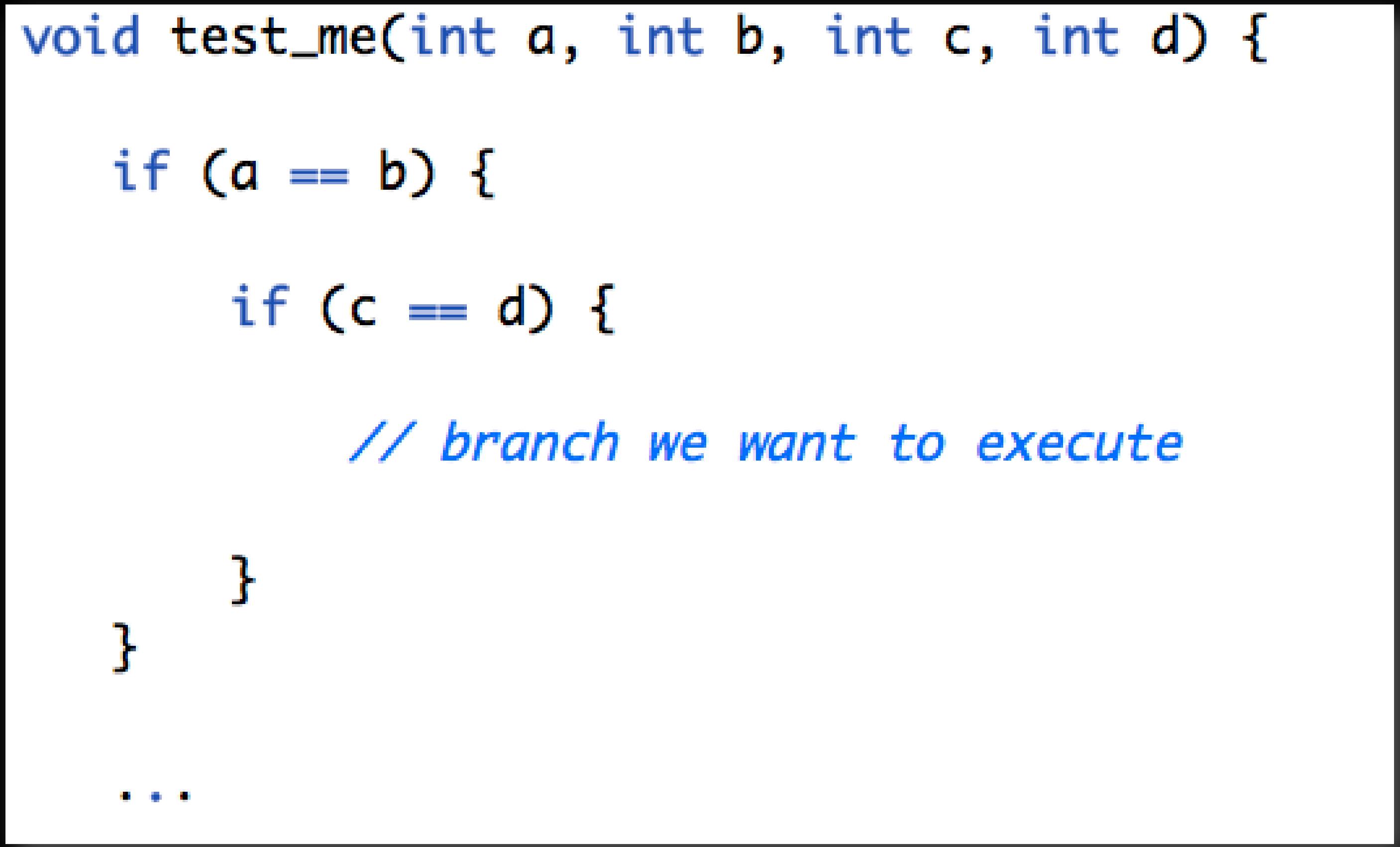


```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

      if (c == d) {

         // branch we want to execute

      }
   }

   ...
```



