Software is Not Fragile

CS-DC'15 World e-conference

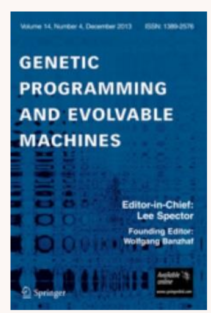
1 October 2015 7:10 (UTC)

W. B. Langdon

Department of Computer Science

UCL

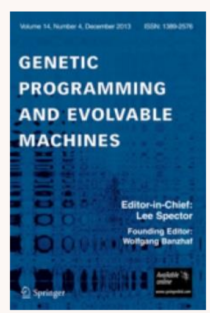
Genetic Improvement special issue of
Genetic Programming and Evolvable Machines,
deadline 19 December 2015



Software is Not Fragile

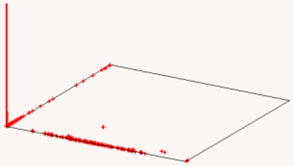
W. B. Langdon
Department of Computer Science
UCL

Genetic Improvement special issue of
Genetic Programming and Evolvable Machines,
deadline 19 December 2015

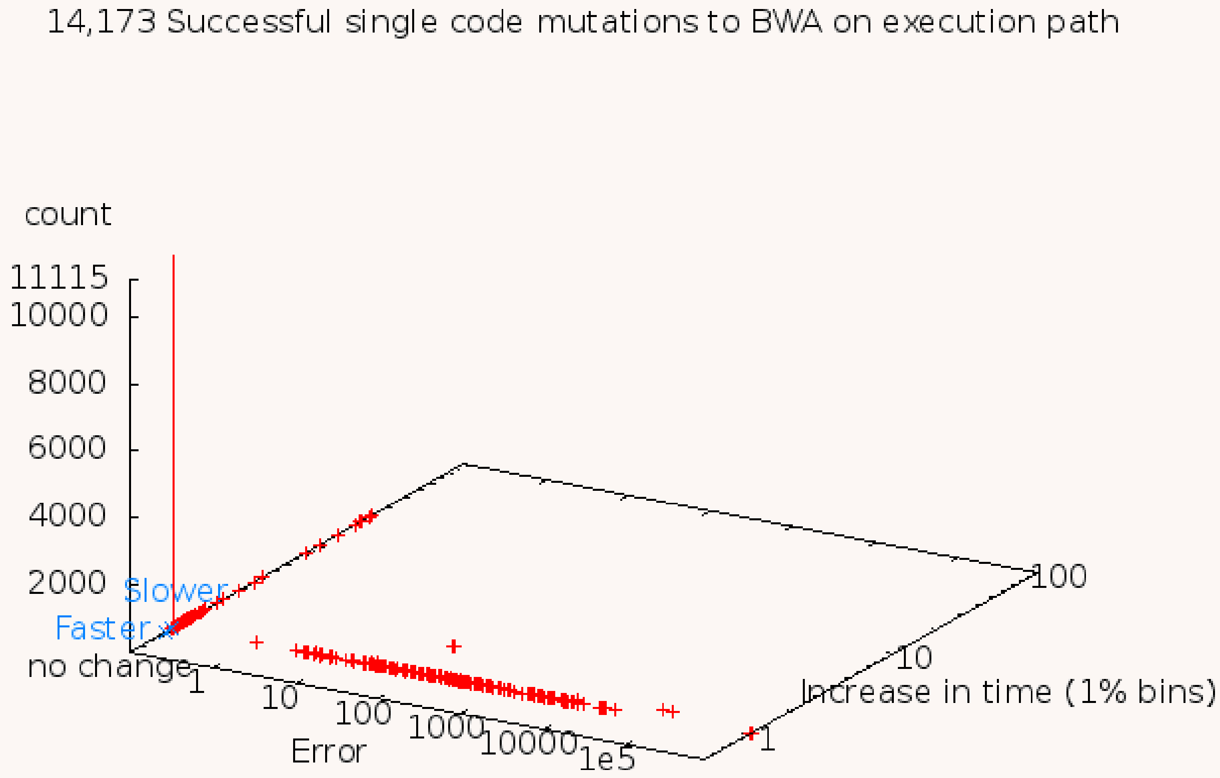


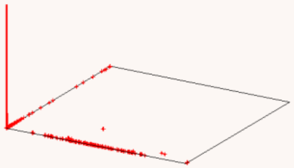
Software is Not Fragile

- Introduction
  - Software not broken by random change
  - Software as a Complex System
  - Zipf distribution in software engineering

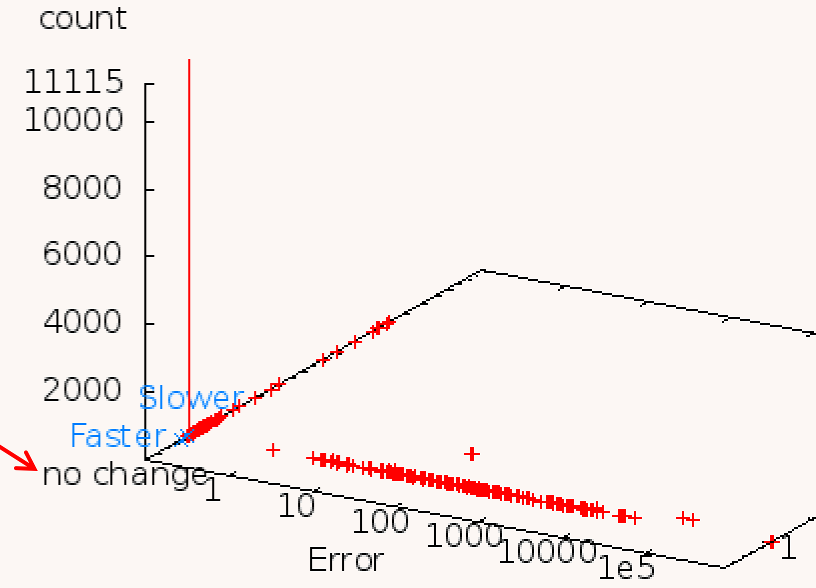


89% Mutations make no change





89% Mutations make no change



What does graph show?

Take C program

Find lines of code in use

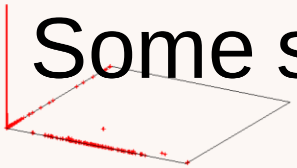
Mutate all of them

Plot change in output(error) v. change in speed

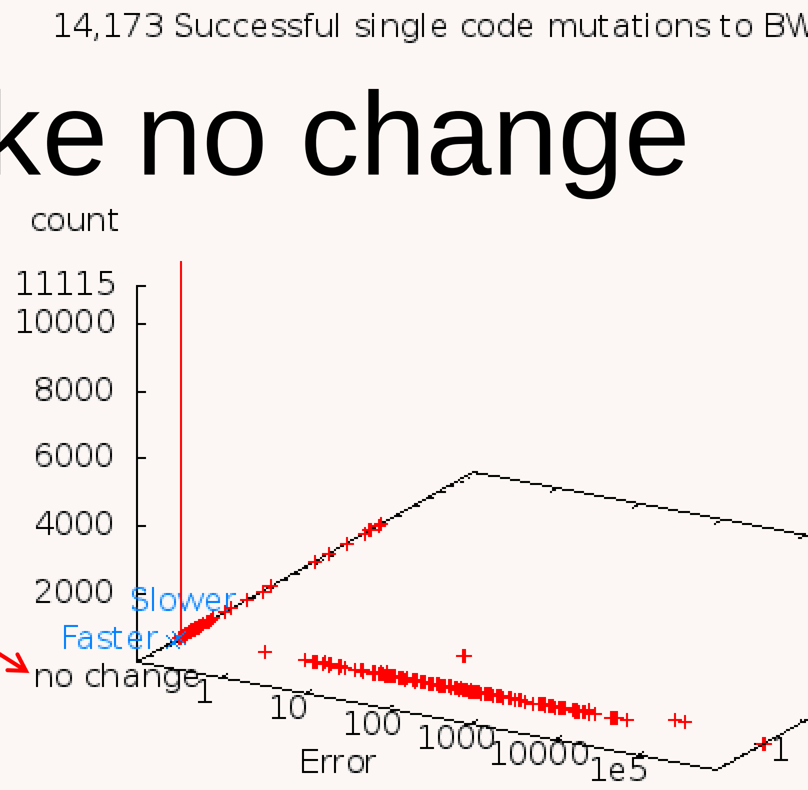
Many fail to compile. Some abort at run time

Most which run give **exactly** the same answer

Some slower, a few faster



89% Mutations make no change



What does graph show?

