Deep Mutations have Little Impact

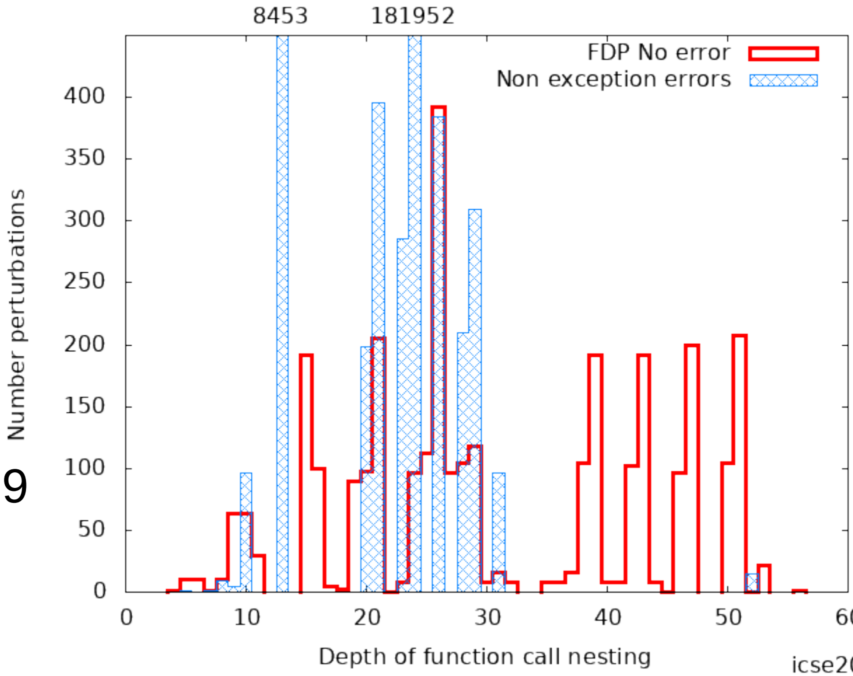
13th International Workshop on Genetic Improvement
Tuesday 16 April 2024

Humies $10000 prizes
Submit by **Friday 31 May**

W. B. Langdon, UCL

Information theory suggests
for most deeply nested mutations
disruption fails to propagate to the output.

W.B. Langdon and D. Clark. In GI@ICSE
2024 Lisbon, DOI:10.1145/3643692.3648259



Deep Mutations have Little Impact

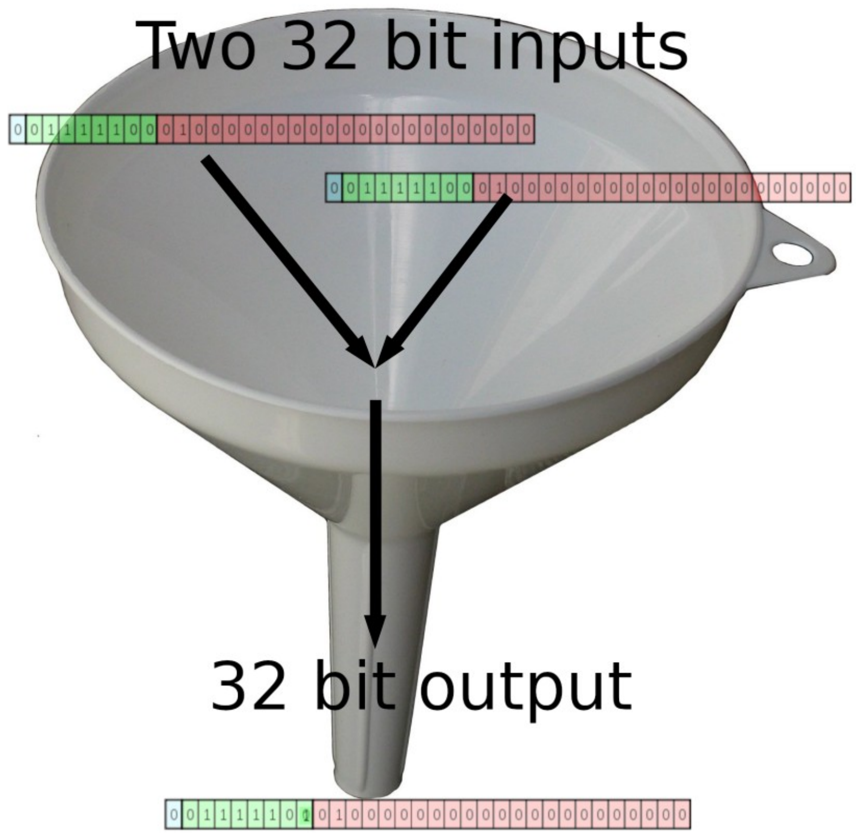
- What is the PIE (Propagation Infection Execution) view of software bugs
  - How PIE leads to Failed Disruption Propagation and
  - Information Theory idea of software robustness.
- Information Theory says impact of disruptions lost with distance when nested
- Deep mutations have little impact in pure functions
  - Genetic programming deeply nested trees
- Traditional imperative software, C++, with side effects, including global variables
- Evidence that deep mutations have little impact in C++
- What is Magpie? Tutorial after lunch
- What is PARSEC, VIPS, C++ vipsthumbnail benchmark
- Implications:
  - unit testing v. system testing, flaky tests
  - Software code bloat, depth important
  - Software robustness: deeper harder to test, improve, optimise, repair?

