CREST

# Evolving Open Complexity

2nd International Conference on Embodied Intelligence
Thursday 24 March 2022

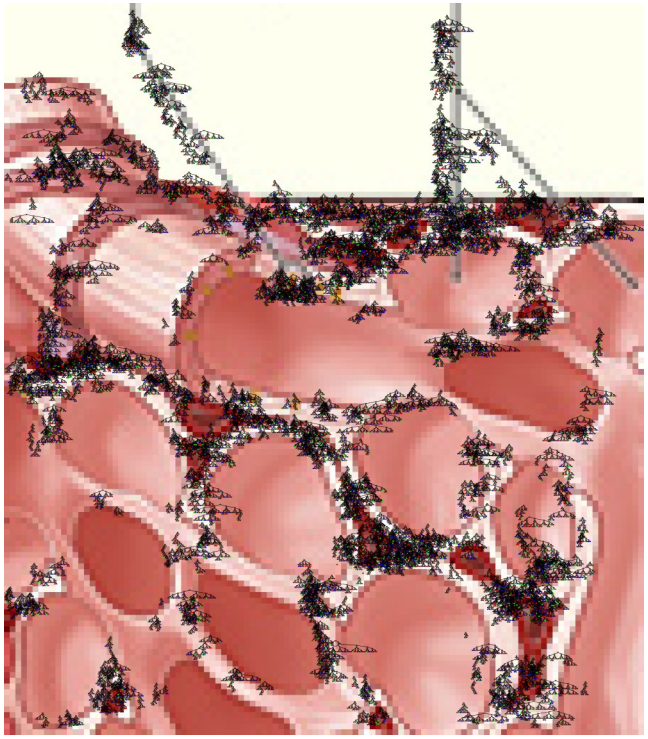
W. B. Langdon, UCL

SIGEVOlution, Spring, 2022

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

Instead suggest lung like open architecture where most code is less than seven levels from the environment.

