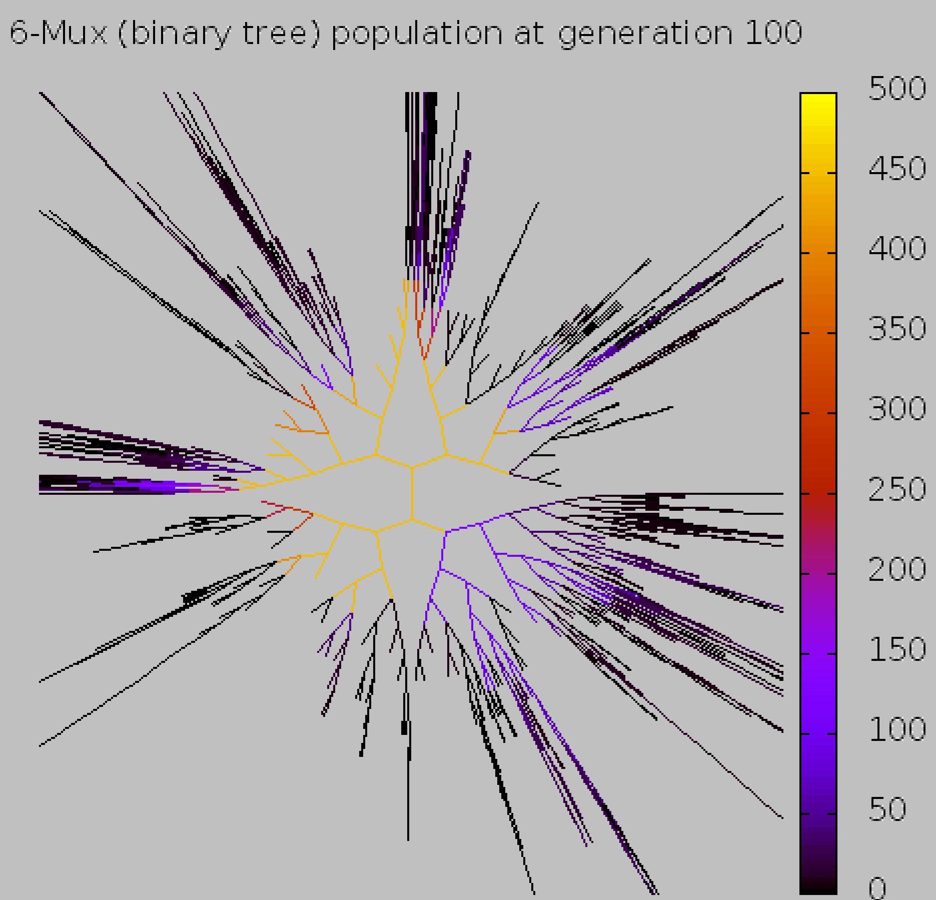
Convergence in Genetic Programming

W. B. Langdon

Computer Science. University College, London

500 trees



Humies

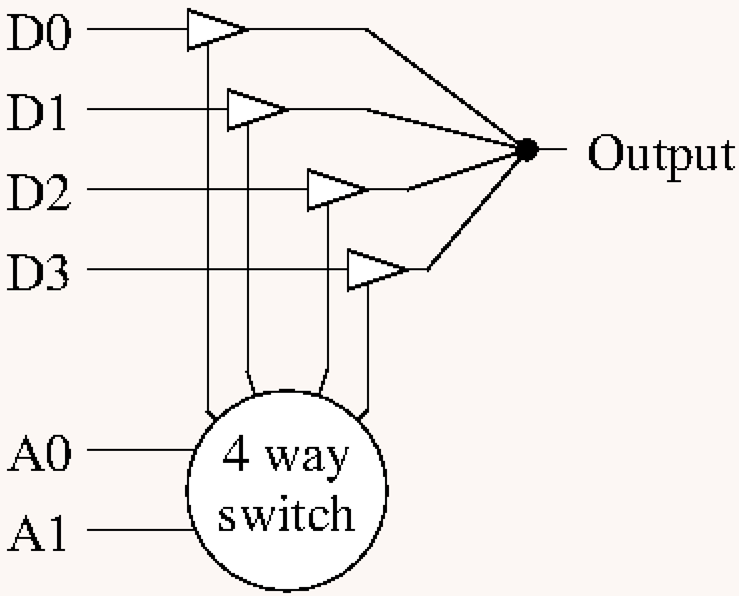
$10000 Human-Competitive Results

CREST

Convergence in Genetic Programming ~~and Long-Term Evolution Experiments~~

- More challenging problems may require running evolution for longer. Hence the need to study what happens in long runs. Perhaps we can anticipate and solve problems that may occur. [Added 13 May 2017]
- Results
  - quadratic tree growth
  - ~~differences from crossover only theory~~
  - converge of binary trees
  - ~~random drift stops bloats~~

6 Multiplexor



- GP bench mark.
- Six inputs:
  - Use two (D4 D5) as binary number to connect corresponding data lines (D0-D3) to the output
- Test on all 2^6=64 possible combinations
- Fitness score (0-64) is number correct

Impact of Subtrees

- Subtree like whole tree.
- Output of subtree is via its root node
- **Intron**: subtree which has no effect on overall fitness. I.e. its output does not impact on root node of whole tree.
- **Constant** subtree always has same output, i.e. same output on all 64 test cases.
- Remaining **effective code** has an impact on root node. Typically it is next root node

Example Intron: AND Function

A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

A	B	Out
0	0	0
~~0~~	~~1~~	~~0~~
1	0	0
~~1~~	~~1~~	~~1~~

Left: two input AND node.