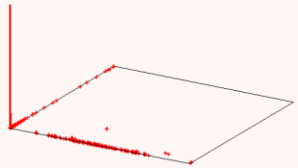
BWA 13000 lines of C

Mature 5 years, 28 release

State of the art code cited 1662 times

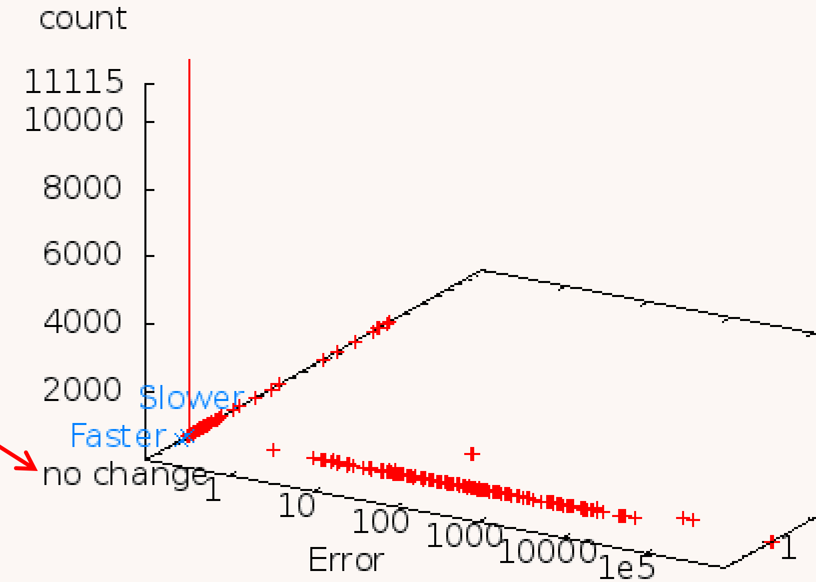
One test case

GNU gcov test coverage tool gives lines used



89% Mutations make no change

What does graph show?



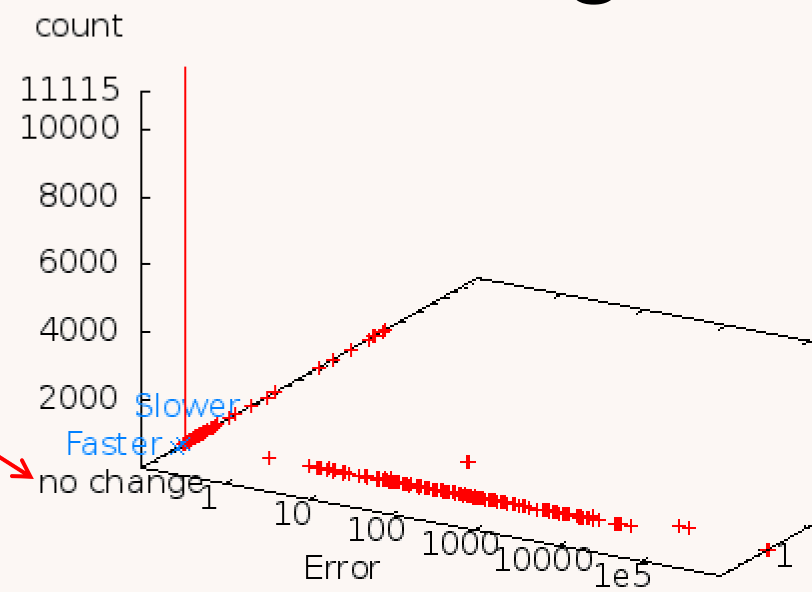
Mutations like those a human might make

All deal with existing lines of C source code

1. Delete line of code
2. Replace line of code
3. Insert copy of line of code before target

14,173 Successful single code mutations to BW

89% Mutations make no change



What does graph show?

532 lines of code used

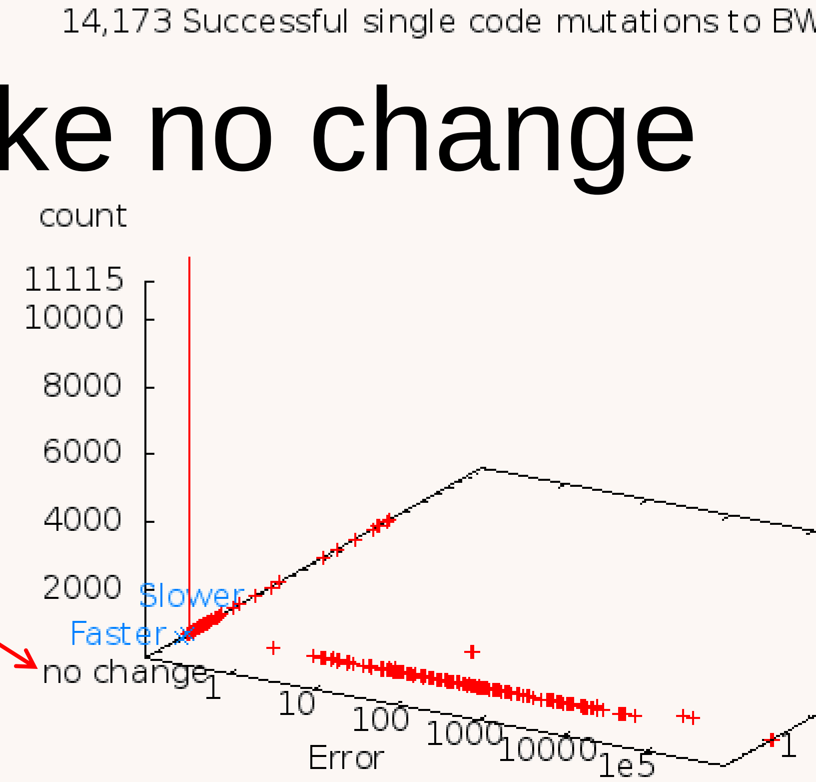
For each create mutants to

1. delete it
2. replace it with every other line of the same type in the same source file
3. insert every other line of same type/same file in front of it

For BWA there are 61775 possible mutations

89% Mutations make no change

What does graph show?



