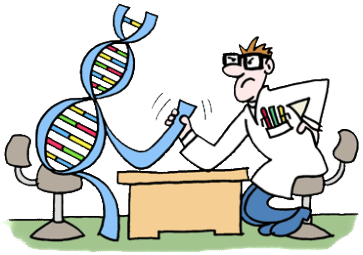
CREST

Deep Mutations have Little Impact

4th International Conference on Embodied Intelligence
Thursday 21 March 2024

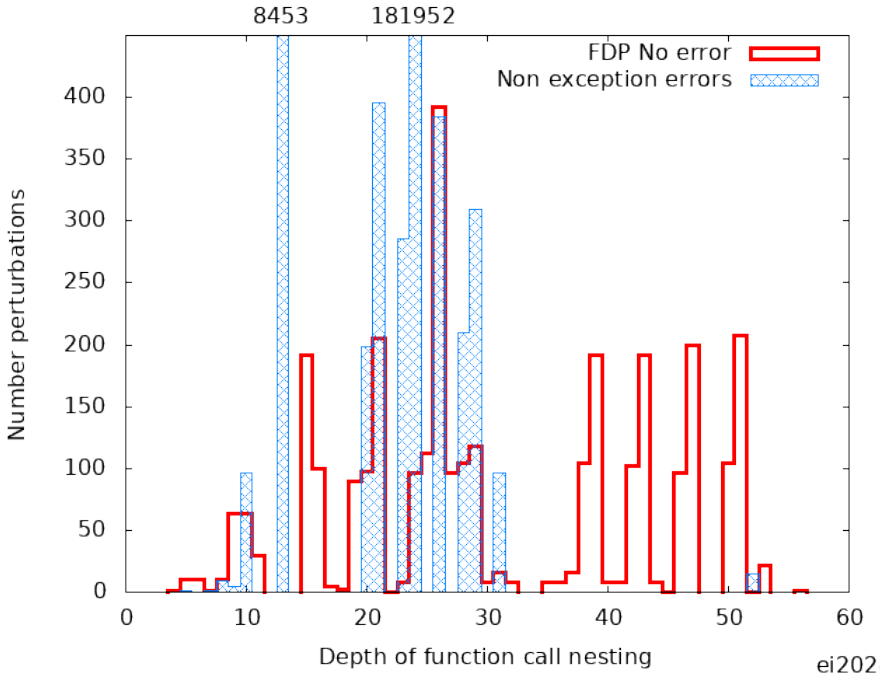


Humies $10000 prizes
Submit by **Friday 31 May**

W. B. Langdon, UCL

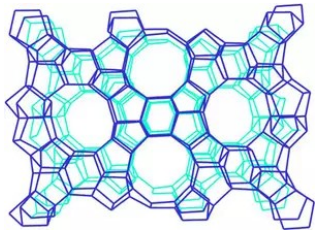
Information theory suggests
for most deeply nested mutations
disruption fails to propagate to the output.