Right: same but input B is always 0.

So output always 0. Input A has no effect.

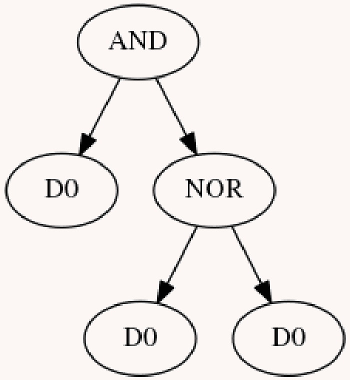
Subtree A is always ignored, even in child.

(NB no side effects)

Constants

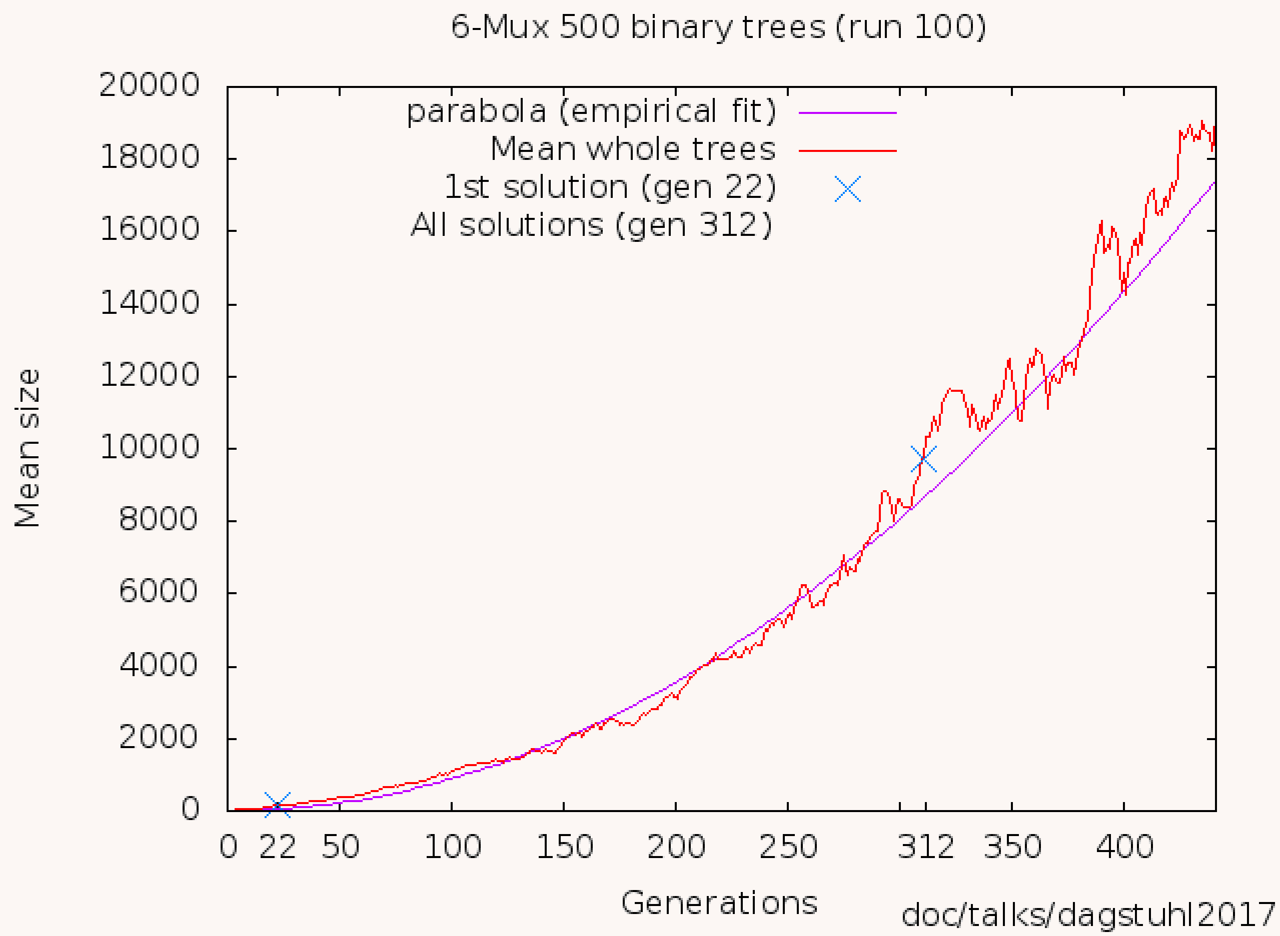
- Two constants: always 0 and always 1 (FFFFFFFFFFFFFFFF).
- E.g. evolve by negating input and ANDing with same input

  (AND D0 (NOR D0 D0)) = 0



- Constants help form introns but may be disrupted by crossover.
- However large subtrees which always output either 0 or 1 tend to be resilient to crossover