For BWA there are 61775 possible mutations

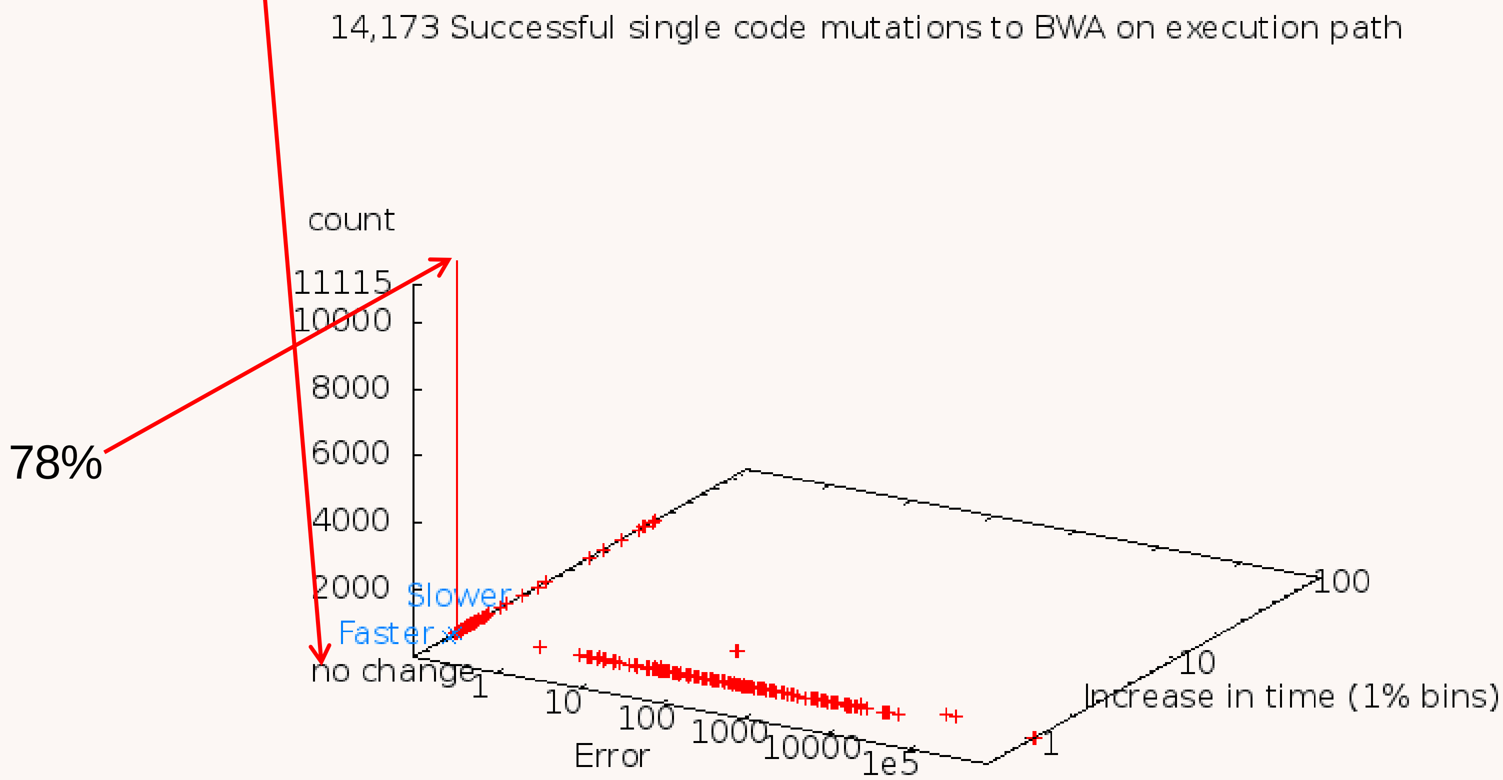
Try them all

For BWA 37% compile

14% abort, eg seg fault, CPU limit exceeded

Effect of 14173 mutations which run ok plotted

89% Mutations make no change



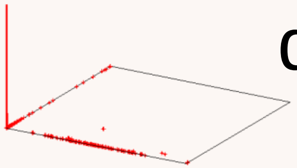
Effect of 14173 mutations which run ok plotted

89% give **exactly** the same answer(nochange)

Some slower, a few faster

Create Mutants which Compile

- Almost all compilation errors are due to mutation moving variables out of scope.
- If a mutant compiles it is liable to run, terminate and produce a reasonable output
- To find mutants which compile is easy:
    1. Use a more relaxed (than C) language
    2. Keep careful track of variable scope
    3. Try several times and let compiler discard duds



Example Mutation

`<_bwtaln.c_228><_bwtaln.c_300>`

Replace line 228 of file bwtaln.c with line 300

```
bwt_destroy(bwt);
```

original code (line 228)

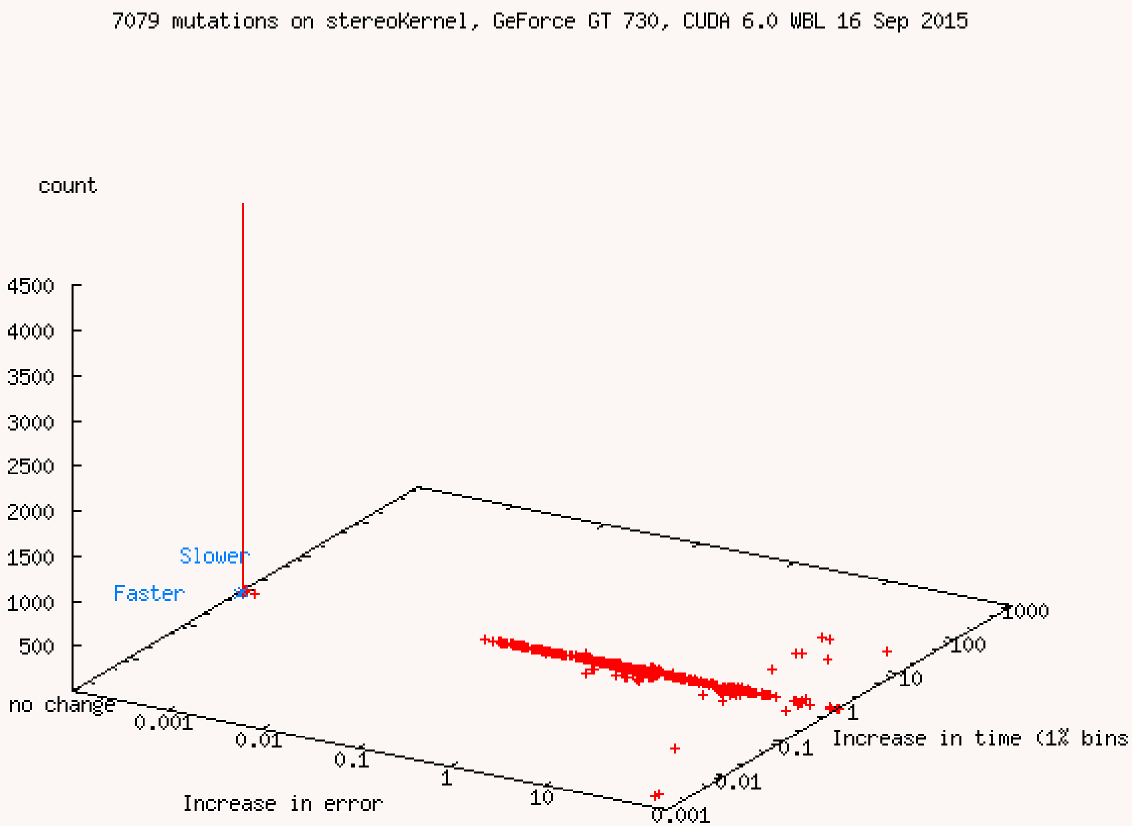
```
fprintf(stderr, " -0 use single-end reads only
(effective with -b)\n");
```