W.B. Langdon and D. Clark. In GI@ICSE
16 April 2024, Lisbon



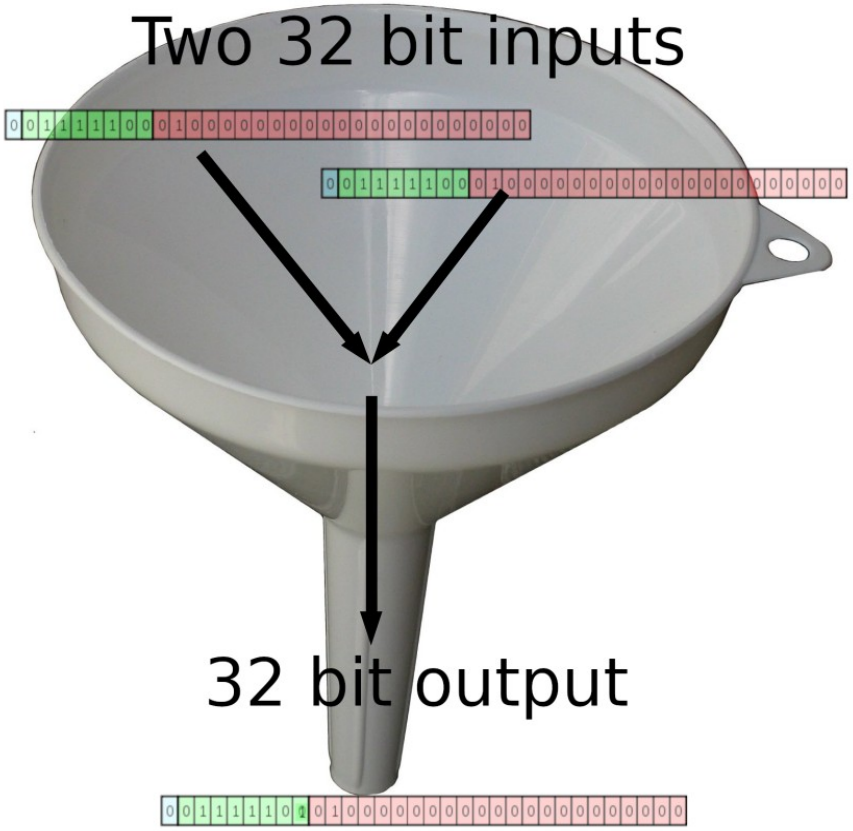
Deep Mutations have Little Impact

- Information theory says impact of disruptions lost with distance when nested
- True in pure functions
- Evidence true in real C++ software with side effects, globals
- Implications:
  - White box: Put test probes near changes
  - Black box: limit depth of nesting
  - Evolved embodied intelligent code must have high surface area
    - lungs, sponge, coral, pumice, zeolites. porose, low density.



Information Funnel

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

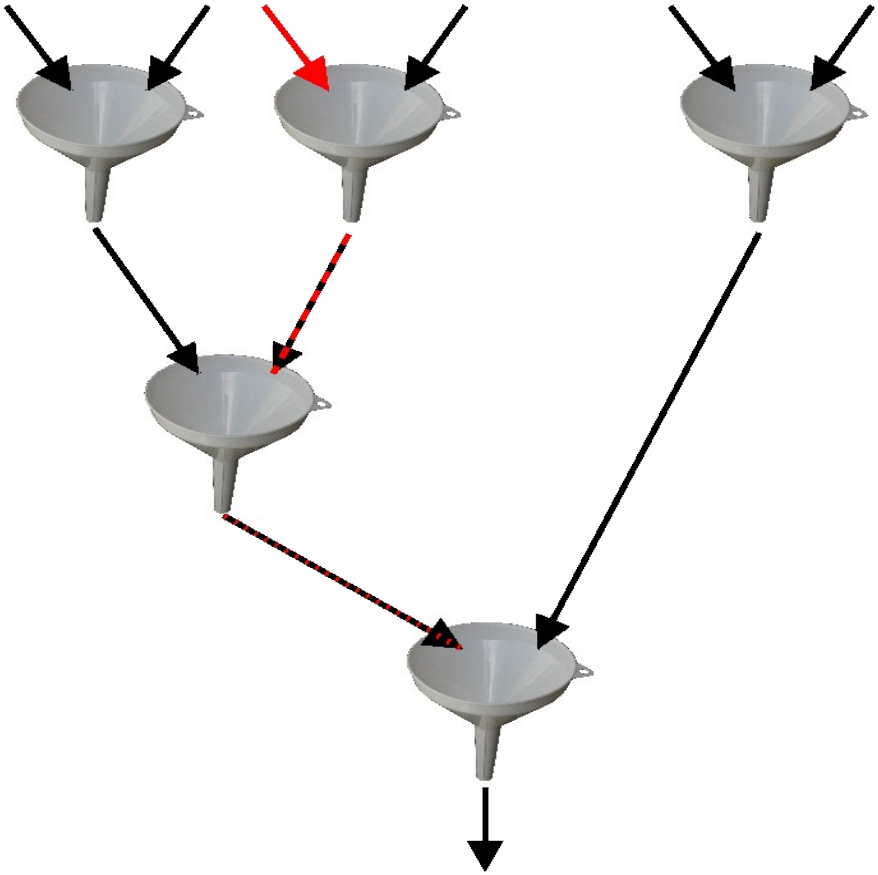


Information funnel

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 σ = √8.25 entropy=$\log_2 10$
- n digit mean 4.5 n, σ = $(n\ 8.25)^{½}$,
  - 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	√(8.25n)		$2+\log_2\sqrt{(8.25n)^{½}}$	$< 100\% = 1 - 2/(3.3n) - (1/3.3n)\log_2\sqrt{(8.25n)^{½}}$