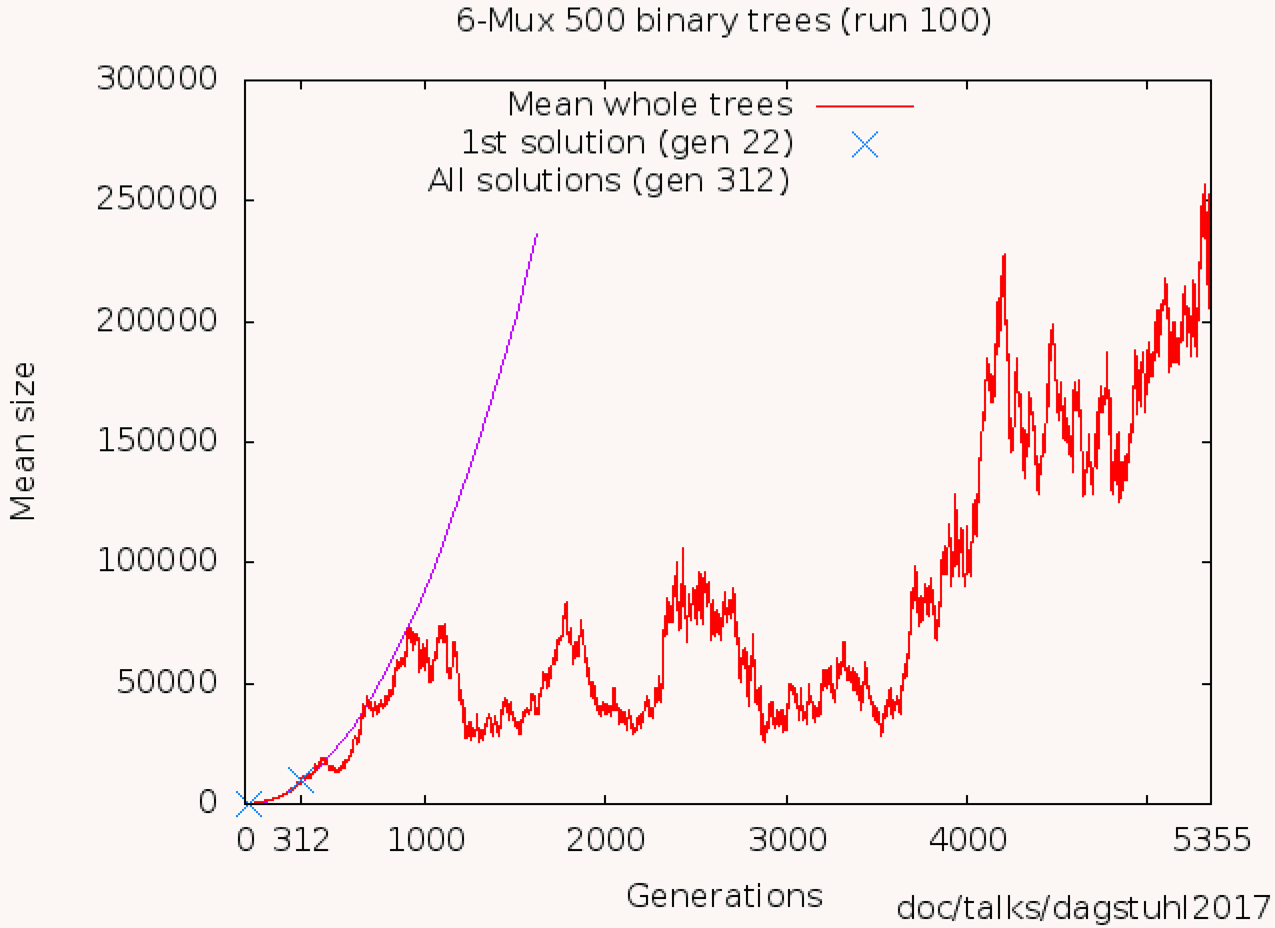
Evolution of Program Size



Note evolution continues after 1st solution found in generation 22 and even after 1st population where everyone has maximum fitness (generation 312).