arXiv:2112.00812

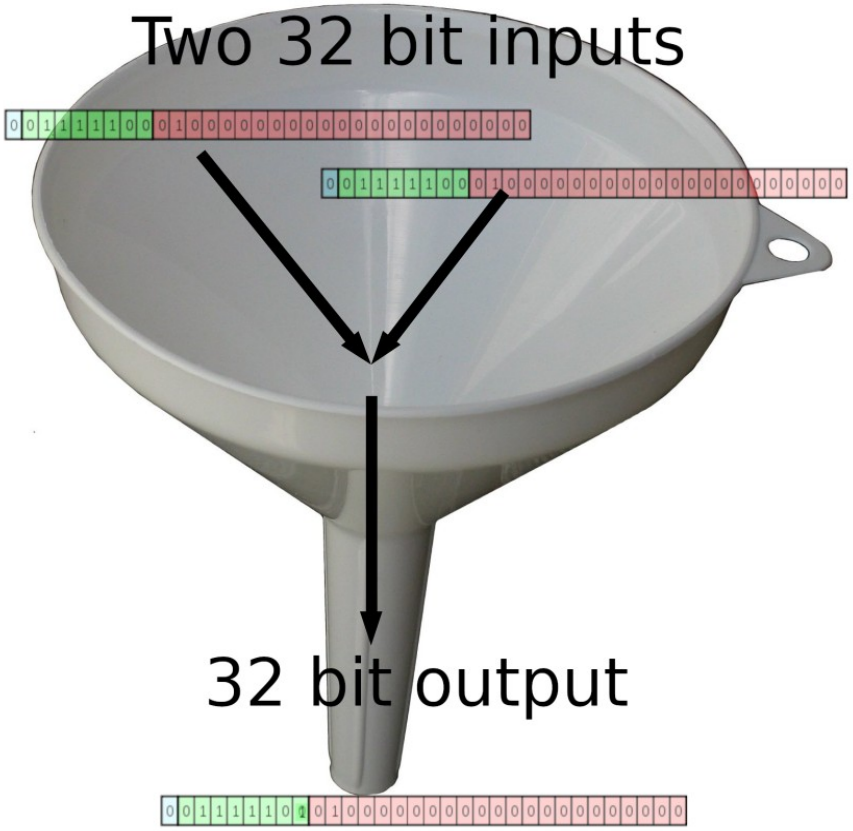


# Long Term Evolution Experiments v. Artificial Evolution

- Information theory predicts nested functions (eg + * - /) will fail to propagate disruption.
- **LTEE** shows E.Coli continued innovation 75000 generations
- Genetic Programming continued fitness improvement a million generations BUT GP slows
  - 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
- Conclude need shallow open architecture to evolve complexity