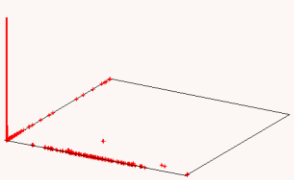
New code

- New code is 9ms faster
- It no longer frees bwt instead writing to log
- Logging seems to be cheap
- Freeing bwt not needed as program is about to stop

9ms is significant: tot=224, 107 compile, n+ 82, n- 19, p=6.89 10^{-10}

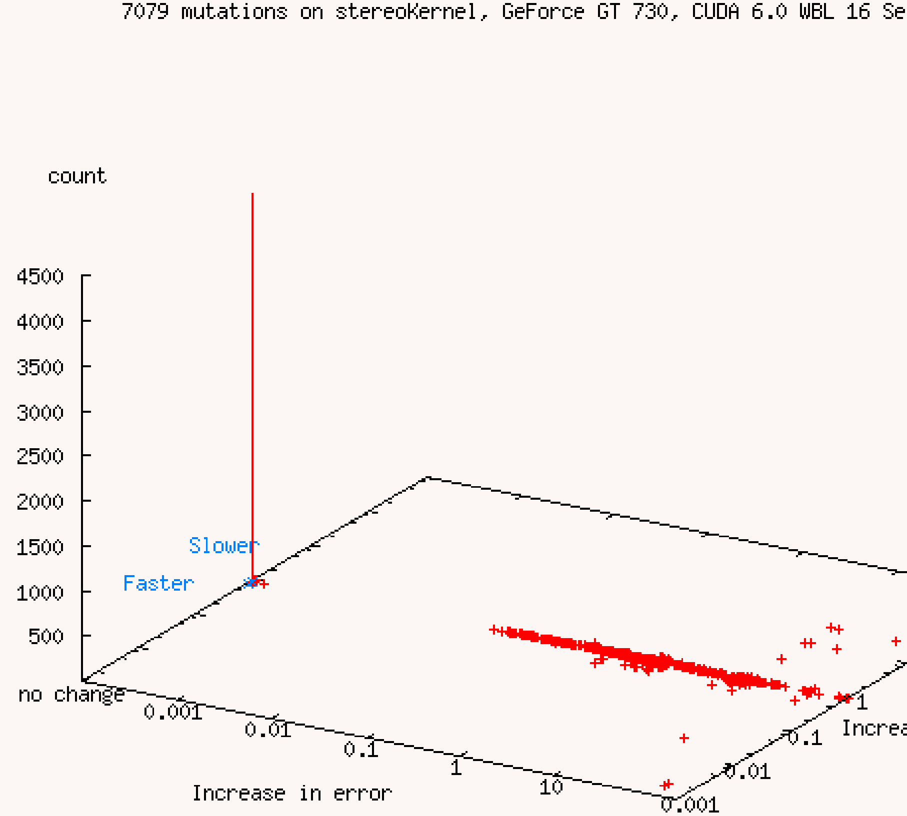
62% StereoCamera Mutations make no change





StereoCamera Mutations

What does graph show?



276 lines of CUDA

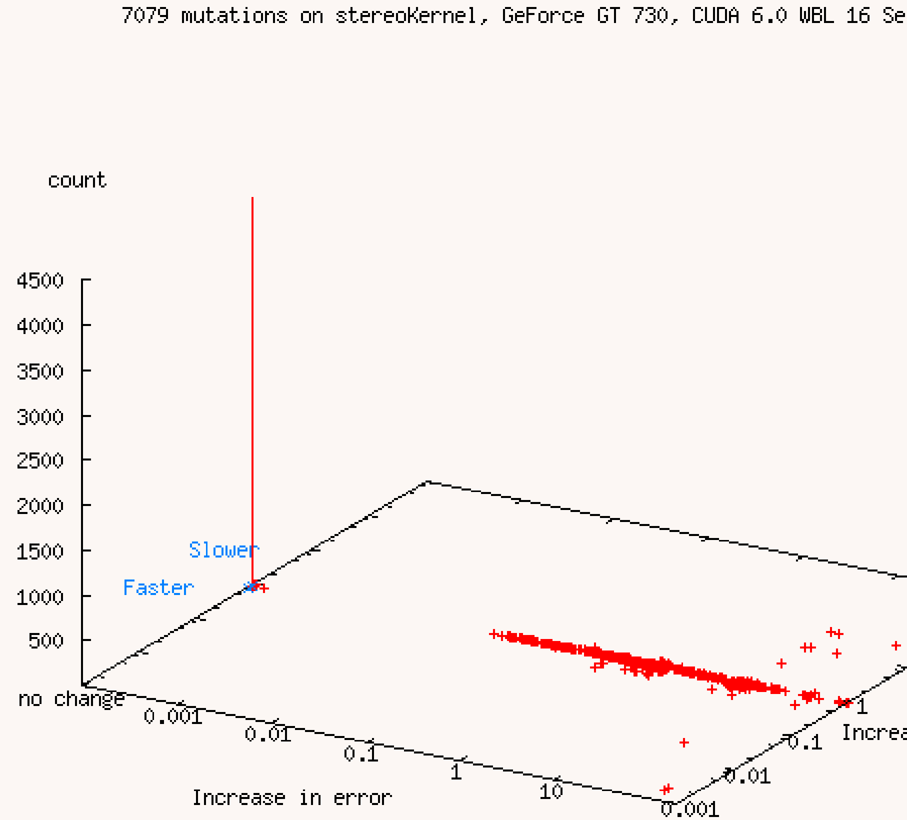
All used

Do all mutations

Plot change in output(error) v. change in speed

StereoCamera Mutations

What does graph show?