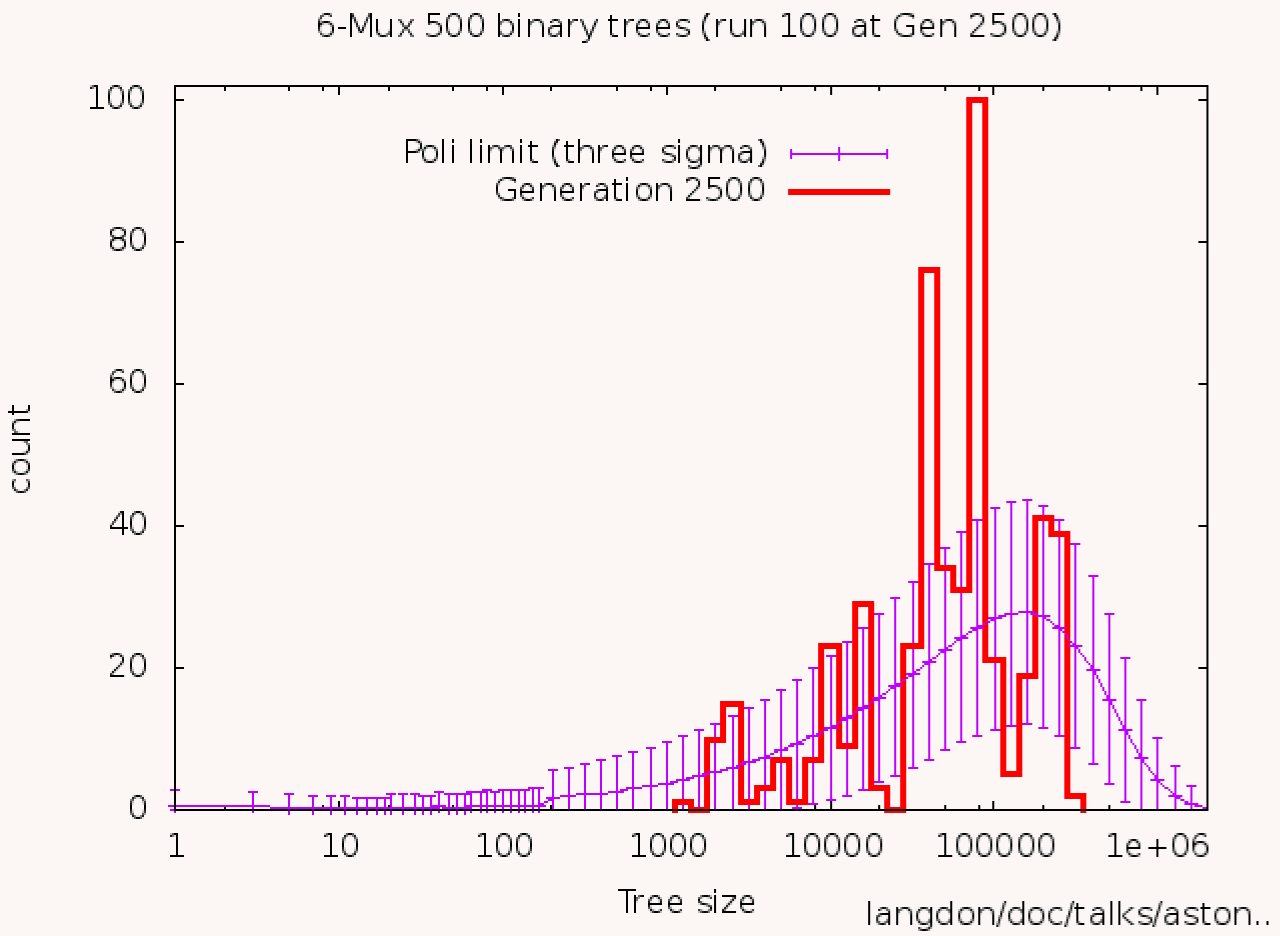
[GP+EM (1)1 pp95-119](GP+EM (1)1 pp95-119)

Evolution of Program Size



Note evolution continues even after 1st population where everyone has maximum fitness (generation 312) but falls as well as rises.

Testing Theory



- Theory assumes crossover only (no selection). In EuroGP2007 distribution of sizes converged to limit rapidly.
- Selection caused by a few runts modifies size distribution

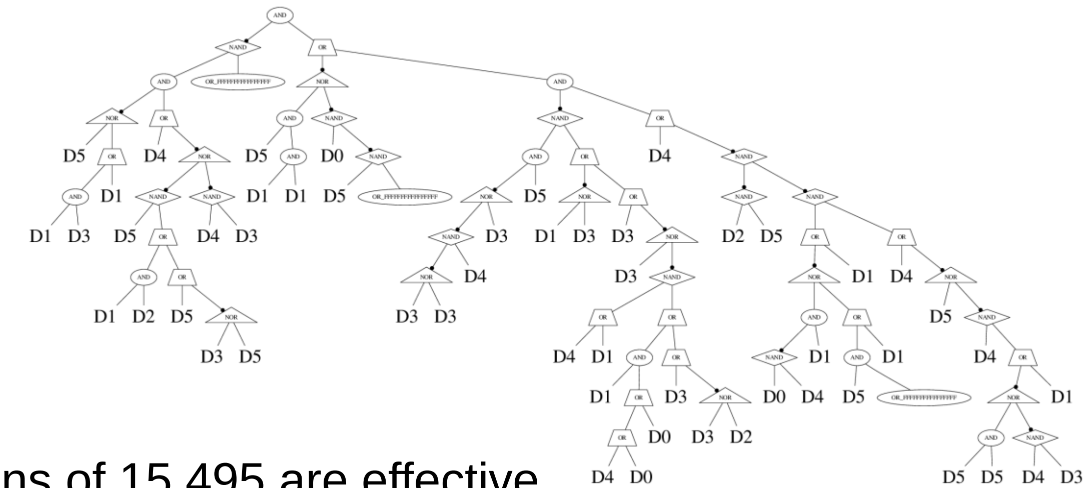
CREST

Convergence in Genetic Programming

- GP genotypes typically do not converge. Even after many generations every tree in the population is different, BUT…
- Every (or almost all) trees give the same answers (phenotypic convergence)
- Effective code, i.e. code to solve problem, does converge.