Voas' PIE and Silent Bugs, Information Theory, Robust Deeply Nested Software

- PIE framework suggested by Jeffrey M. Voas and Keith W. Miller, 1995:
  - For a software bug buried in code to have impact, it must:
    - be **E**xecuted
    - it must change (**I**nfect) the program's state
    - the state change must **P**ropagate from the bug to some externally visible point (eg a print statement)
- Expand to consider any type of **D**isruption, e.g. radiation, power glitch and mutations.
- **F**ailure of **D**isruption to **P**ropagate to output, means software is robust
- Information Theory helps to explain why **FDP** is common and software is often robust
- Initial experiments suggest **FDP** more common in **deeply nested** software

Information Funnel

Computer operators are irreversible. Meaning input state cannot be inferred from outputs. Information is lost

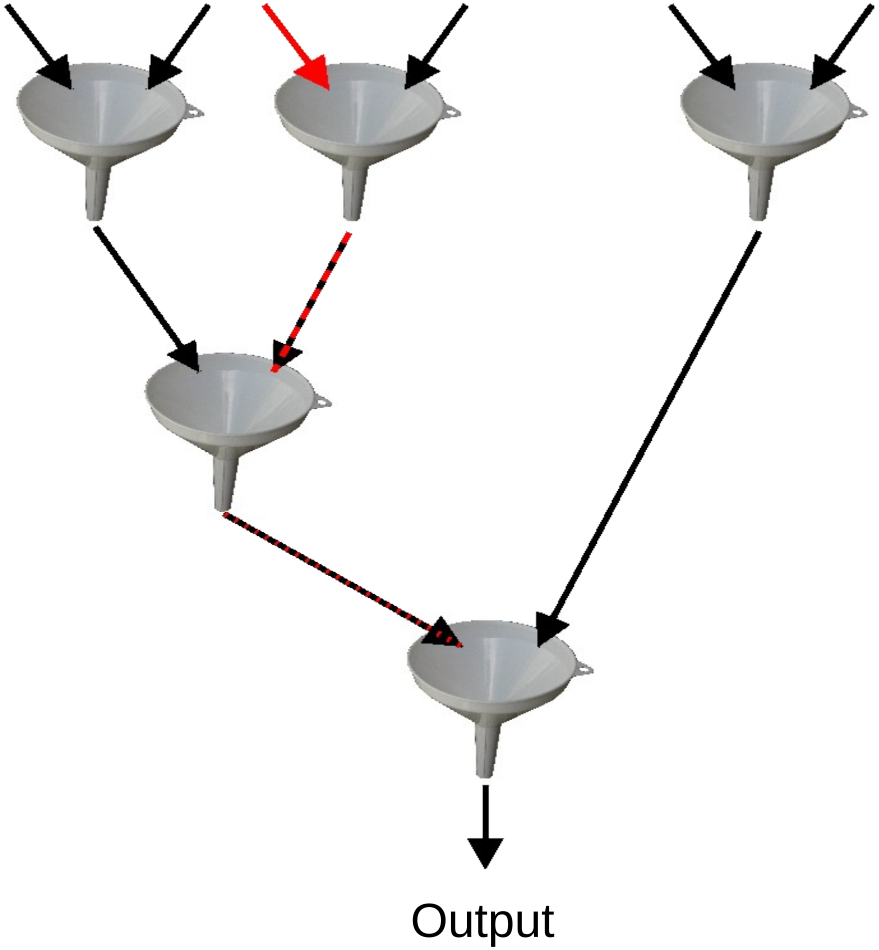


Information funnel

More information enters than leaves

Information flow in five nested functions

Potential information loss at each (irreversible) function



Disruption may fail to reach reach output.

(No side effects.)