All (7070) compile

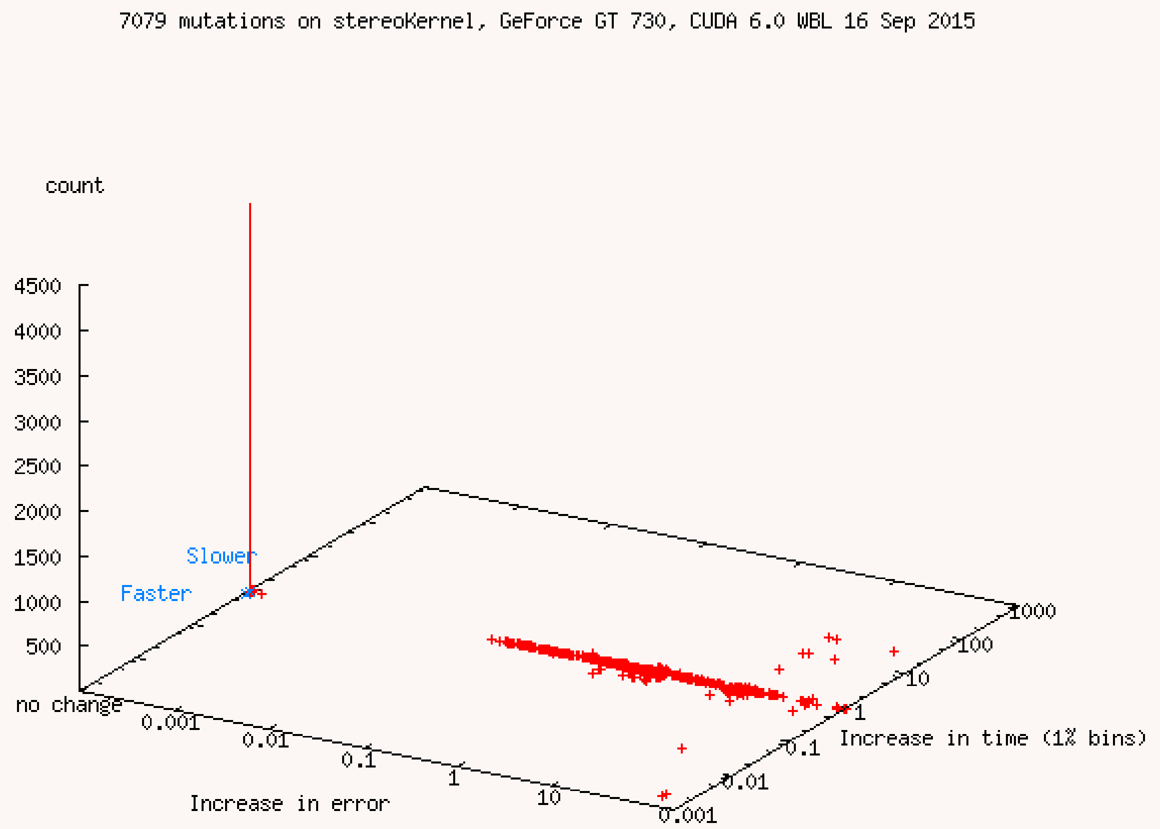
All stop

334 abort at run time

4400 give **exactly** the same answer(nochange)

Some slower, 41 faster

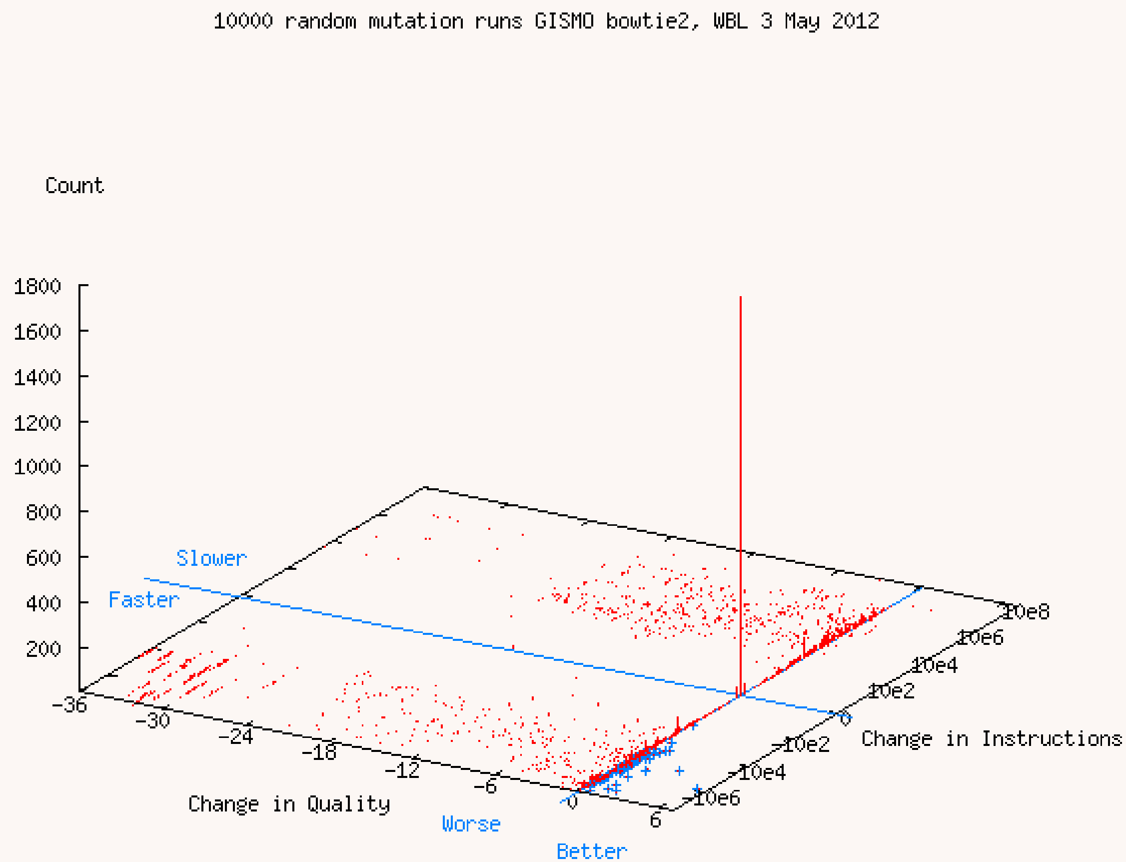
62% StereoCamera Mutations make no change



4400 give **exactly** the same answer(nochange)

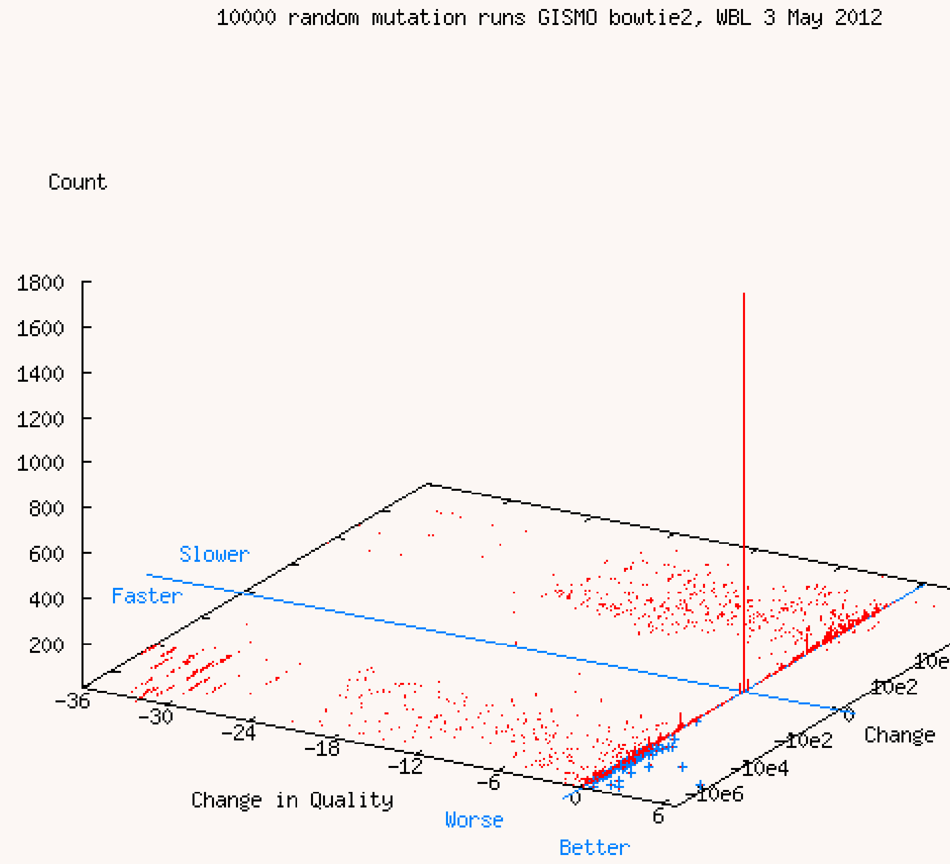
Some slower, 41 faster

Bowtie2 Mutations



Bowtie2 Mutations

What does graph show?