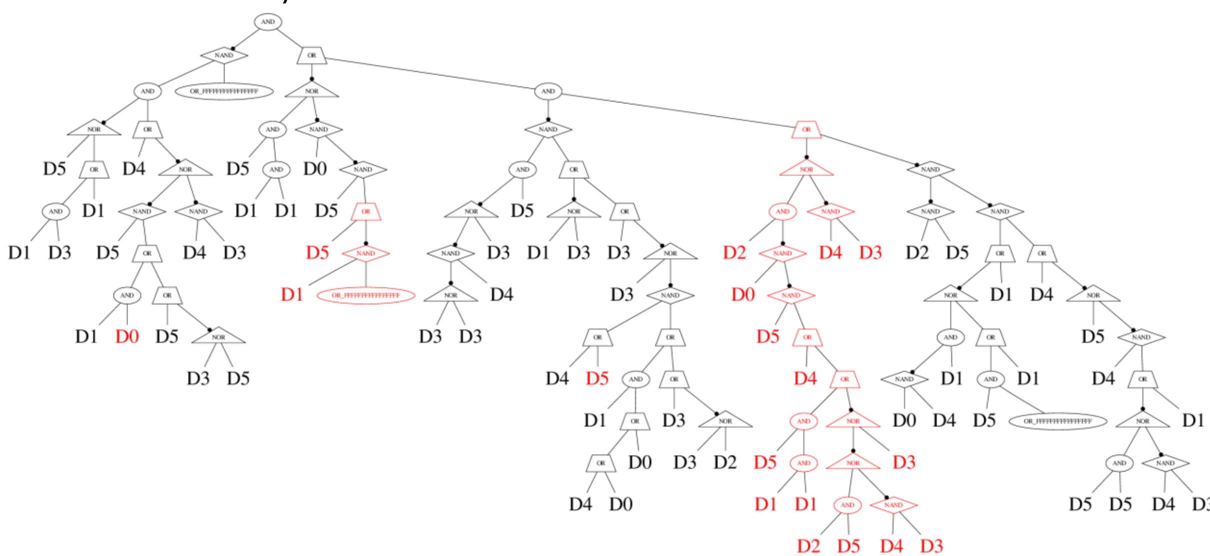
Effective code other runs converges differently