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

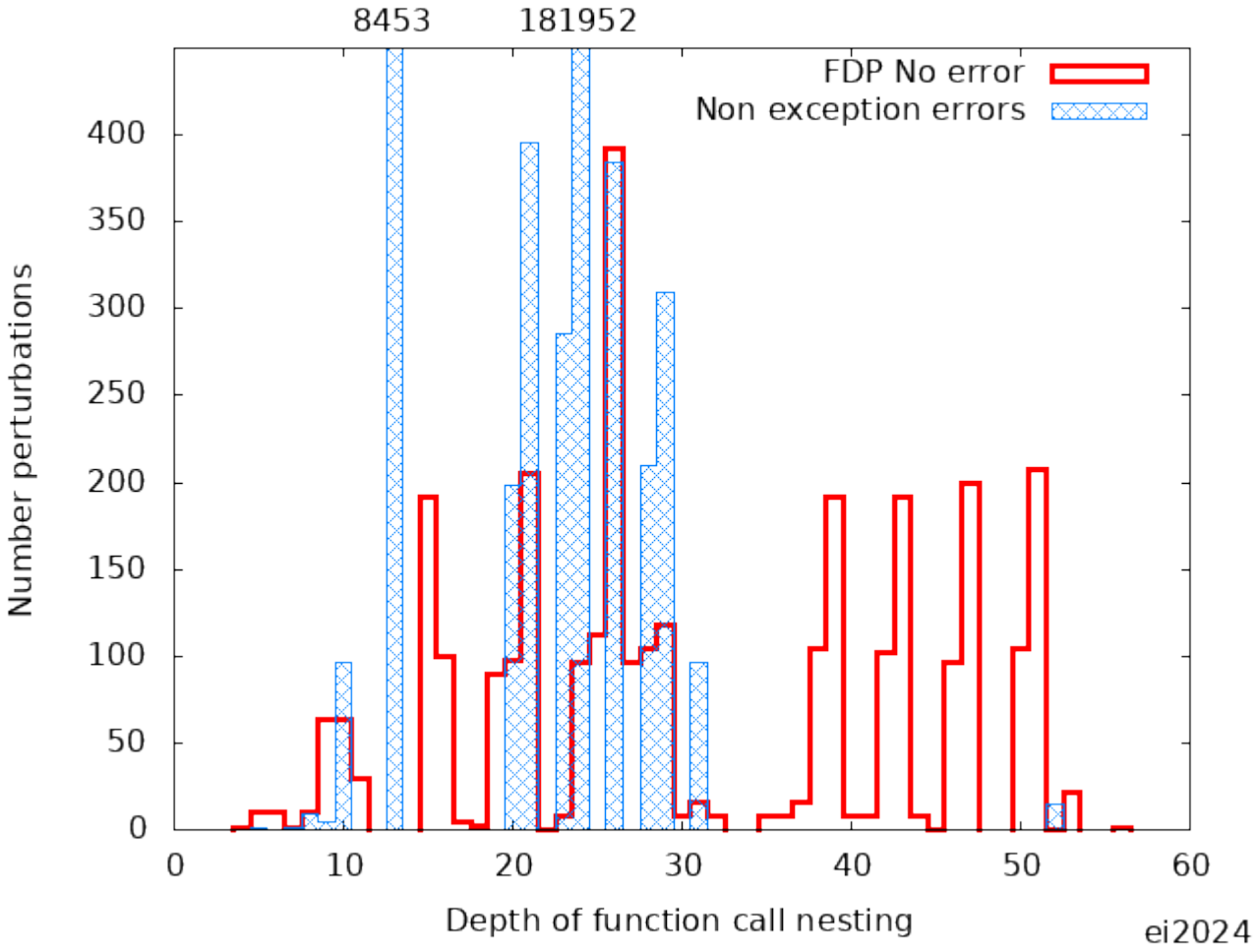
1000 random Magpie VIPS mutants

- 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		

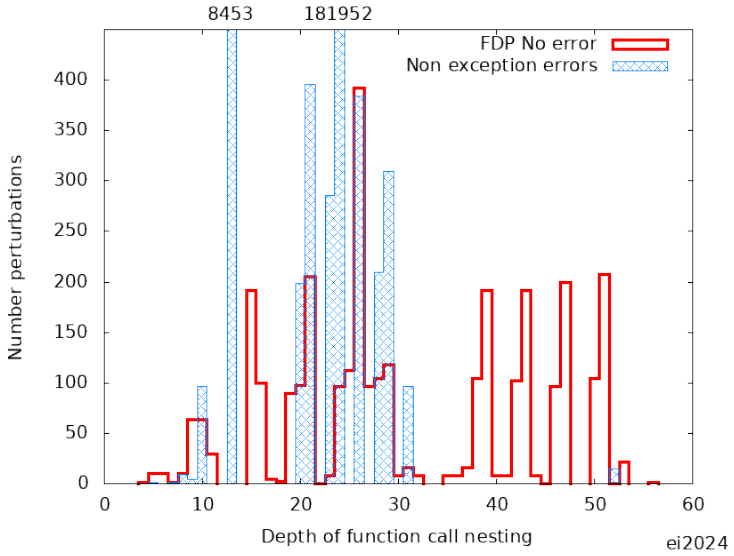
25 v 25 Mutants. Deep less impact

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



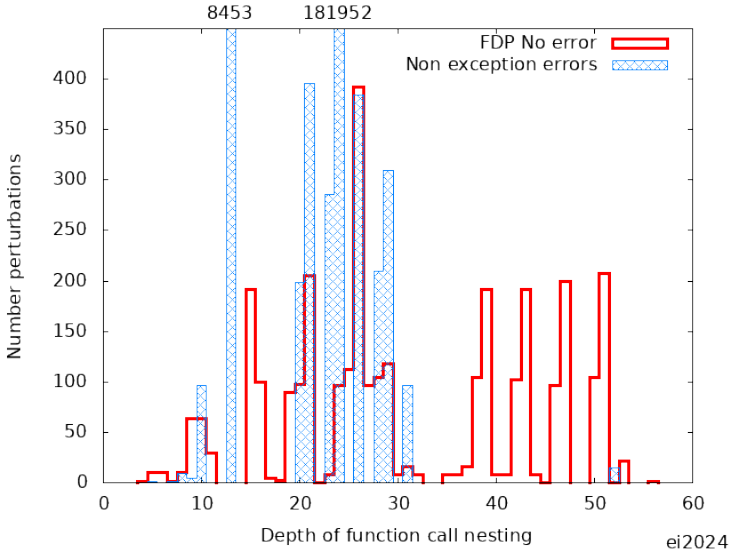
Avoid deep mutations: Make shallow changes

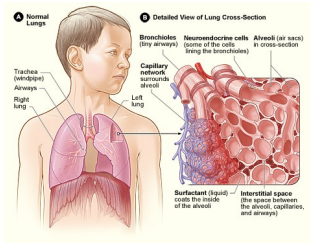
- Information theory predicts failed disruption propagation.
  - Deeply nested C++ code
    - Excluding segfault etc., most mutations >30 nested function calls did not change output
  - In evolved pure nested functions (Genetic Programming)
    - Impact of most mutations lost. Exponential decay with depth
  - Need to be close to error for tests to find them
- Need shallow open embodied architecture to evolve complexity



Deep structures are robust, hard to adapt: Shallow code

Many, small, interlinked, low density, high surface area, codes close to fitness ecosystem