Entropy = Information content

- Simple example, function = addition, inputs random 0-9 digits
- 1 digit mean 4.5, standard deviation $\sigma = \sqrt{8.25}$ entropy=$\log_2 10$
- n digit mean 4.5 n, $\sigma = (n\ 8.25)^{½}$,
  - large n distribution tends to Gaussian entropy=$2.047+\log_2\sigma$
  - I.e, information content falls from 3.3n to $3.6+\log_2(n)/2$
  - Adding many digits loses almost all the information
  - Impossible to infer inputs from their sum

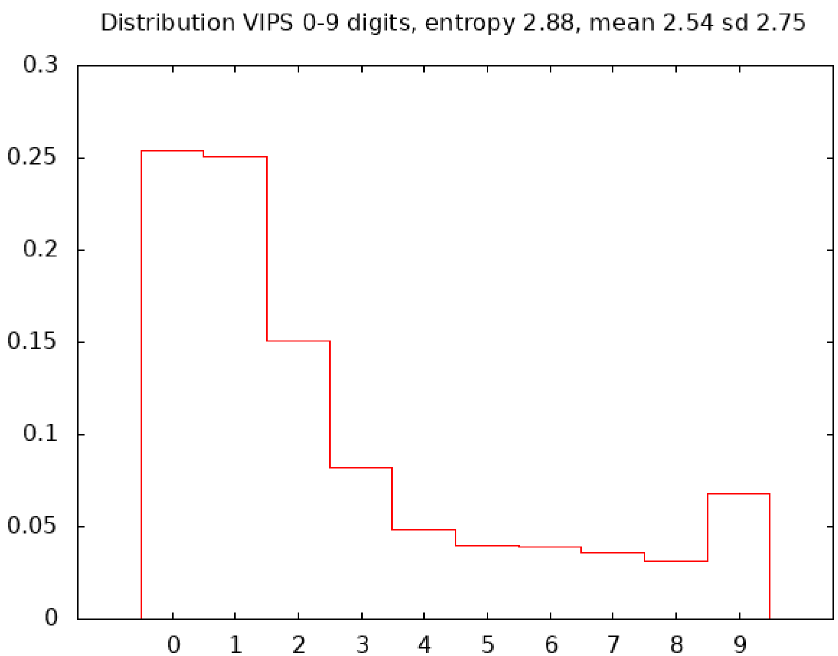
Number inputs	mean	sd σ	entropy	Gaussian entropy	Information loss
1	4.5	2.9	3.3	3.6	0%
2	9.0	4.1	4.0	4.1	39%
3	13.5	5.0	4.4	4.4	56%
4	18.0	5.7	4.6	4.6	66%
5	22.5	6.4	4.7	4.7	72%
n	4.5n	$\sqrt{(8.25n)}$		$2+\log_2\sqrt{(8.25n)}^{½}$	$< 100\% = 1 - 2/(3.3n) - (1/3.3n)\log_2\sqrt{(8.25n)}^{½}$

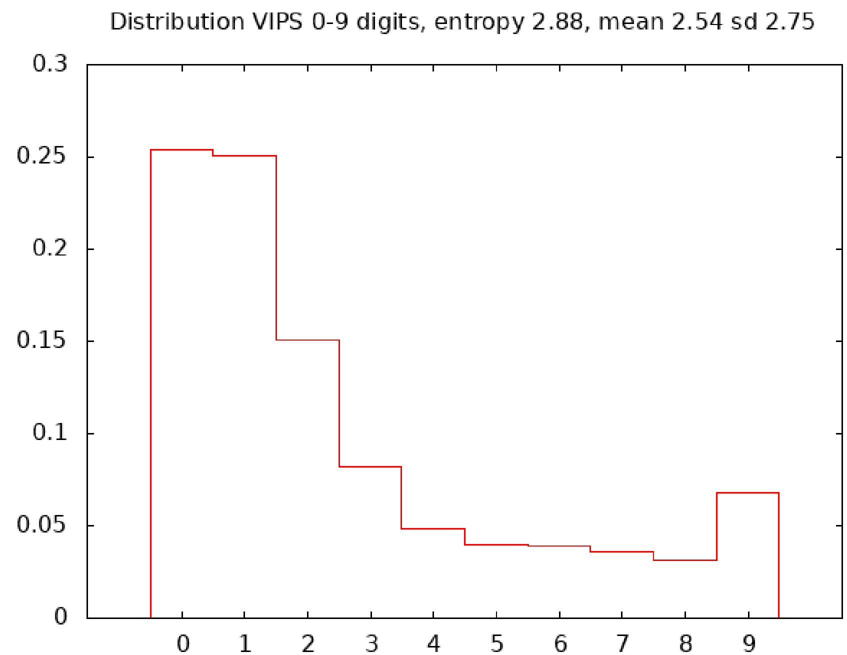
Entropy lost when adding digits

Add n non-uniformly distributed digits (0-9, left)

quickly converges on Gaussian.