Convergence of typical Effective Code



Gen 400

Only 111 instructions of 15,495 are effective



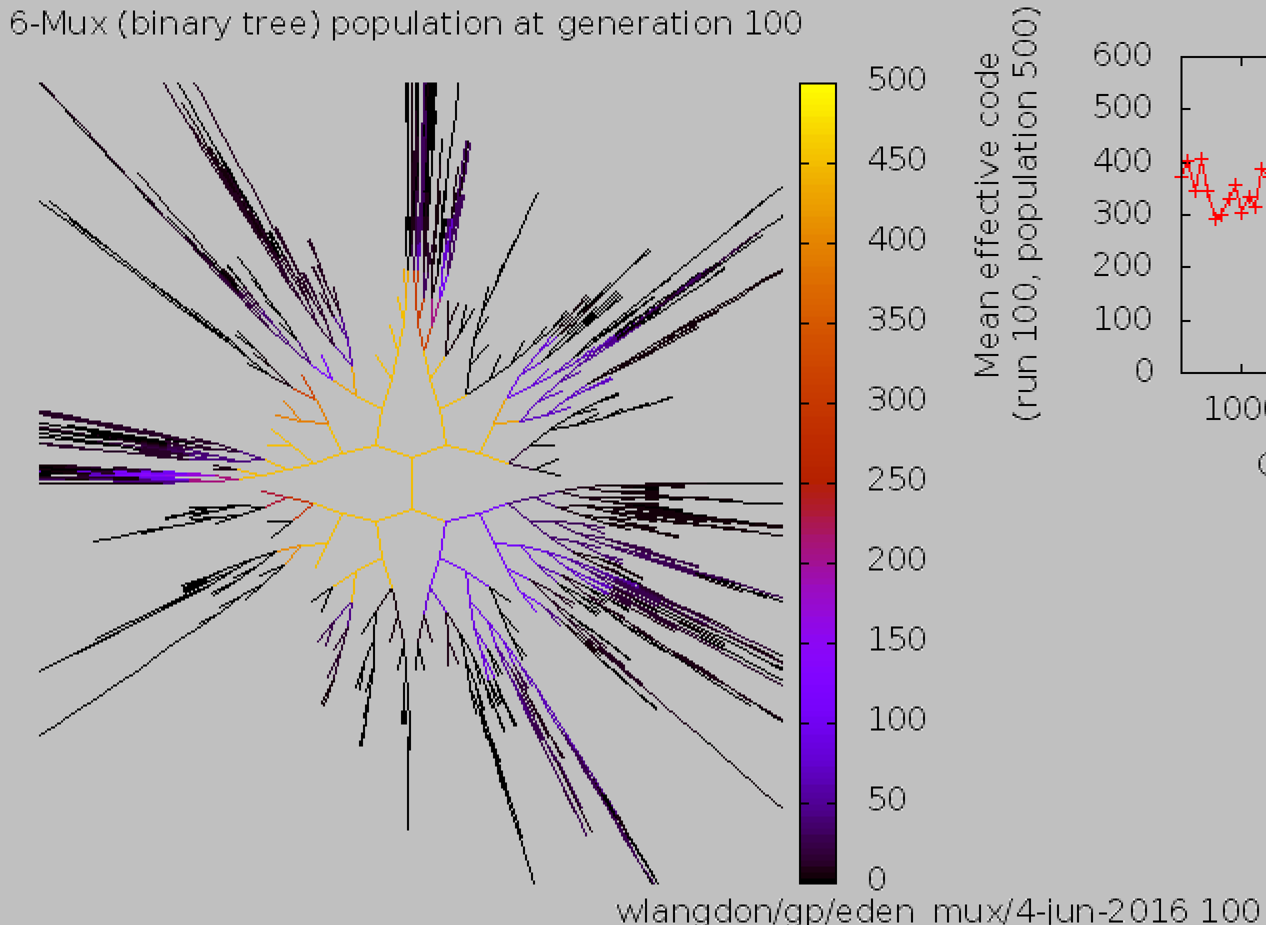
Gen 500

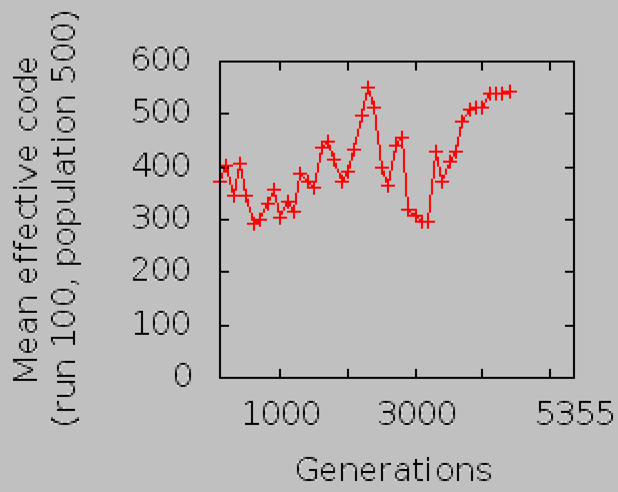
Only 141 instructions of 16,831 are effective

Tree drawing code [lisp2dot.awk]()

CREST

Convergence of Effective Code



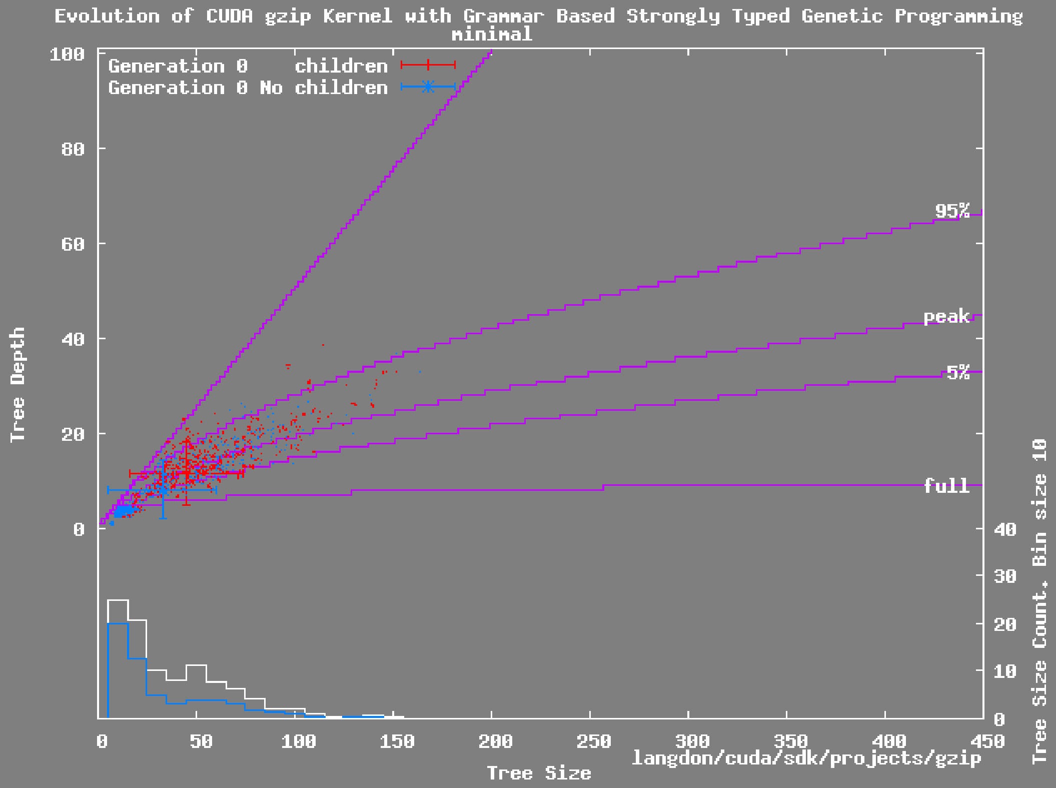


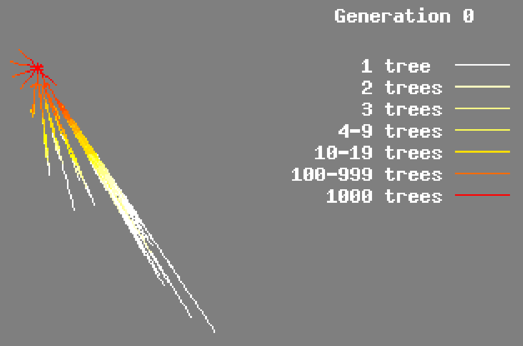
Effective code only. Yellow highly converged.