# Information Funnel

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

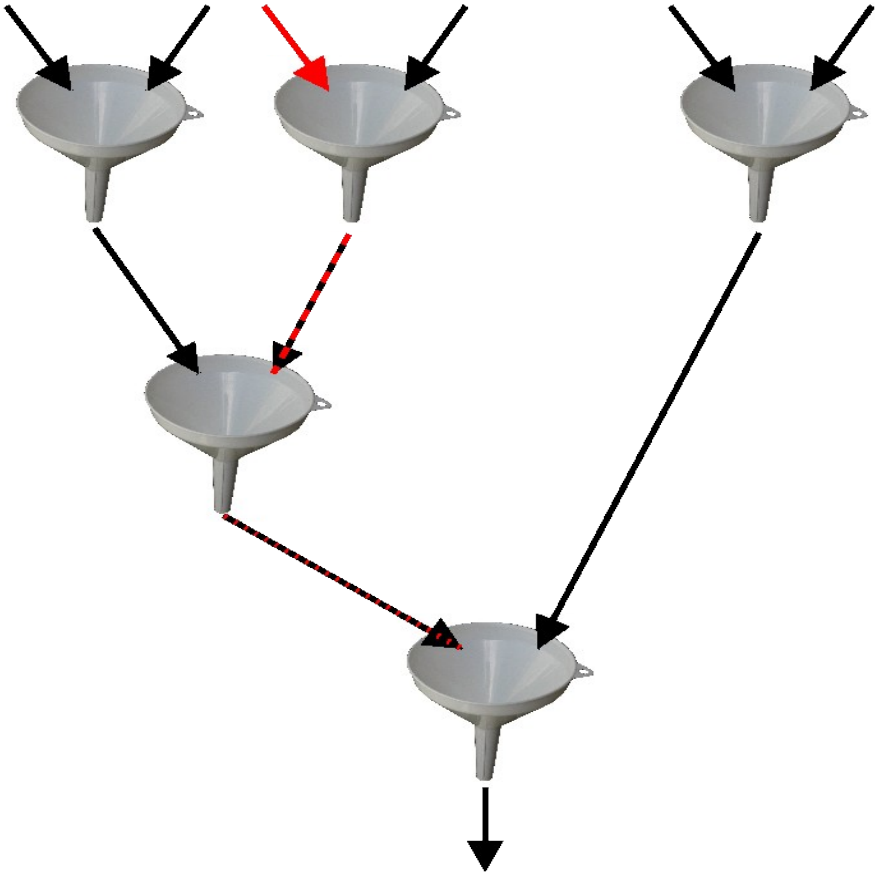



**Information funnel**

More information enters than leaves

# Information flow in five nested functions

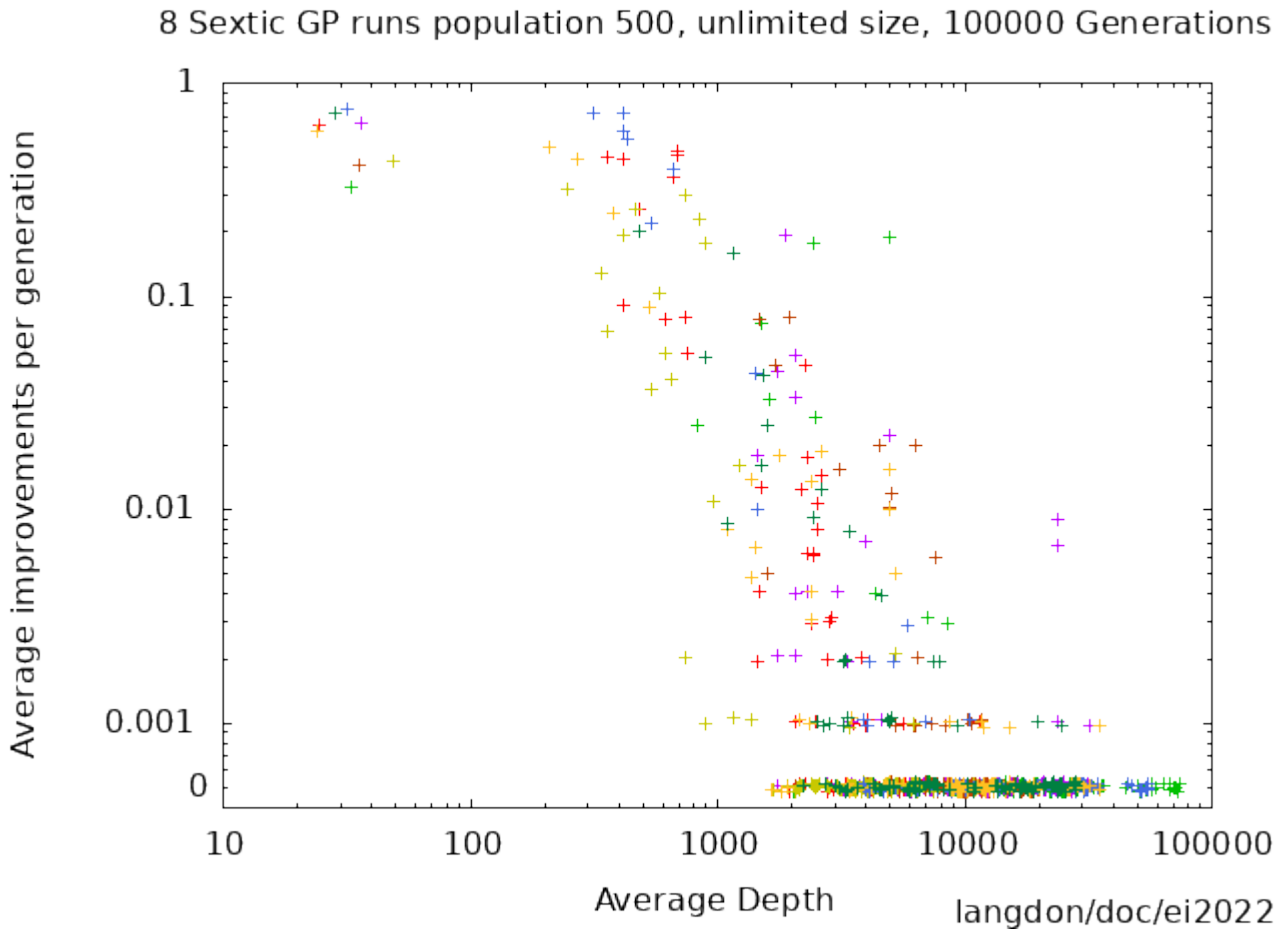
Potential information loss at each (irreversible) function




Disruption may fail to reach reach output.