Addition not reversible(given output do not know inputs) addition losses information (entropy)





Entropy falls from 2.88n in inputs to $\approx 2+\log_2(7.7n)/2$ in output

http://www.cs.ucl.ac.uk/staff/W.Langdon/icse2024/langdon_2024_GI/add10.gif

Evolve Deep Integer GP Trees

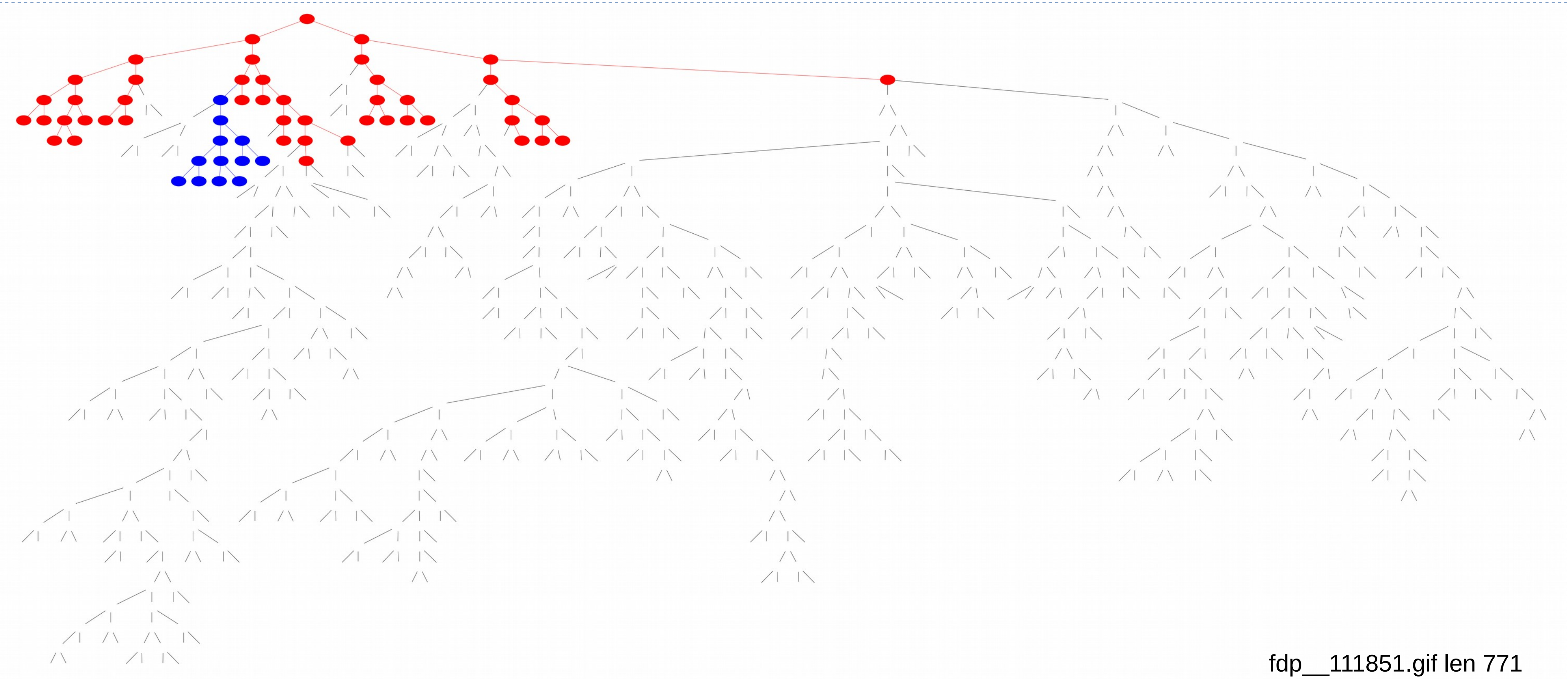
- Integer Koza's Fibonacci problem [GECCO 2022 Companion pp574-577].
  - Recursive program to generate Fibonacci sequence

    $X_J = X_{J-1} + X_{J-2}$ 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...
  - 0 1 2 3 J + - * SRF

    SRF(j,default) = j^{th} value. default applies if j is invalid
  - Twenty tests J=0 … 19
  - Population 50000, 1000 generations
  - Ten runs
- Change at run time each point in tree on each of the 20 tests
  - Two run time disruptions: +1 or replace with random int
  - +1 and RANDINT very similar
- Almost all run time disruptions make no difference