Black unique code

Circular lattice code [gp2lattice.awk]

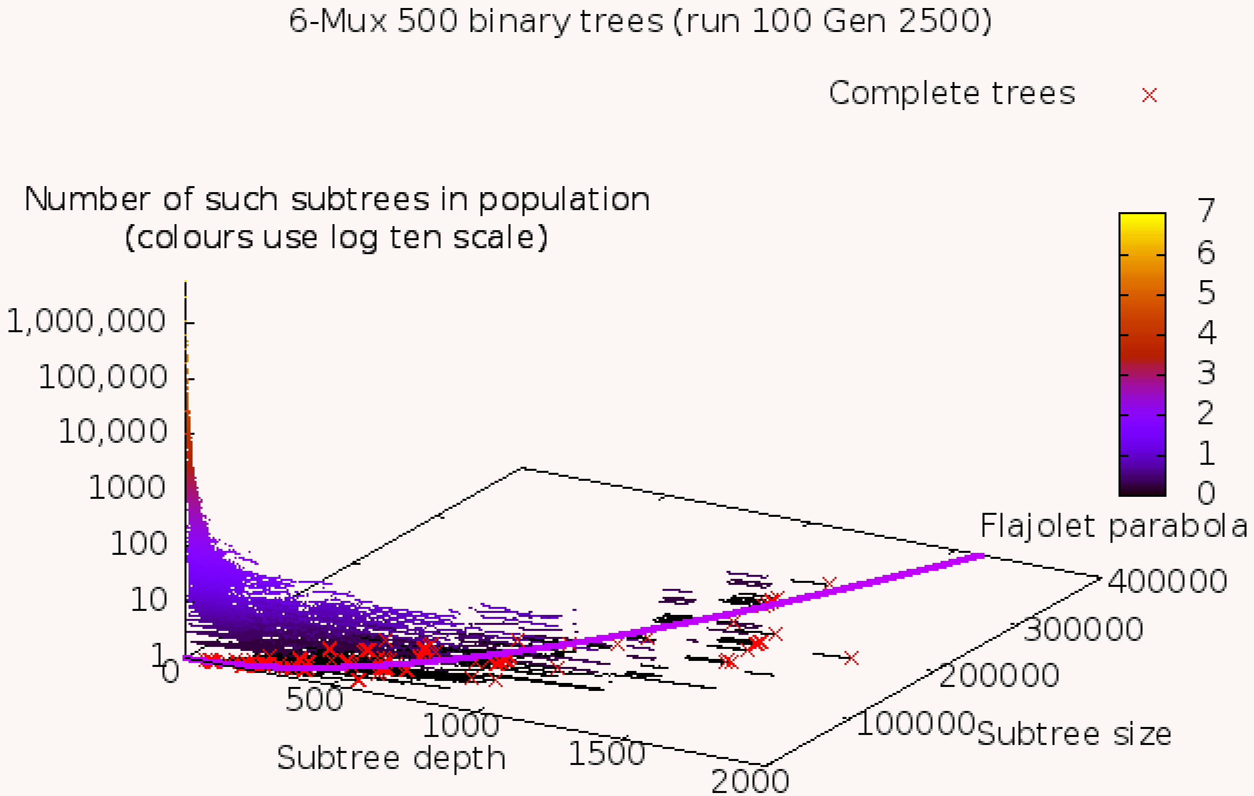
Evolved Trees Random Shapes





Plot whole tree (different population)

Shapes of Evolved Trees



Both whole trees × and subtrees lie near Flajolet $\text{Depth} \approx 2\left(\frac{\pi\, size}{2}\right)^{½}$ limit[1] for random trees

Bloat limited by Gambler's ruin

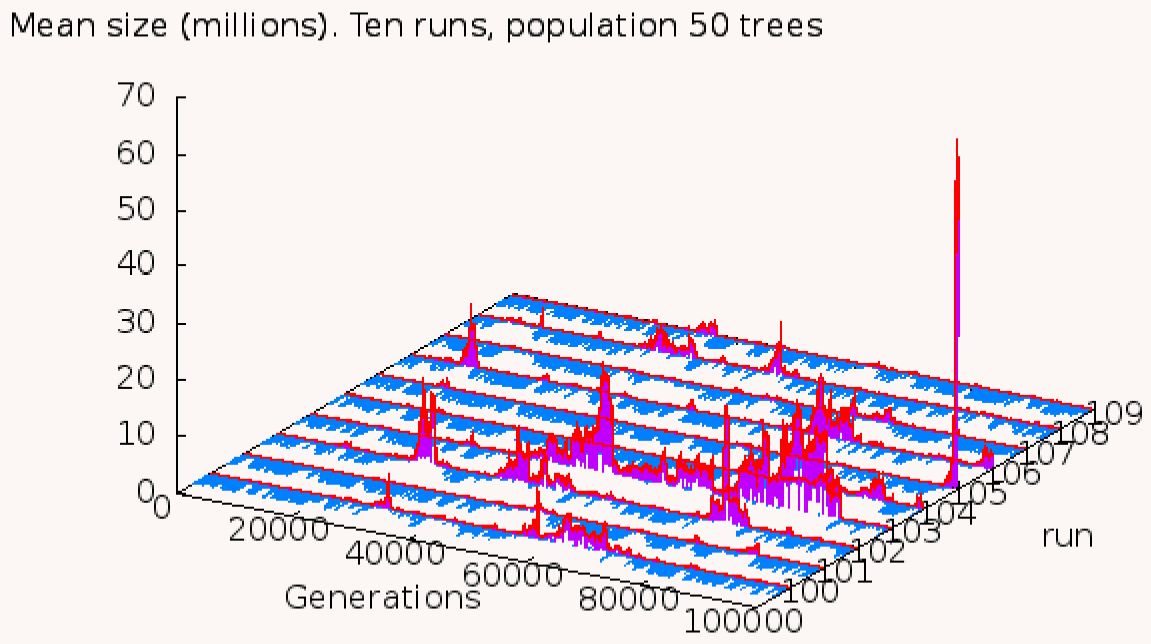
- Tiny fraction of disrupted (low fitness) children sufficient to drive evolution towards every bigger trees.
- As trees get bigger chance of hitting protected effective code near root node falls.
- In a finite population eventually no child will be disrupted.
- Size, without fitness, difference just wanders at random.
- Crossover cannot escape from population of tiny trees.
- So we have a lower limit on the random fluctuation.

