Generalisation in Genetic Programming

Poster page 205. Full text RN/11/10

William B. Langdon

Dept. of Computer Science, CREST centre, University College London, Gower Street, WC1E 6BT, UK

W.Langdon@cs.ucl.ac.uk

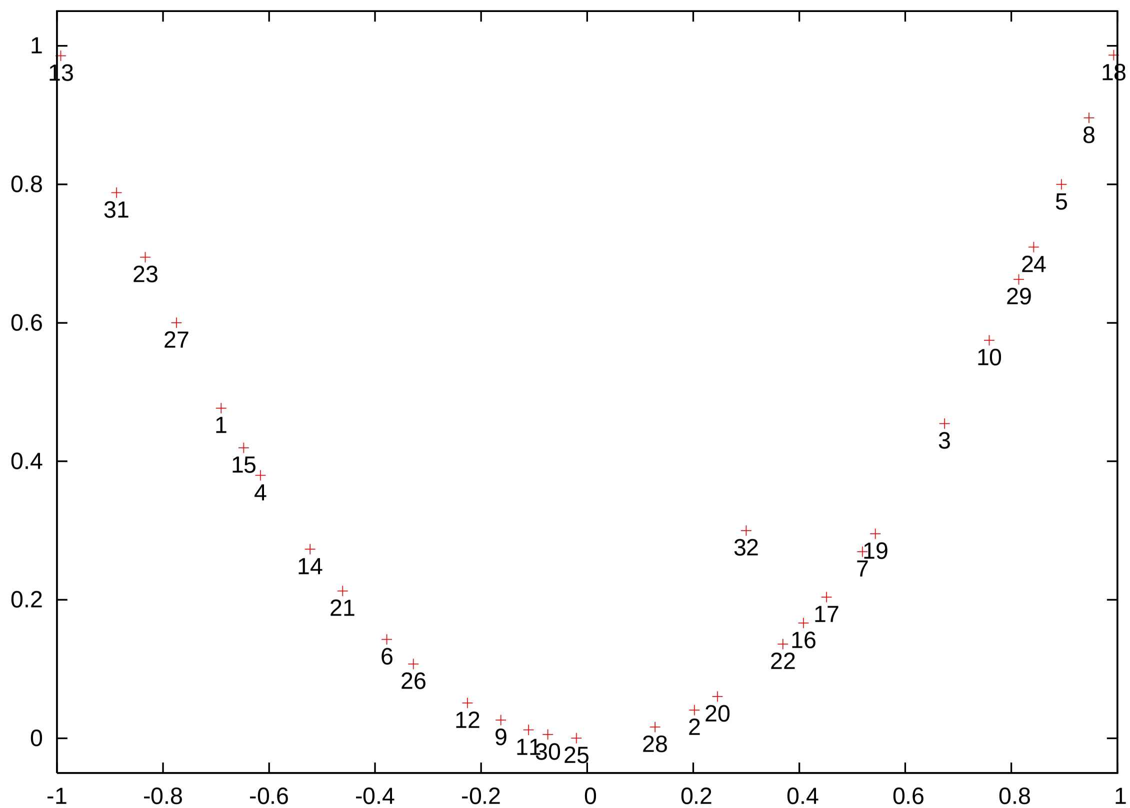
Fitness Sub Samples

- We make fitness cheaper by running fewer tests
- Fewer tests means less accurate fitness
- As long as no bias, evolution can proceed

11-Mux	$2^{11}=$	2048 solved by	1	test
20-Mux	$2^{20}=$	1 048 576 solved by	32	tests
37-Mux	$2^{37}=$	137 438 953 472 solved by	8192	tests

- For the larger problems there are test cases which nobody in the whole course of the evolution has seen.
- Yet the evolved solutions pass them.
- GP has extrapolated from training data to unseen cases.

Counter Example



Thirty one points in a parabola.