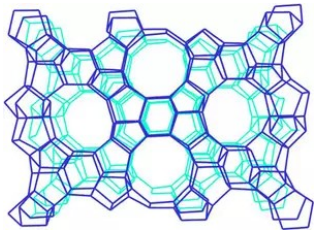


Lungs

Sponge

Coral

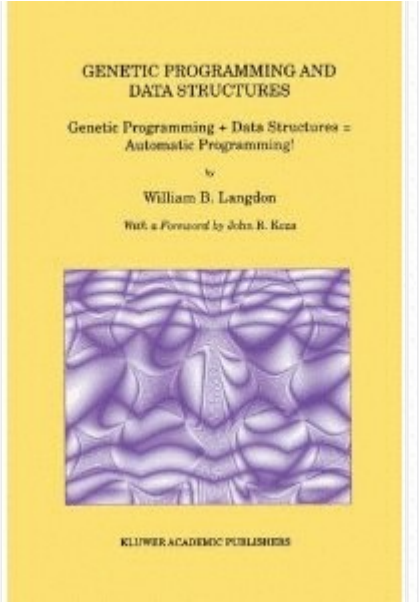
Pumice

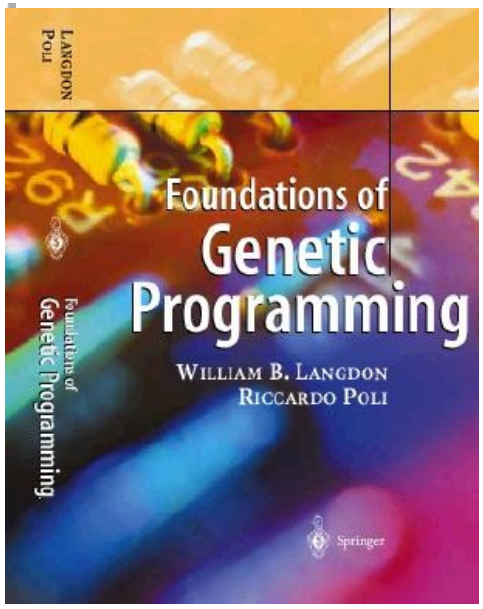


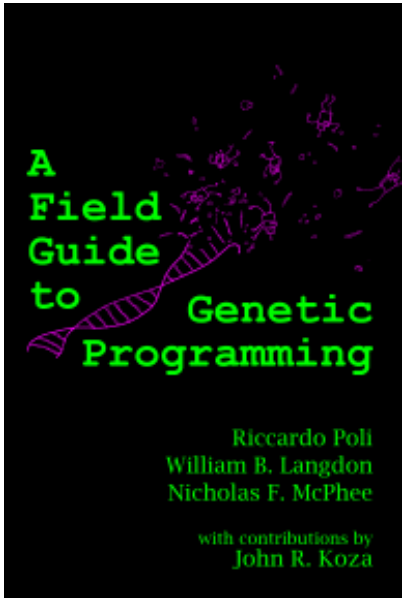
Zeolite

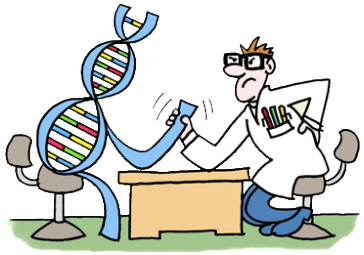
Genetic Programming

W. B. Langdon









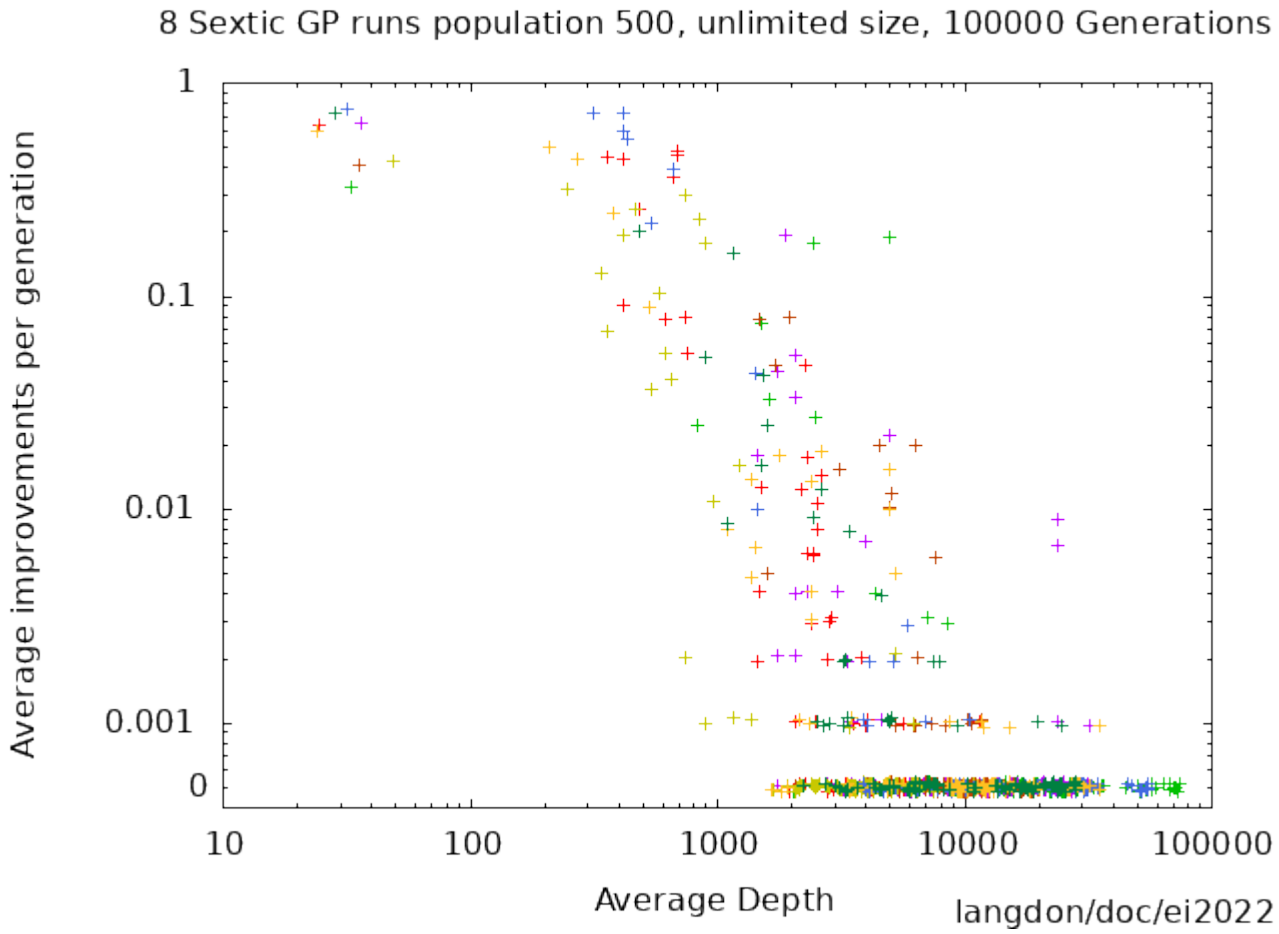
Human-Competitive results $10,000 prizes