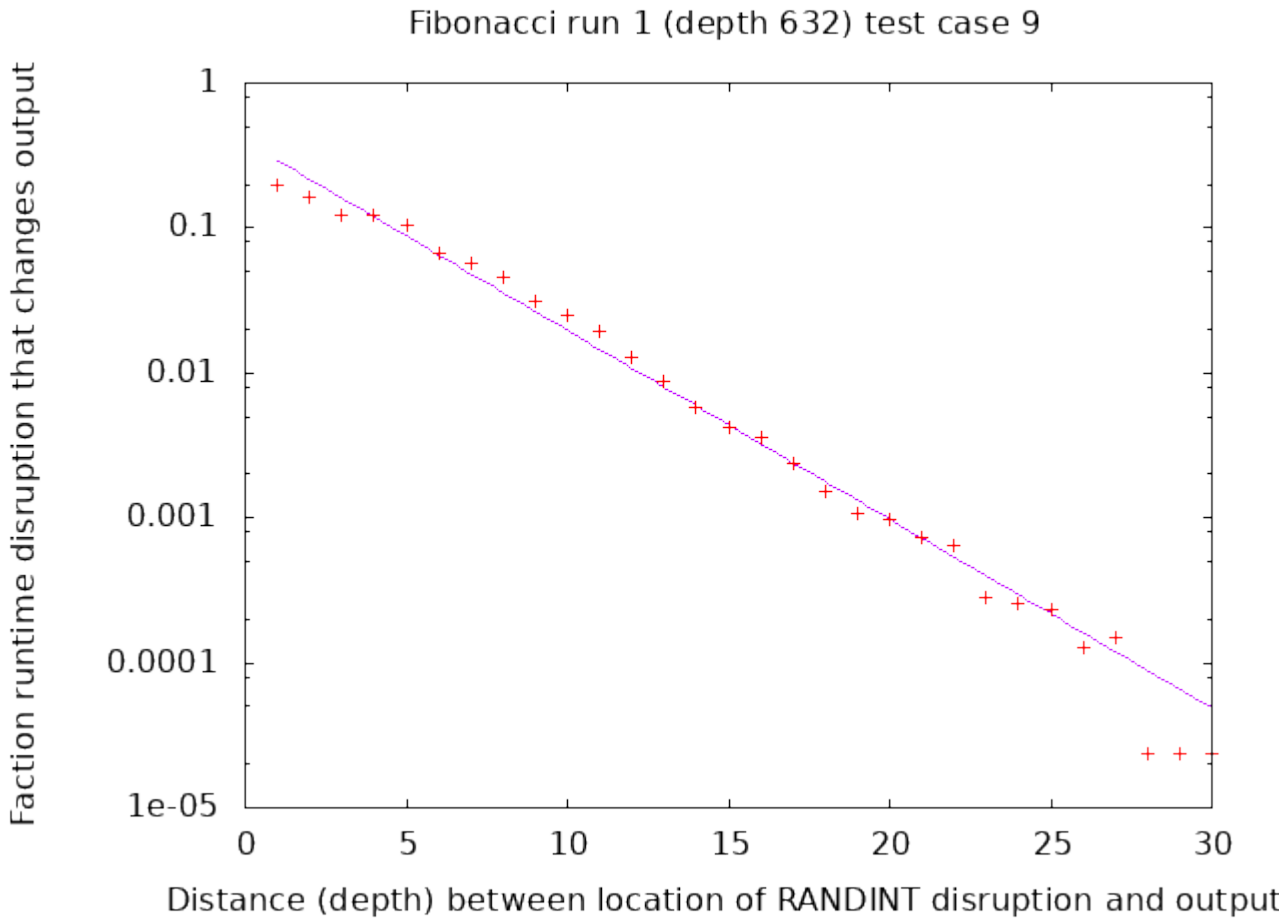
(No side effects.)