a	b	c	d
10	10	20	20

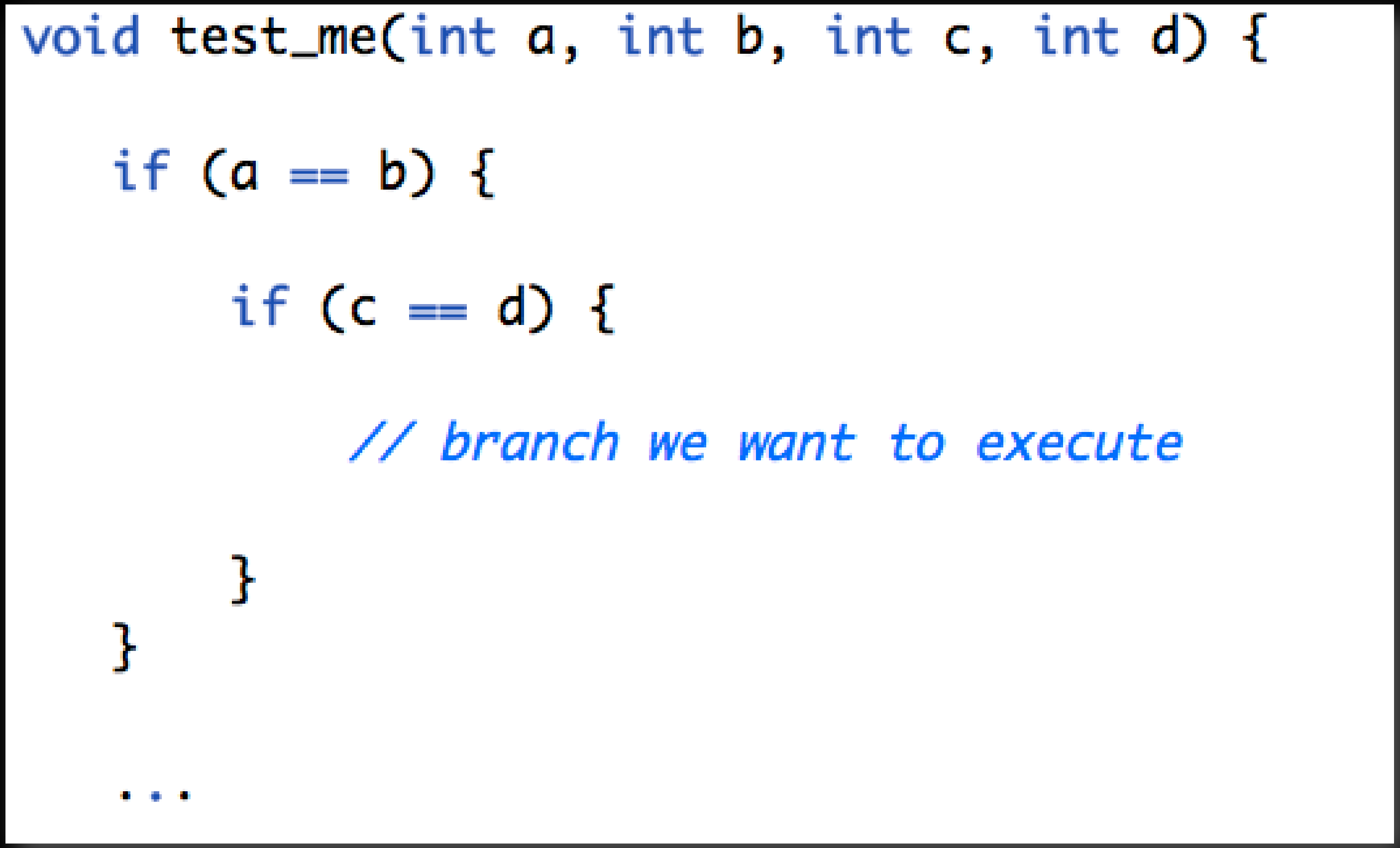
Mutation



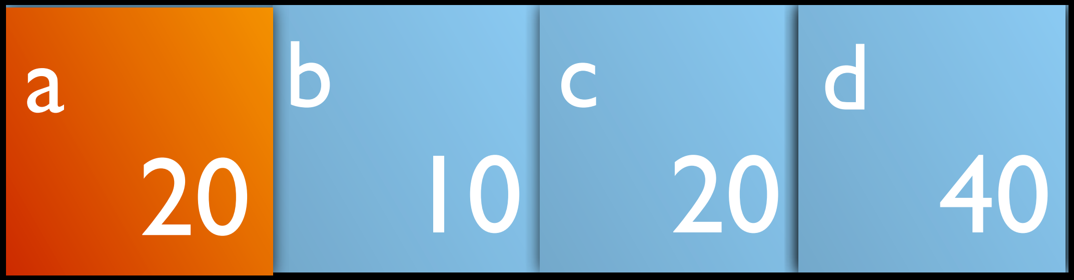
```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

      if (c == d) {

         // branch we want to execute

      }
   }

   ...
```

a	b	c	d
10	10	20	40

Mutation



```
void test_me(int a, int b, int c, int d) {

  if (a == b) {

     if (c == d) {

        // branch we want to execute

     }
  }

  ...
```



a	b	c	d
20	10	20	40