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

Long Term Evolution Experiments

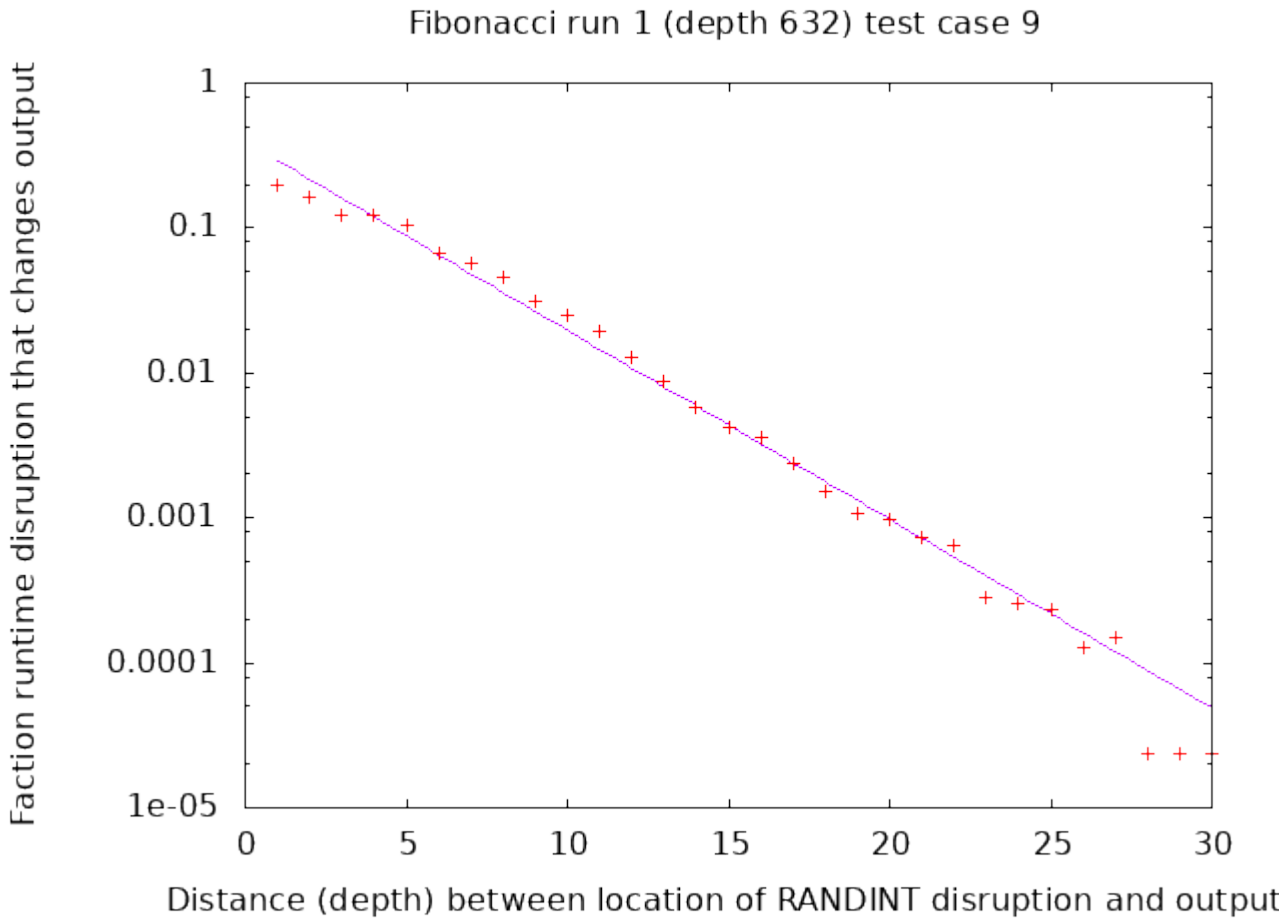
- **LTEE** shows E.Coli continued innovation 75000 generations
- Genetic Programming continued fitness improvement a million generations BUT GP slows as expressions get deeper
  - Impact of mutations lost, mostly due to rounding error
  - In deep integer trees 92% to 99.97% of evaluation changes have no effect
- Exponential decay with depth
  - Need to be close to error for tests to find them
  - On average <7 more than 50% errors detected

Deeper programs harder to evolve



As the GP populations evolve they find thousands of improvements but at a slower rate as the trees get deeper. Note log scales.