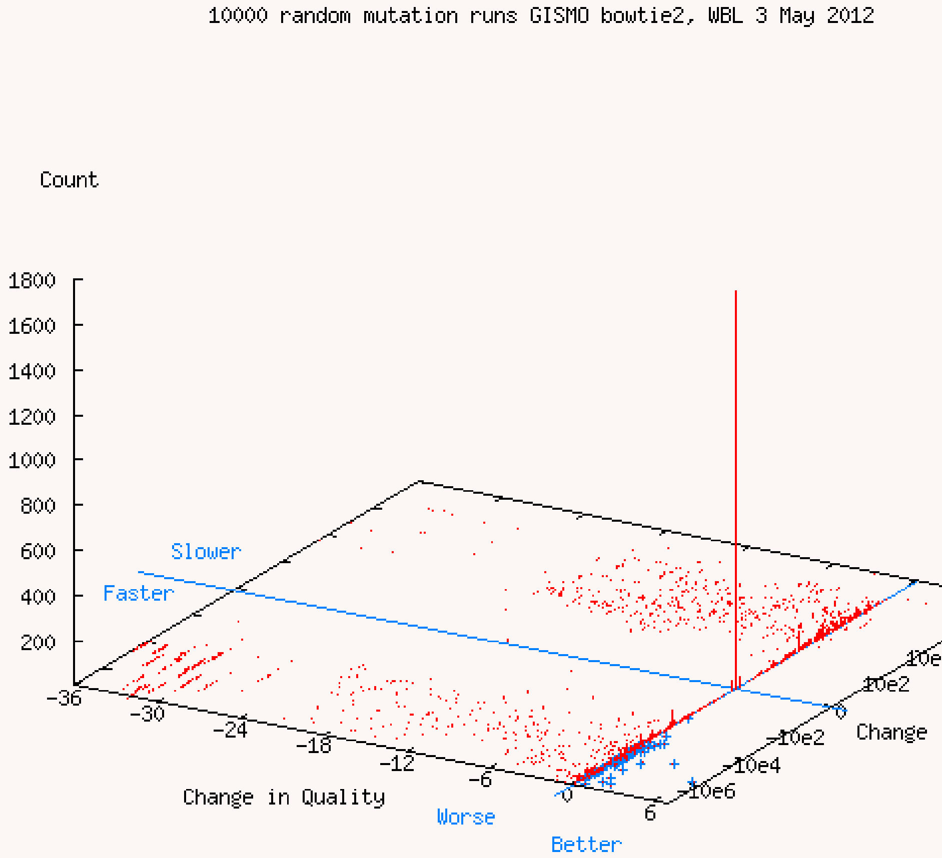
Bowtie2 is state of the art Bioinformatics tool written by an expert.

Cited by 2444.

50000 lines of C++. 106 source files.

Bowtie2 Mutations

What does graph show?



Previous [analysis](#)

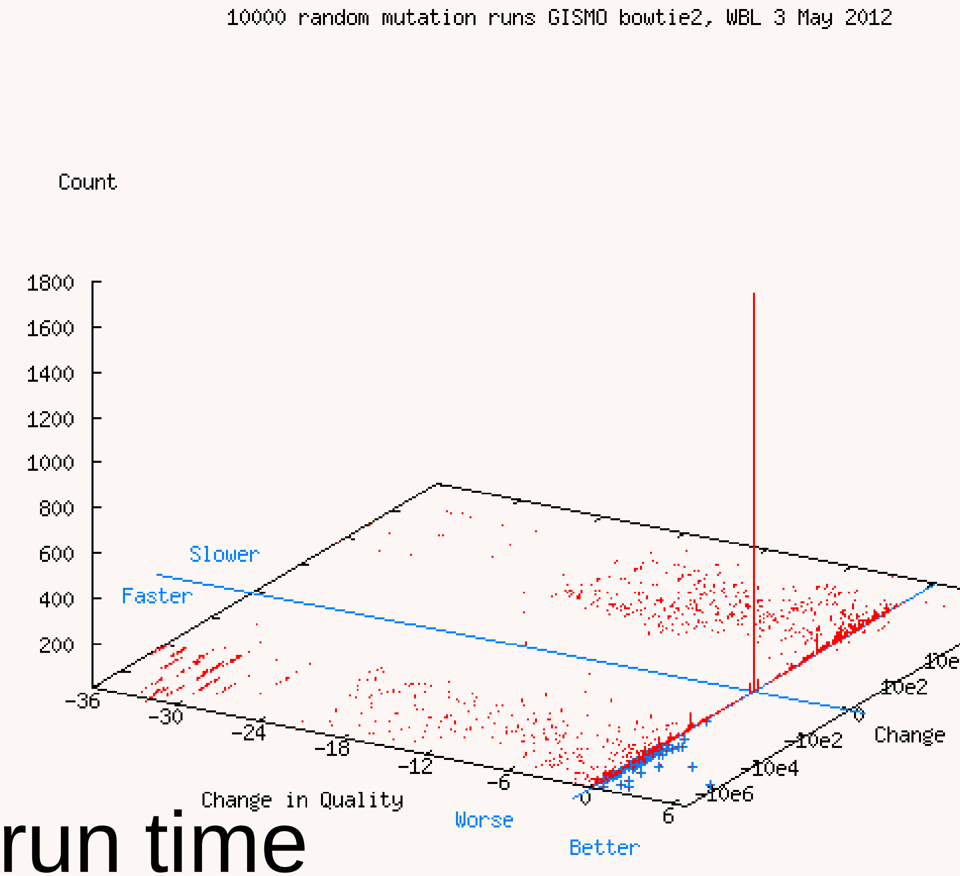
showed which lines

are heavily used.

These are targeted by mutation. Take a random sample of possible mutations.

Plot change in output(error) v. change in speed

Bowtie2 Mutations



What does graph show?