  I.e. a Gambler's ruin.
- But wondering towards lower limit will re-establish the conditions for bloat.
- Very approximate limit on tree size:

  tree size ≈ number of trees × core code size

CREST

Bloat limited by Gambler's ruin

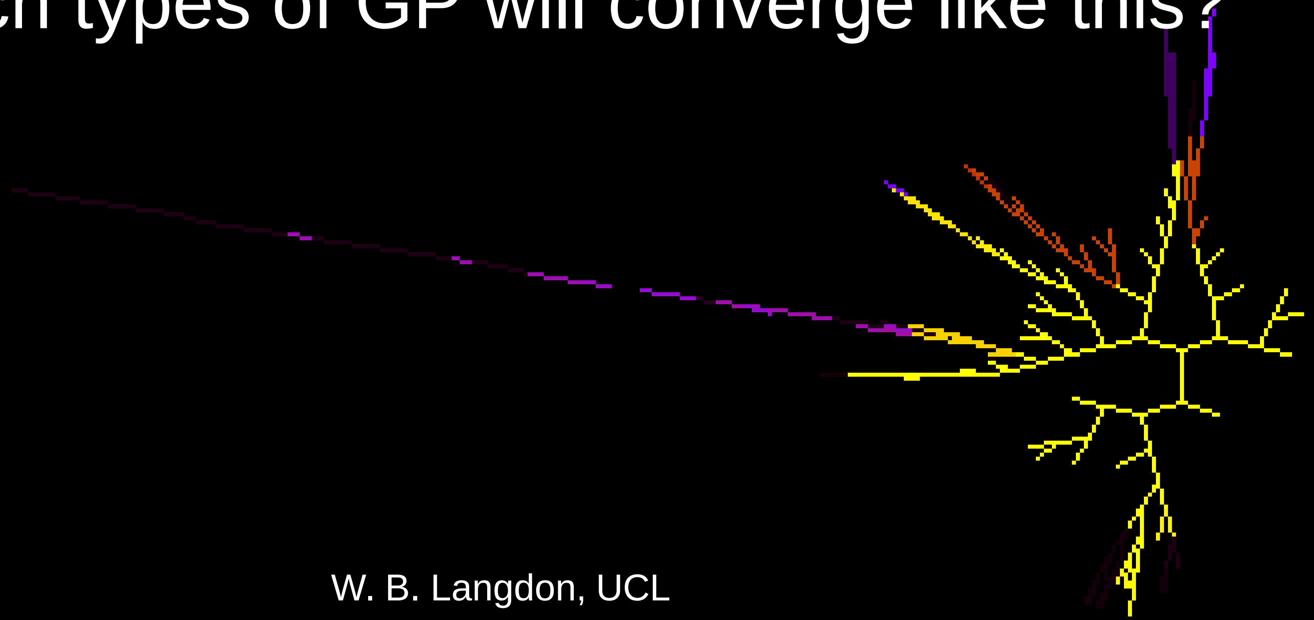


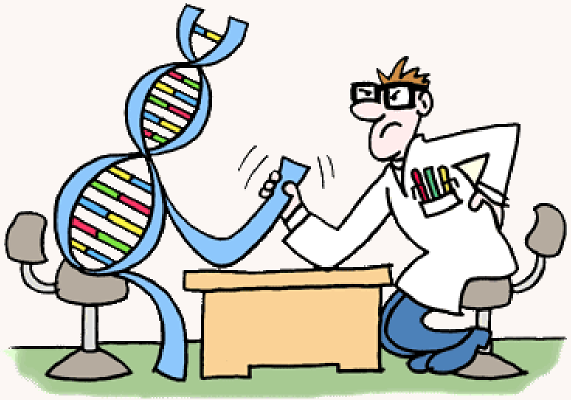
- tree size ≈ number of trees × core code size
- tree size ≈ 50 × 497 ≈ 25000
- Across ten runs and 100,000 generation, median mean size 42,507 (smallest tree in pop size=10,513)

In all ten runs the whole population repeatedly collapses towards smaller trees

Questions

- Next:
  - – Why quadratic increase in size < gen 350
    - Existing theory
  - – differences from crossover only limit
  - – Formalise random drift
  - – Which types of GP will converge like this?





Humies
$10000 Human-Competitive Results

END

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

http://www.epsrc.ac.uk/ EPSRC

Genetic Programming

W. B. Langdon

CREST

Department of Computer Science