# 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.

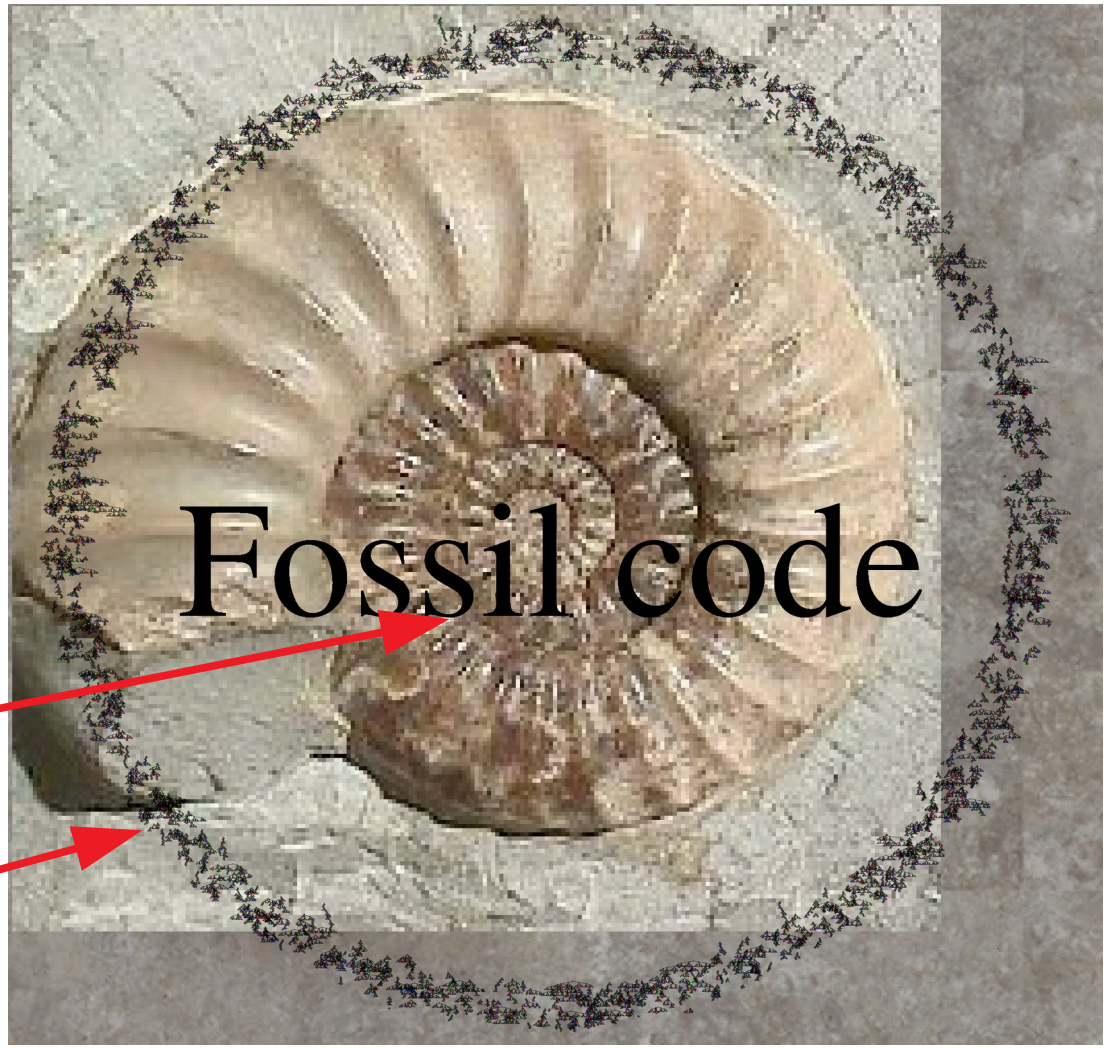
# Exponential fall in fraction of run time disruption changing program output with depth