Many do not compile

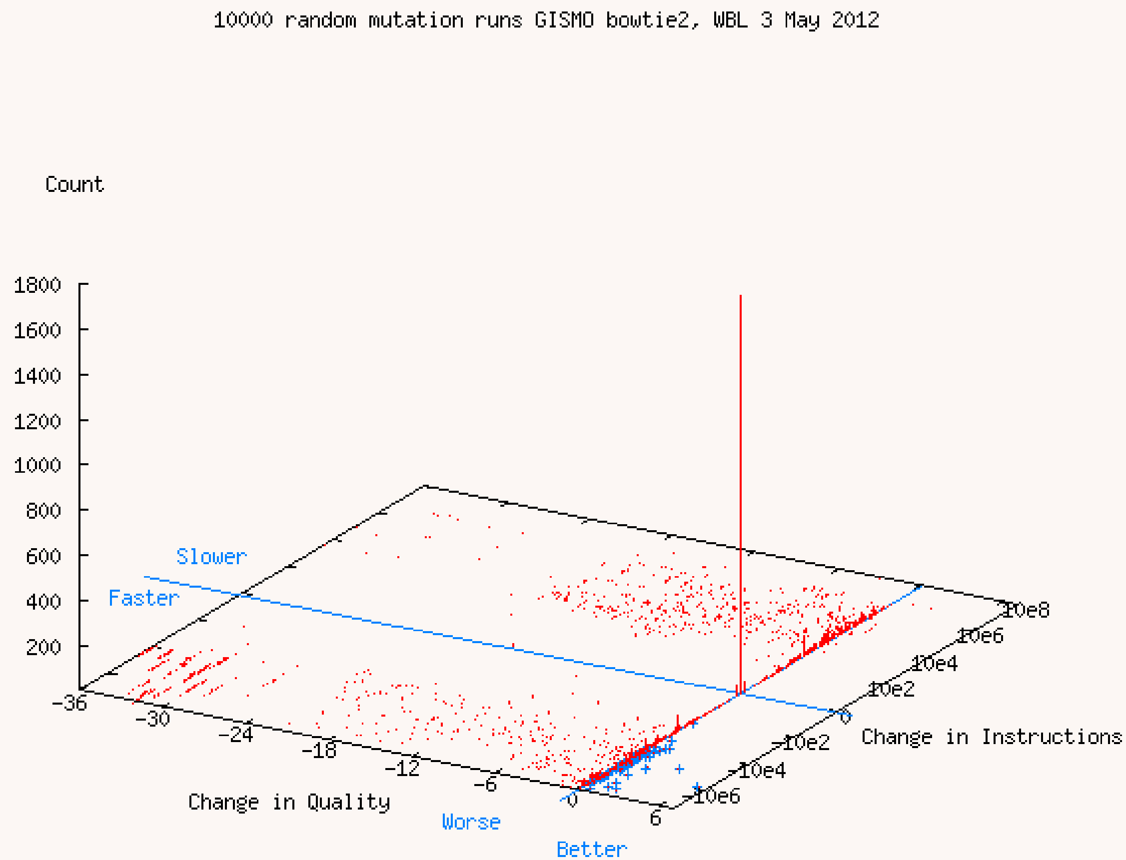
Some mutants abort at run time

Some produced very poor quality output

Most which run give **exactly** the same answer

Some slower, a few faster

Bowtie2 Mutations



Most which run give **exactly** the same answer

Some slower, a few faster. 139 better.

Software as a Complex System

- Up to 5.5 billion transistors in CPU chip
- A billion PCs, 1+ billion phones
  - 10^{19} transistors?
- 4.9 billion lines of opensource code (2009)
- 86.06 billion neurons in human brain
  - Up to 10^{15} synapses per person
- Items in London's water system, 0.4 10^{9} ?

How big are Complex Systems?

- Computing is big (10^{19}? transistors in use) but is it complex?
- Estimate half a billion items in London's water system.
  - No one person understands all London's pipes
  - No team of people understand them
  - Not even the whole of the population of London understand their water pipes.
  - Yet we still use them
- Plumbing on this scale is a complex system

Software as a Complex System

- Software engineering has produced some of the most complex systems on the planet
  - Software systems may exceed a million lines
  - No one person understands 10^6 lines of code
  - No team of people understand it
  - Not even the team of experts who wrote it
  - Yet we still use them
- Software is beyond comprehension.

Software as a Complex System

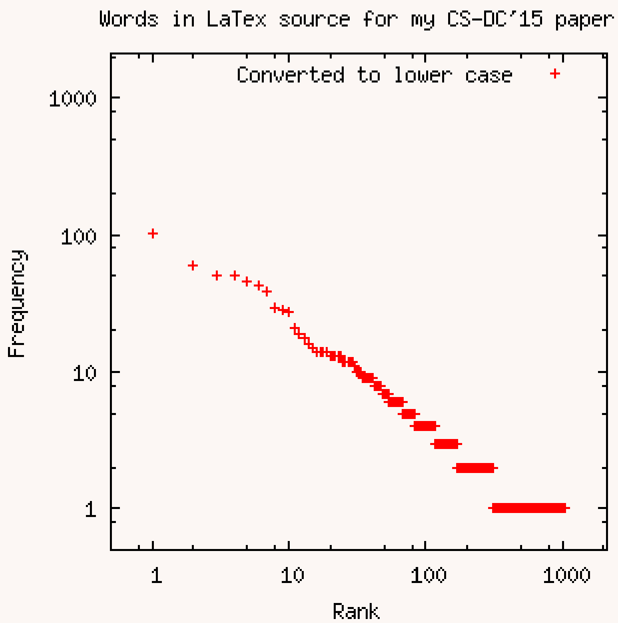
- Large software systems contain bugs
- The only practical way of verifying them is testing
- Yet testing can never be complete
- Computing not so much better software as:
- Computing needs ways to live with imperfections

Software as a Complex System

- Software is beyond comprehension
- The richest man in the world owes his wealth to software (not computer hardware, not oil, not steel, not coal).
- Software yields enormous economic advantages
- The world is addicted to software

Zipf's Law

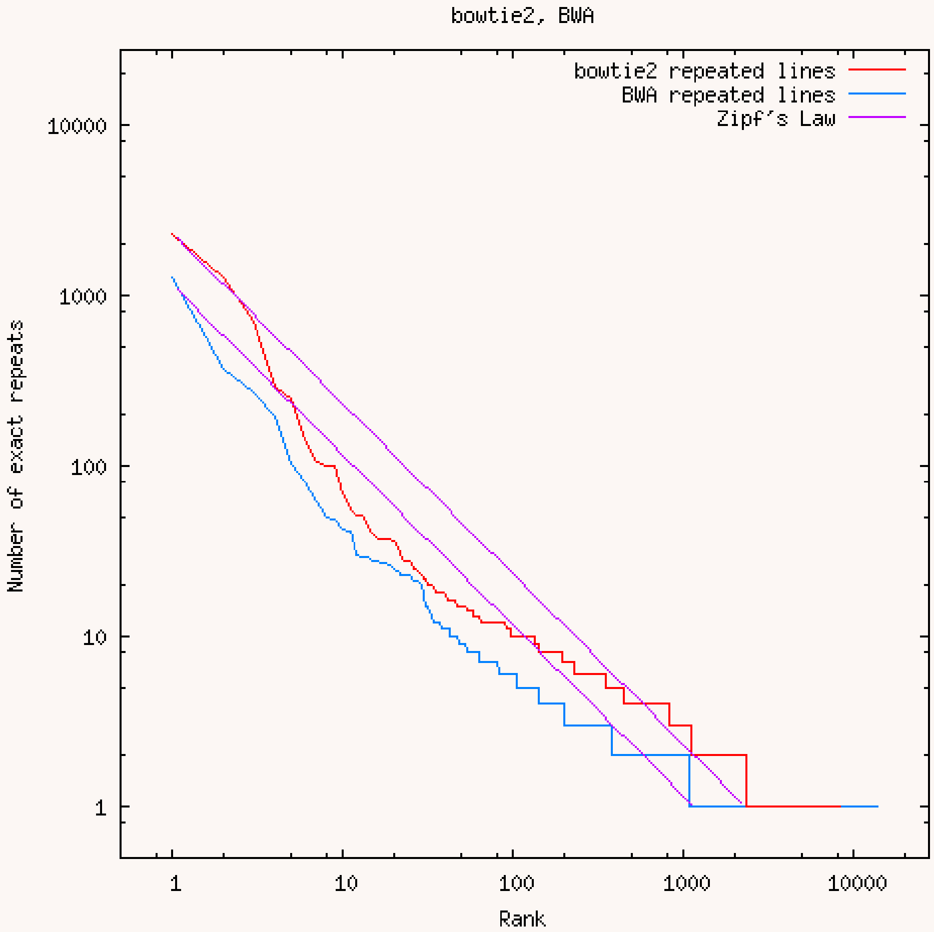
- George Zipf proposed his "universal" law
- Eg order english words by their frequency
- Plot frequency v. position in list (rank)
- The graph must be monotonic, but
- If plot on log-log typically get a straight line with a slope of -1.