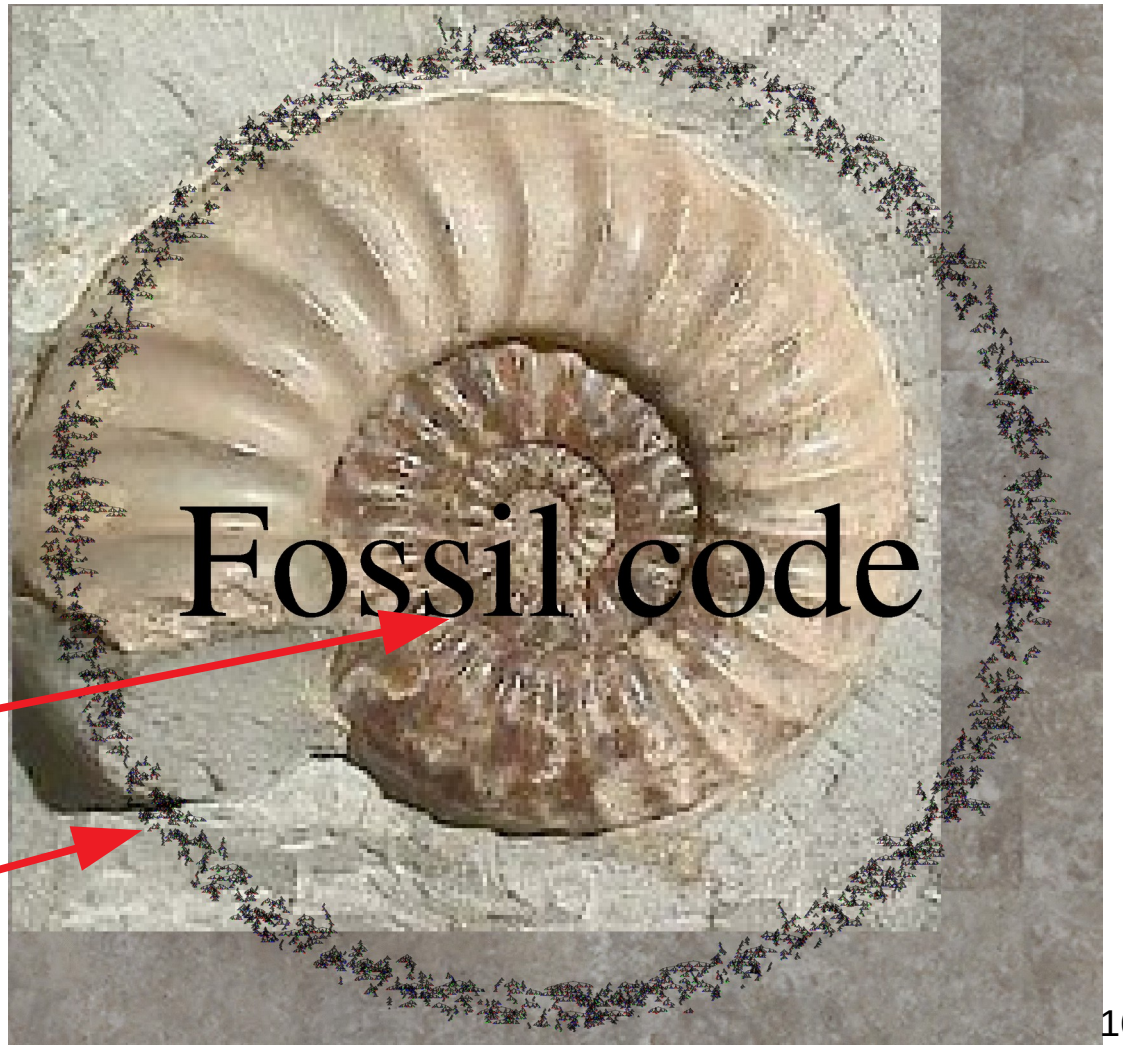
Exponential fall in fraction of run time disruption changing program output with depth



Perturb evaluation of deep evolved Fibonacci program. Replacement with random value seldom has externally visible impact. Note log vertical scale.