+1 Disruption, Fibonacci run 7, depth 33

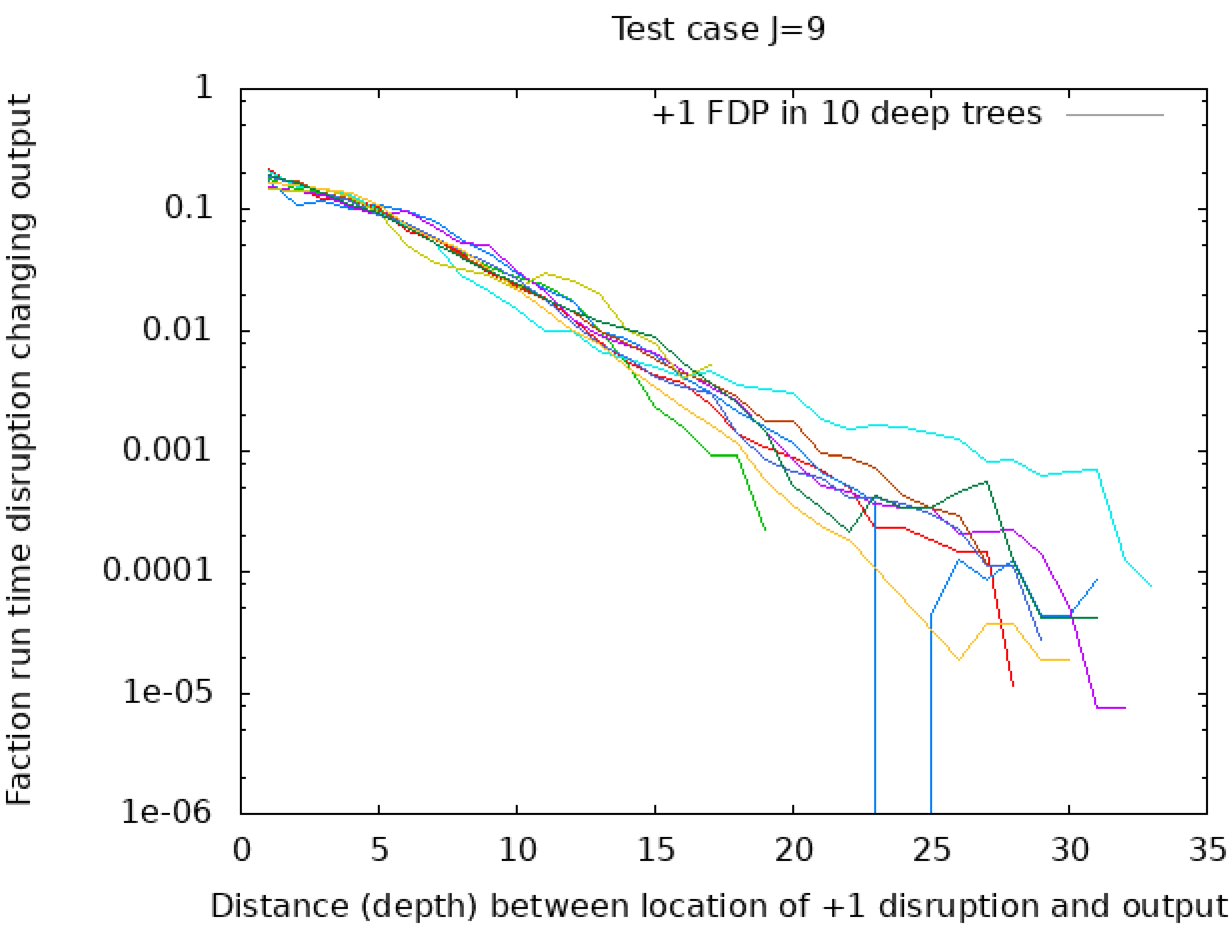
red 16-20 test cases, blue 1 test cases



fdp__111851.gif len 771

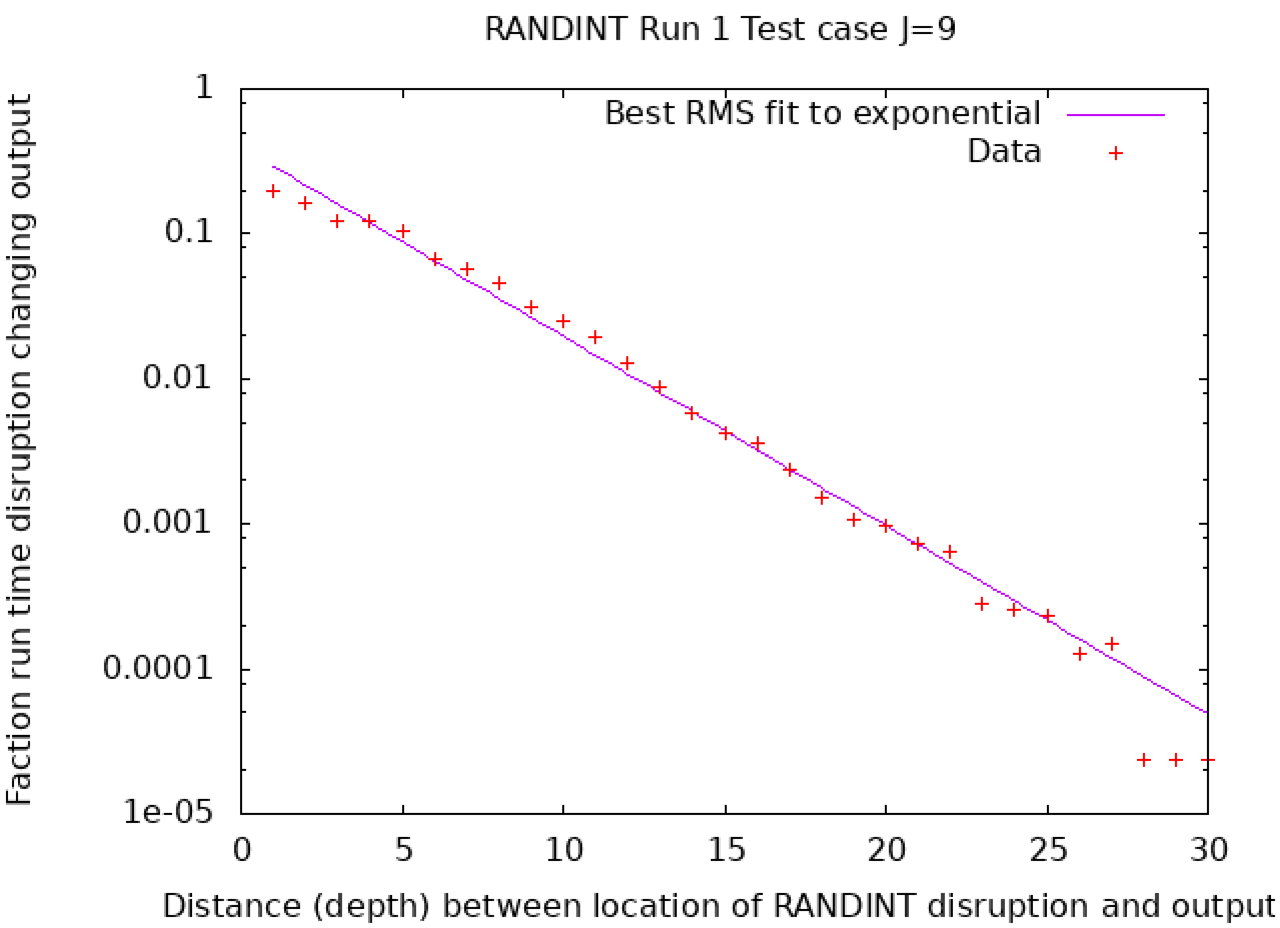
Only disruption near root node reaches output

Exponential fall in fraction of run time disruption changing program output with depth



fdp.gif

Exponential fall in fraction of run time disruption changing program output with depth



Deep floating point GP trees similar

params.gnu

Using Magpie to Sample C++ Mutations

- Genetic Improvement tool Magpie
- PARSEC suite of benchmarks to test parallel super computers for NASA. Mostly numeric but includes some image processing, including VIPS
- VIPS C++ image processing library 90,000 lines
- Chose vipsthumbnail (parallel multi-threaded take large image create small image 128 pixels wide
- Use linux perf to profile vipsthumbnail select all VIPS functions perf reports
- Use GDB to select all functions called enroute to top CPU using function
- Remove unused functions (a few unused lines, eg if/switch case included)
- 90,000 => 7328 lines, in 37 files. srcml => 37 XML files
- 1000 random Magpie mutations, measure their impact, measure depth

Magpie Mutating C++

- Magpie https://github.com/bloa/magpie
- VIPS image thumbnail benchmark (use 37 files 7328 LOC)