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

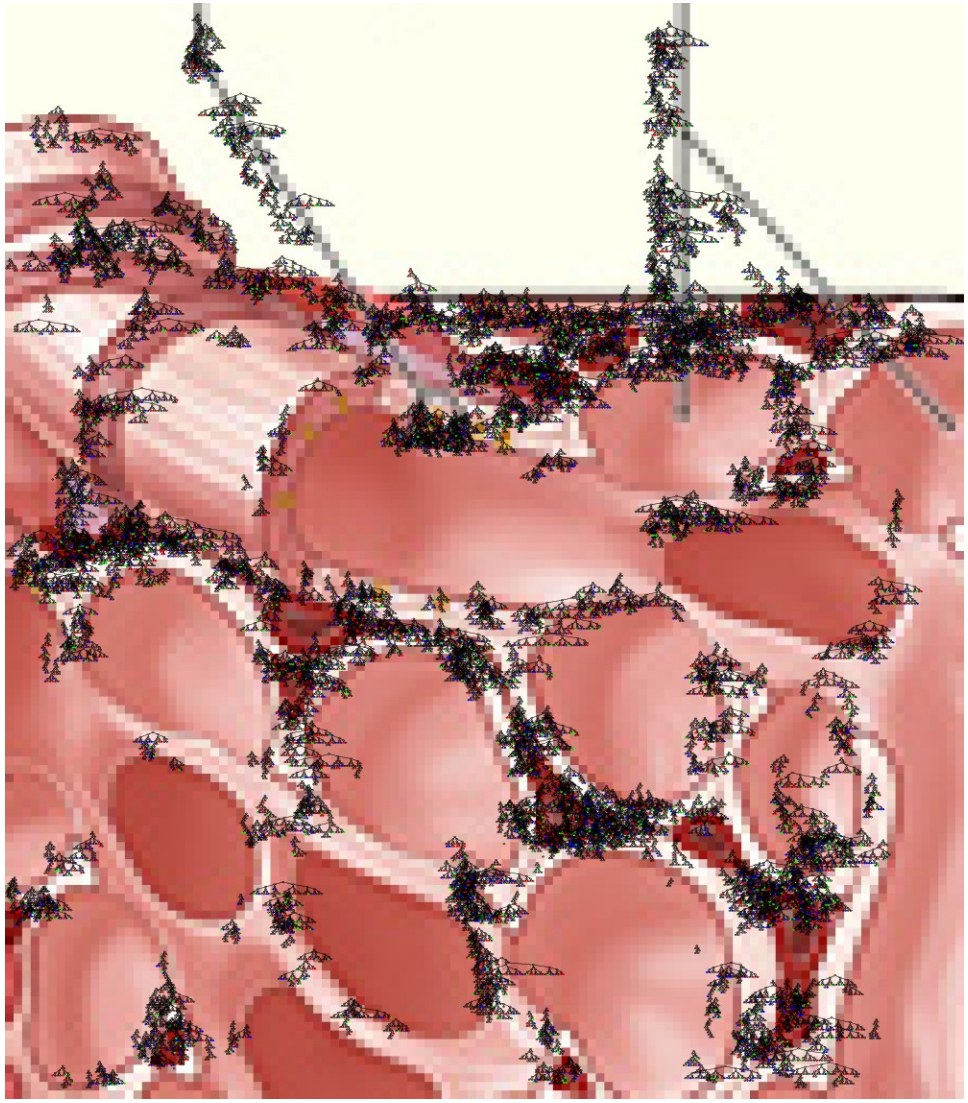
# To evolve large complex code, Must **AVOID** large fossil of dead code

- 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.