Zipf law for code

- Used BNF grammar as convenient way to strip comments and convert to standard white space.
- Bowtie2 and BWA

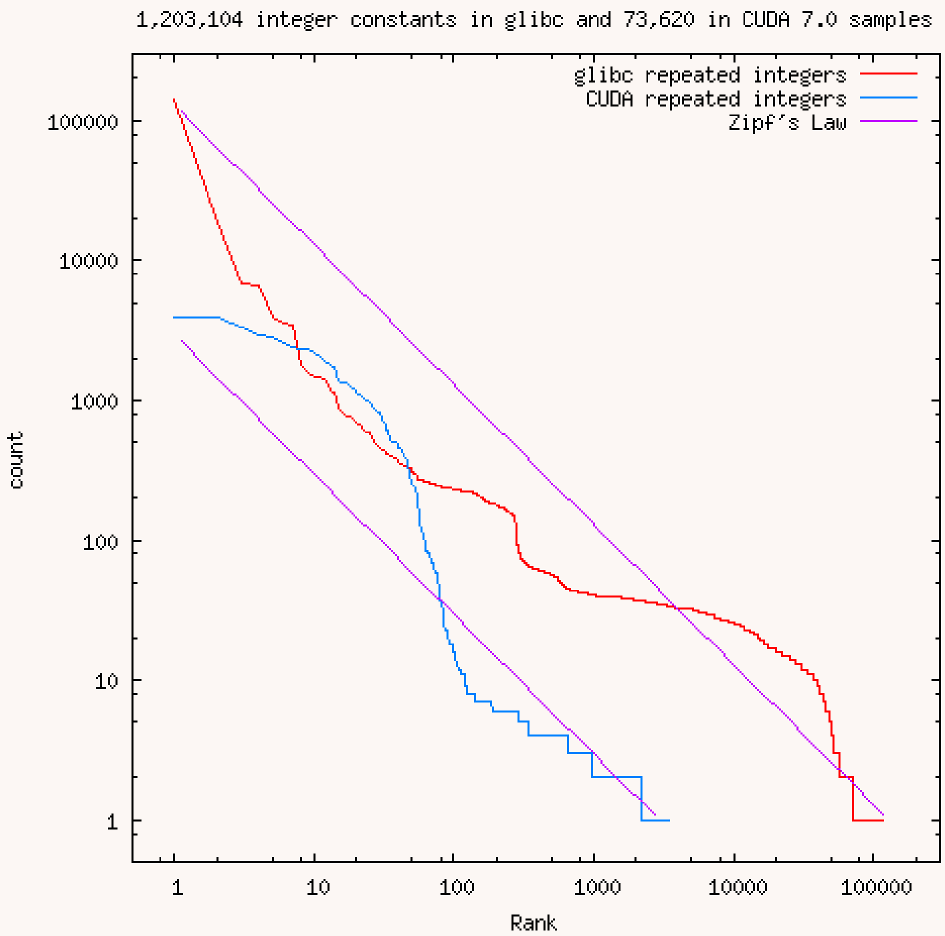
Repeated lines of code



Zipf for integer constants

- Extract integer constants from
  - The GNU C library (excluding test suite) 845,360 lines of code
  - All of the nVidia CUDA 7.0 sample code 85,711 lines of code
- Convert to hex and octal to decimal
- Remove leading zeros

Repeated lines of code



Repeated lines of code

- 0 most common in both cases
- 1 2nd in glibc but 32 2nd in CUDA
- Few negative numbers, especially in CUDA

Six impossible things before breakfast

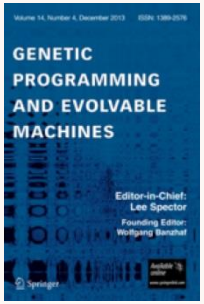
- To have impact do something considered impossible.
- If you believe software is fragile you will not only be wrong but shut out the possibility of mutating it into something better.
- Genetic Improvement has repeatedly shown mutation need not be disastrous and can lead to great things.

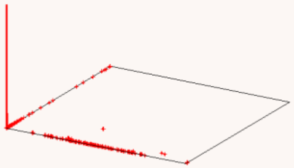
Conclusions

- Zipf law can apply to constants as well as code
  - May be useful in both test and code generation
- Software is complex but addictive
- Many mutants do not break the code
- Software is not fragile

break it, bend it, Evolve it

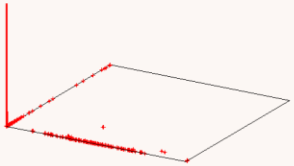
GI special issue
GP+EM
deadline 19 Dec





END

http://www.cs.ucl.ac.uk/staff/W.Langdon/ http://www.epsrc.ac.uk/

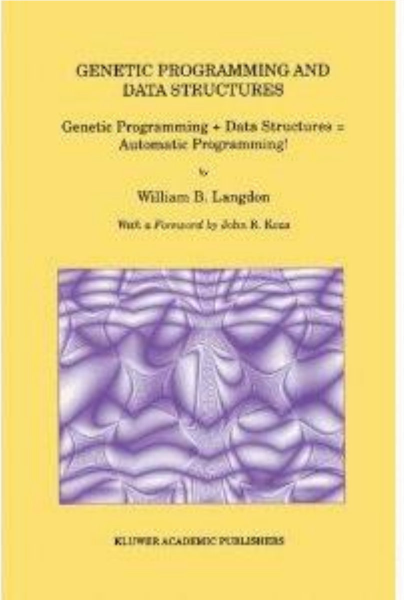


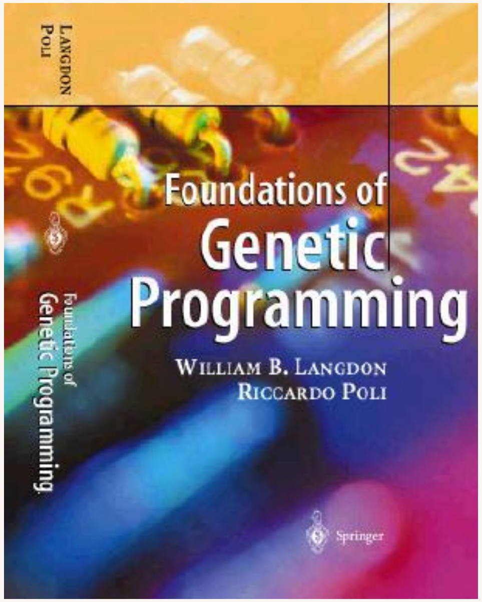
Genetic Improvement

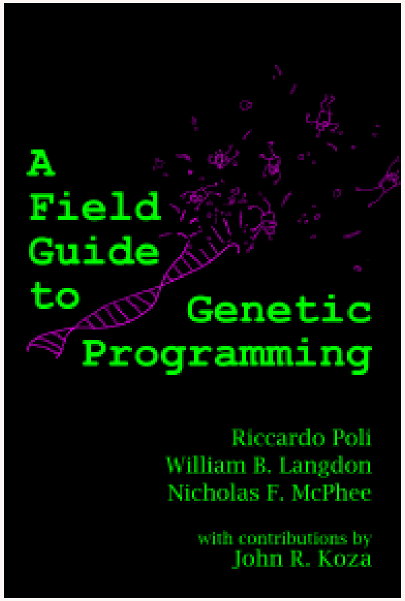
W. B. Langdon

CREST

Department of Computer Science







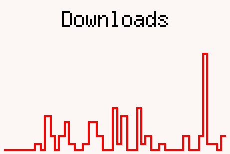
The Genetic Programming Bibliography

http://www.cs.bham.ac.uk/~wbl/biblio/

10454 references

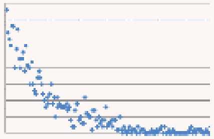
RSS Support available through the Collection of CS Bibliographies. XML RSS

A web form for adding your entries. Co-authorship community. Downloads



A personalised list of every author's GP publications.

blog.html



Search the GP Bibliography at
http://liinwww.ira.uka.de/bibliography/Ai/genetic.programming.html