3264 x 2448

128 x 96
thumbnail

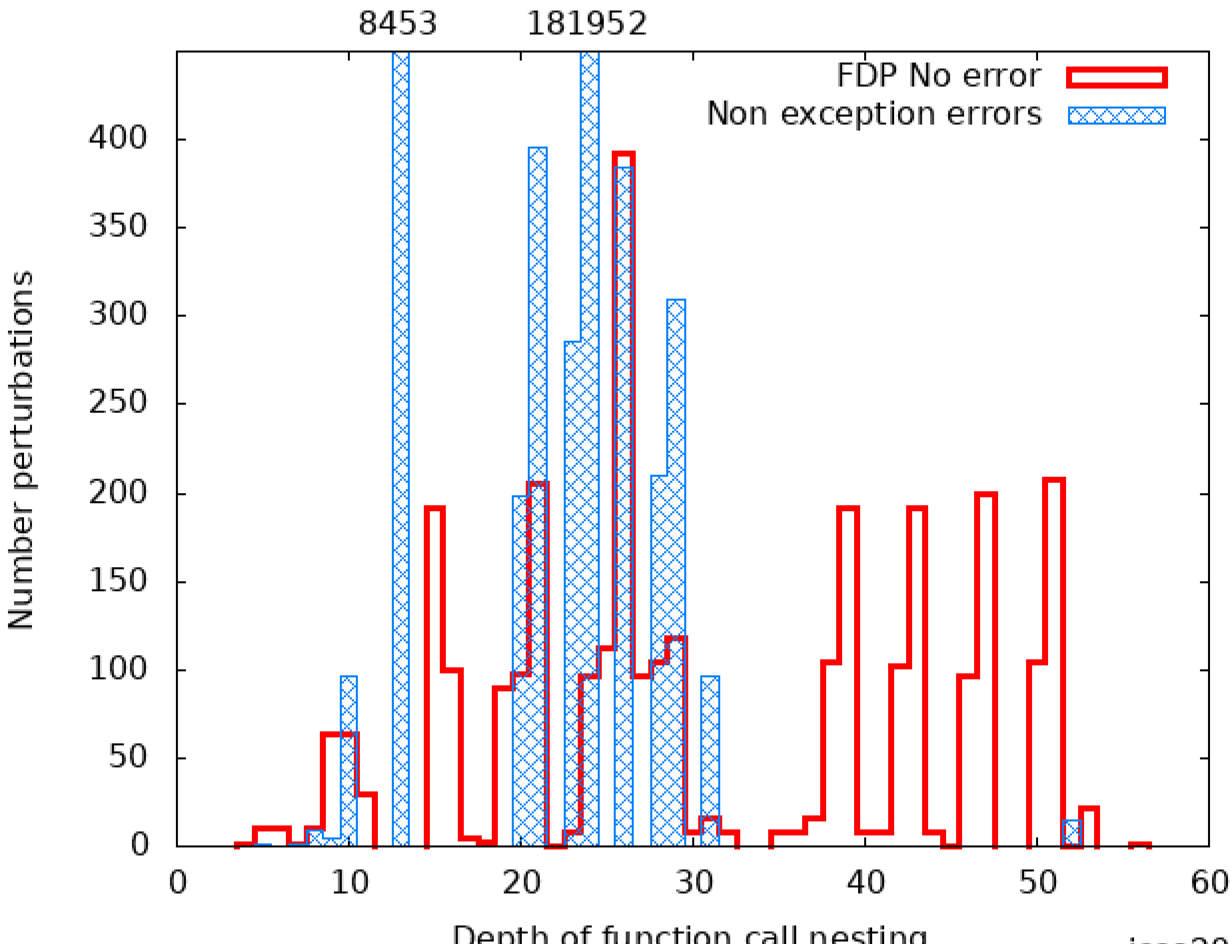
1000 random Magpie VIPS mutants

- VIPS image thumbnail benchmark (use 37 files 7328 LOC)
  - try to exclude unused code
- Magpie mutating source code as XML, (mostly) syntax preserving, mostly compiles, runs, gives right answer 526
- 37 cases output wrong but no exception.
  - Randomly choose 25 of 37, compare with 25 where mutant code is run, changes state but output is unchanged

Compiled, ran correct output	526	Correct output	438
		Mutation is identical to original code	88
Failed to compile	302		
Failed to run correctly or gave incorrect output	164	exception	127
		output error	37
Magpie TypeError	8		

25 v 25 Mutants. Deep less impact

- 25 mutants change execution but no change to output
- 25 mutants which change execution (without causing segfault) but change output