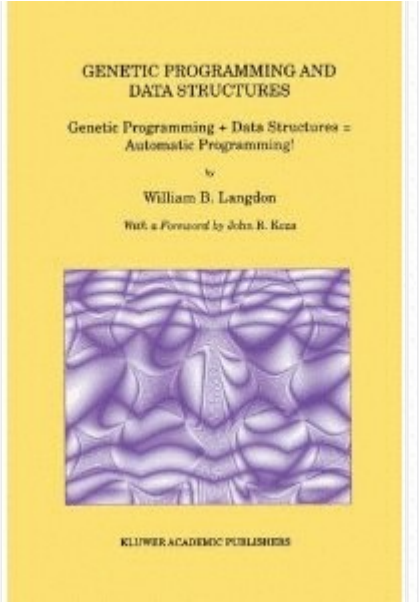


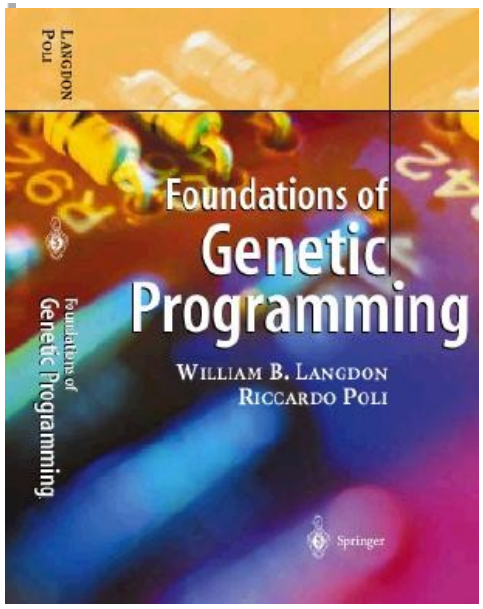
# Evolve **Open** Complexity

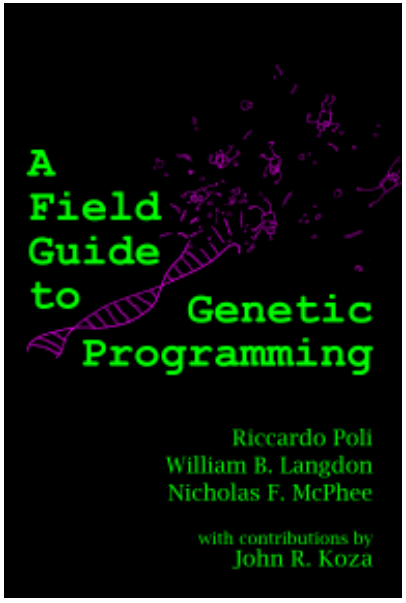
1) Information theory predicts, without side effects, nested irreversible computation will loose information and so
2) nested expressions suffer failed disruption propagation.
3) Meaning impact of deep code changes does not reach output
4) **Deep mutations do not change fitness**
5) Without fitness changes 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

# Genetic Programming

## W. B. Langdon



CREST

# References


1) R.E.Lenski et al. Sustained fitness gains and variability in fitness trajectories in the long-term evolution experiment with *Escherichia coli*. Proc. Royal Soc. B, 282(1821), 2015. DOI

2) Evolving Open Complexity, W.B. Langdon, SIGEVOlution, 2022(1), arXiv:2112.00812

3) Long-Term Evolution Experiment with Genetic Programming, W.B. Langdon and W.Banzhaf, Artificial Life, *forthcoming.*

4) Information Loss Leads to Robustness, W.B. Langdon and J.Petke and D. Clark, IEEE Software Blog, 12 Sept. 2021.

5) Dissipative Polynomials, W.B. Langdon and J. Petke and D.Clark,in GECCO 2021 comp., pp1683-1691. DOI

6) 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

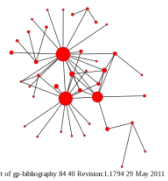
7) Long-Term Evolution of Genetic Programming Populations, W.B. Langdon. In GECCO 2017 Comp., 235-236. DOI

# The Genetic Programming Bibliography

**15405** references, 15000 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

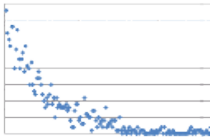


Co-authorship community.
Downloads

Downloads by day

A personalised list of every author's GP publications.

blog



Your papers

Googling GP bibliography, eg:
Development and learning site:gpbib.cs.ucl.ac.uk