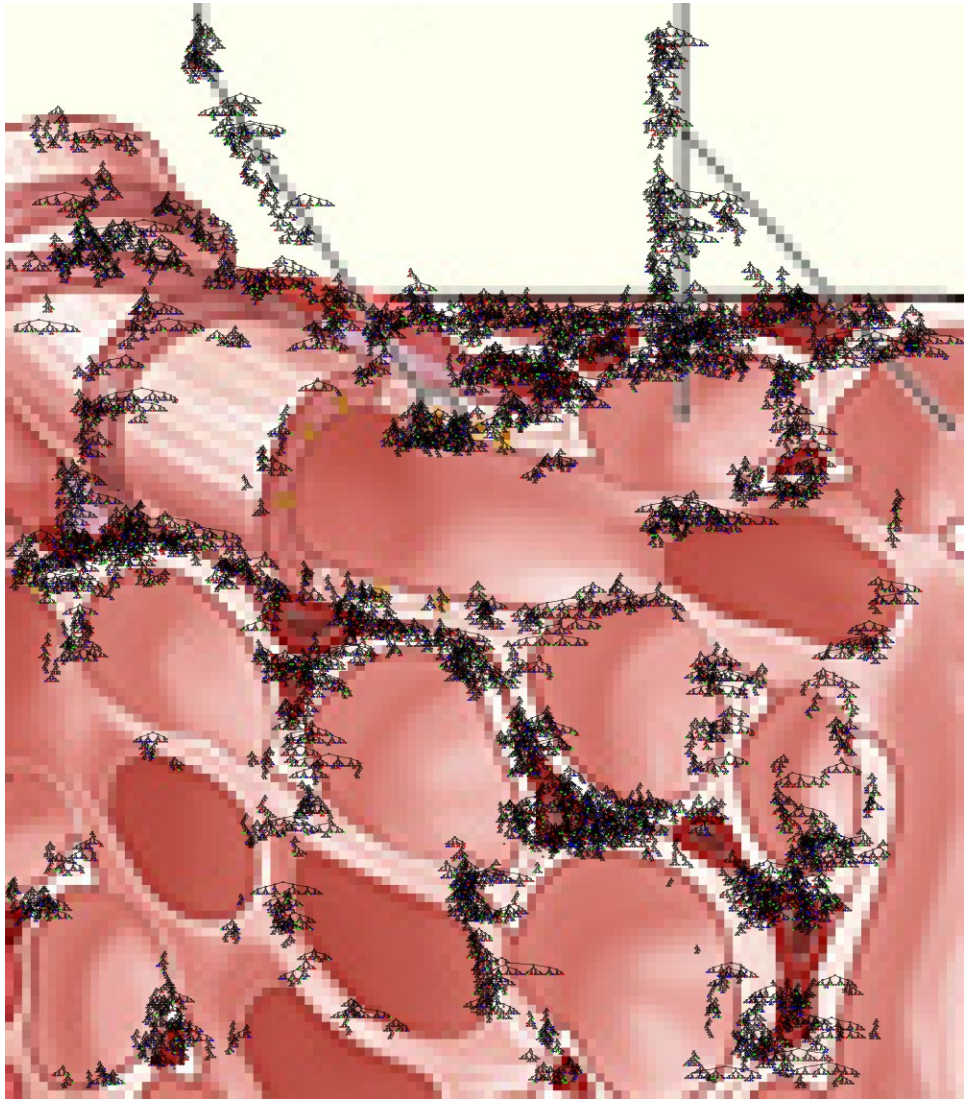
To evolve large complex code, Must **AVOID** large fossil of dead code

- With **deep code** most crossovers and mutations make **no difference**.
- Leading to random drift
- Not directed evolution
- To avoid dead center evolving code must be near environment.



Evolve Large Open, Lung Like, Open Architecture

- Make code is shallow.
- Shallow code does not suffer failed disruption propagation.
- Instead fitness disruption caused by mutations and crossover do have impact.
- Fitness can direct evolution.
- Suggest large porous code
- All code near organism's environment.
- Communication between code internally & externally eased by globals, side effects, pipes, TCP/IP etc.



W.B. Langdon, EI 2022

Evolve **Open** Complexity

1) Information theory predicts, without side effects, nested irreversible computation will loose information and so
2) nested expressions suffer failed disruption propagation.
3) Meaning impact of deep code changes does not reach output
4) **Deep mutations do not change fitness**
5) Without fitness changes there is no evolution
6) To avoid code fossilising, changes must impact performance
7) To **evolve** code it must be **shallow**, close to environment
8) Open porous lung like code, possibly in many dimensions, with open channels between shallow <7 code modules

References

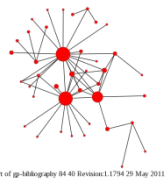
1) Deep Mutations have Little Impact, W.B. Langdon and David Clark, GI 2024
2) Evolving Open Complexity, W.B. Langdon, EI 2022
3) Long-Term Evolution Experiment with Genetic Programming, W.B. Langdon and W.Banzhaf, Artificial Life, 2022 28(2) pp173-204.
4) Dissipative Arithmetic, W.B. Langdon, Complex Systems. 2022, 31(3) 287-309
5) Deep Genetic Programming Trees are Robust, WB Langdon, ACM TELO, 2022 2(2) 6.
6) Genetic Programming Convergence, WB Langdon, GP+EM, 23(1) 71-104
7) Measuring Failed Disruption Propagation in Genetic Programming, GECCO 2022, 964-972, WB Langdon, *et al.*
8) Failed Disruption Propagation in Integer Genetic Programming, GECCO comp 2022, 574-577, WB Langdon, *et al.*
9) Information Loss Leads to Robustness, W.B. Langdon and J.Petke and D. Clark, IEEE Software Blog, 12 Sept. 2021.
10) Dissipative Polynomials, W.B. Langdon and J. Petke and D.Clark,in GECCO 2021 comp., pp1683-1691. DOI
11) 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
12) Long-Term Evolution of Genetic Programming Populations, W.B. Langdon. In GECCO 2017 Comp., 235-236. DOI

The Genetic Programming Bibliography

16702 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

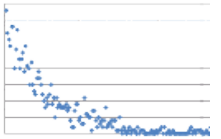


Co-authorship community.
Downloads

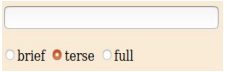
Downloads by day

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