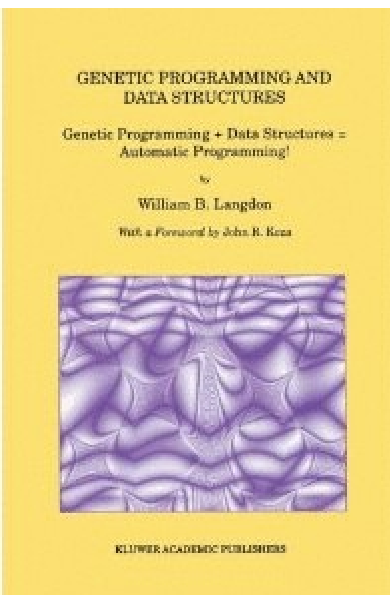


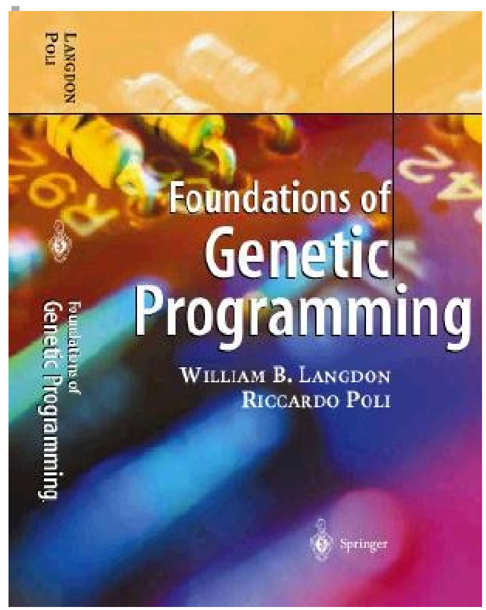
Conclusions

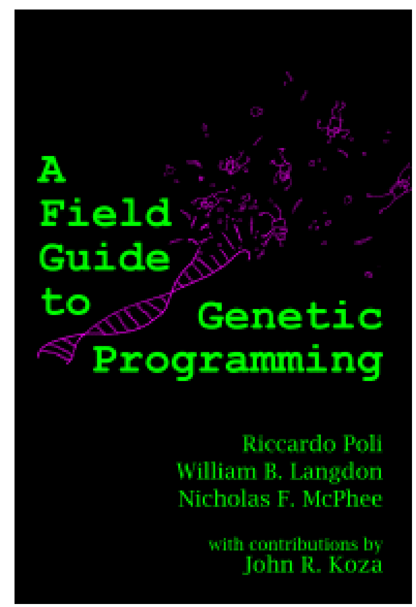
- Information theory predicts failed disruption propagation.
  - In evolved pure nested functions (Genetic Programming)
    - Impact of most mutations lost. Exponential decay with depth
  - In deeply nested C++ code
    - Excluding segfault etc., most mutations >30 nested function calls did not change output
- Test oracle need to be close to error for tests to find them
- Unit testing v. system testing, flaky tests
- Software code bloat, depth important
- Software robustness: deeper code harder to test, improve, optimise, repair?
- Automatic bug fixing (APR): avoid deep mutations, make shallow changes near output? Add multiple test probes (test oracles)?

Genetic Programming

W. B. Langdon







Human-Competitive results $10,000 prizes

Email your entry to goodman@msu.edu
by **Friday 31 May**

CREST

References

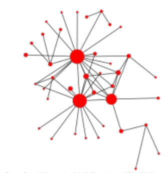
1) Evolving Open Complexity, W.B. Langdon, EI 2022
2) Long-Term Evolution Experiment with Genetic Programming, W.B. Langdon and W.Banzhaf, Artificial Life, 2022 28(2) pp173-204.
3) Dissipative Arithmetic, W.B. Langdon, Complex Systems. 2022, 31(3) 287-309
4) Deep Genetic Programming Trees are Robust, WB Langdon, ACM TELO, 2022 2(2) 6.
5) Genetic Programming Convergence, WB Langdon, GP+EM, 23(1) 71-104
6) Measuring Failed Disruption Propagation in Genetic Programming, GECCO 2022, 964-972, WB Langdon, *et al.*
7) Failed Disruption Propagation in Integer Genetic Programming, GECCO comp 2022, 574-577, WB Langdon, *et al.*
8) Information Loss Leads to Robustness, W.B. Langdon and J.Petke and D. Clark, IEEE Software Blog, 12 Sept. 2021.
9) Dissipative Polynomials, W.B. Langdon and J. Petke and D.Clark,in GECCO 2021 comp., pp1683-1691. DOI
10) Software Robustness: A Survey, a Theory, and Some Prospects, J.Petke, D.Clark and W.B. Langdon, in ESEC/FSE 2021 (IVR), pp 1475-1478, DOI
11) Long-Term Evolution of Genetic Programming Populations, W.B. Langdon. In GECCO 2017 Comp., 235-236. DOI

The Genetic Programming Bibliography

16873 references, 16000 authors

Make sure it has all of your papers!

E.g. email W.Langdon@cs.ucl.ac.uk or use | Add to It | web link



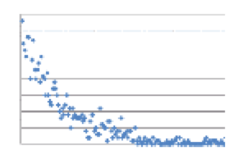
Co-authorship community.

Downloads

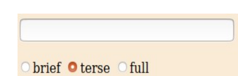
Downloads by day

A personalised list of every author's GP publications.

blog



Your papers



Text search

Googling GP bibliography, eg:

Development and learning site:gpbib.cs.ucl.ac.uk