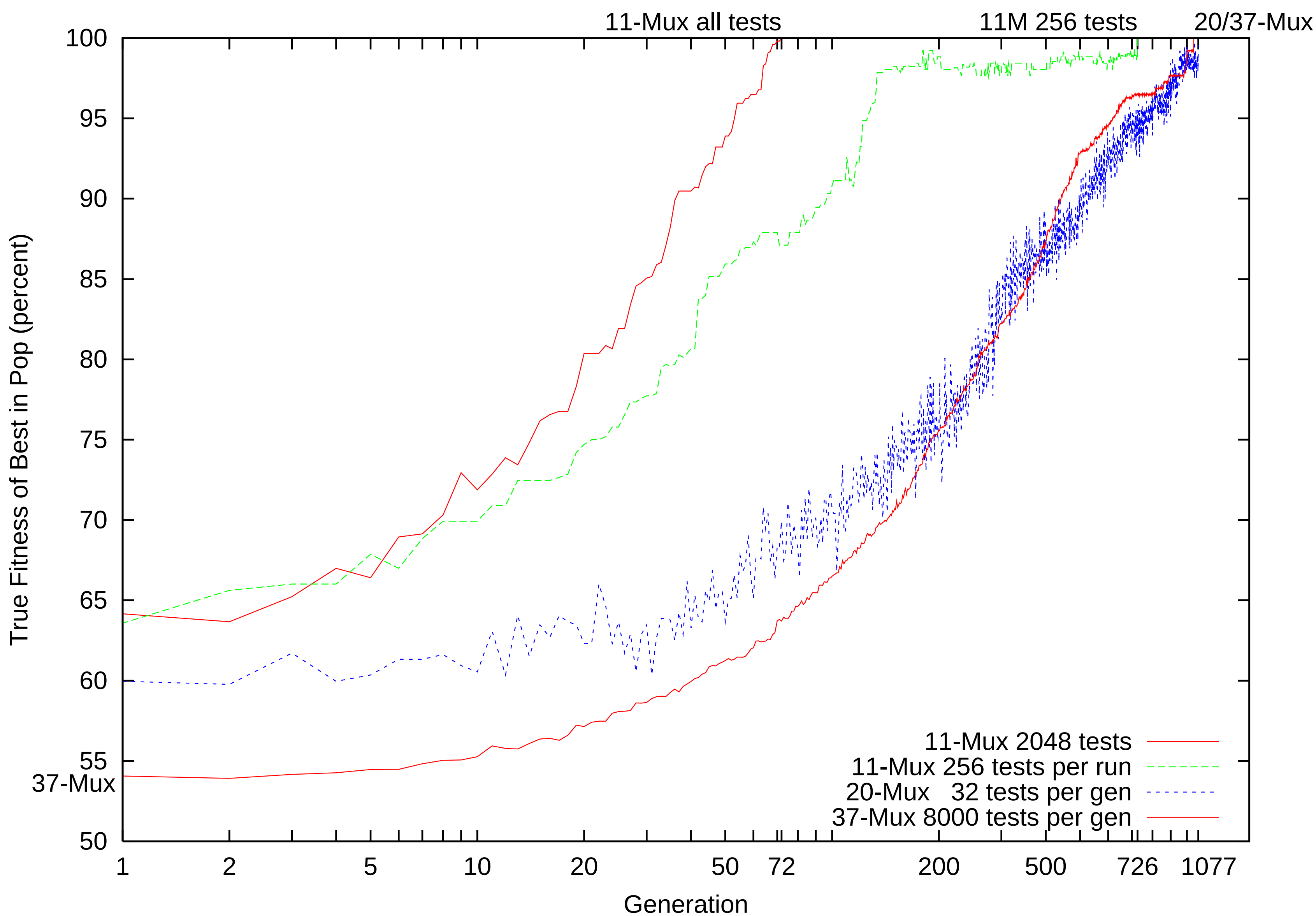
One point not on the parabola.

GP Parameters Multiplexor Small Test Cases

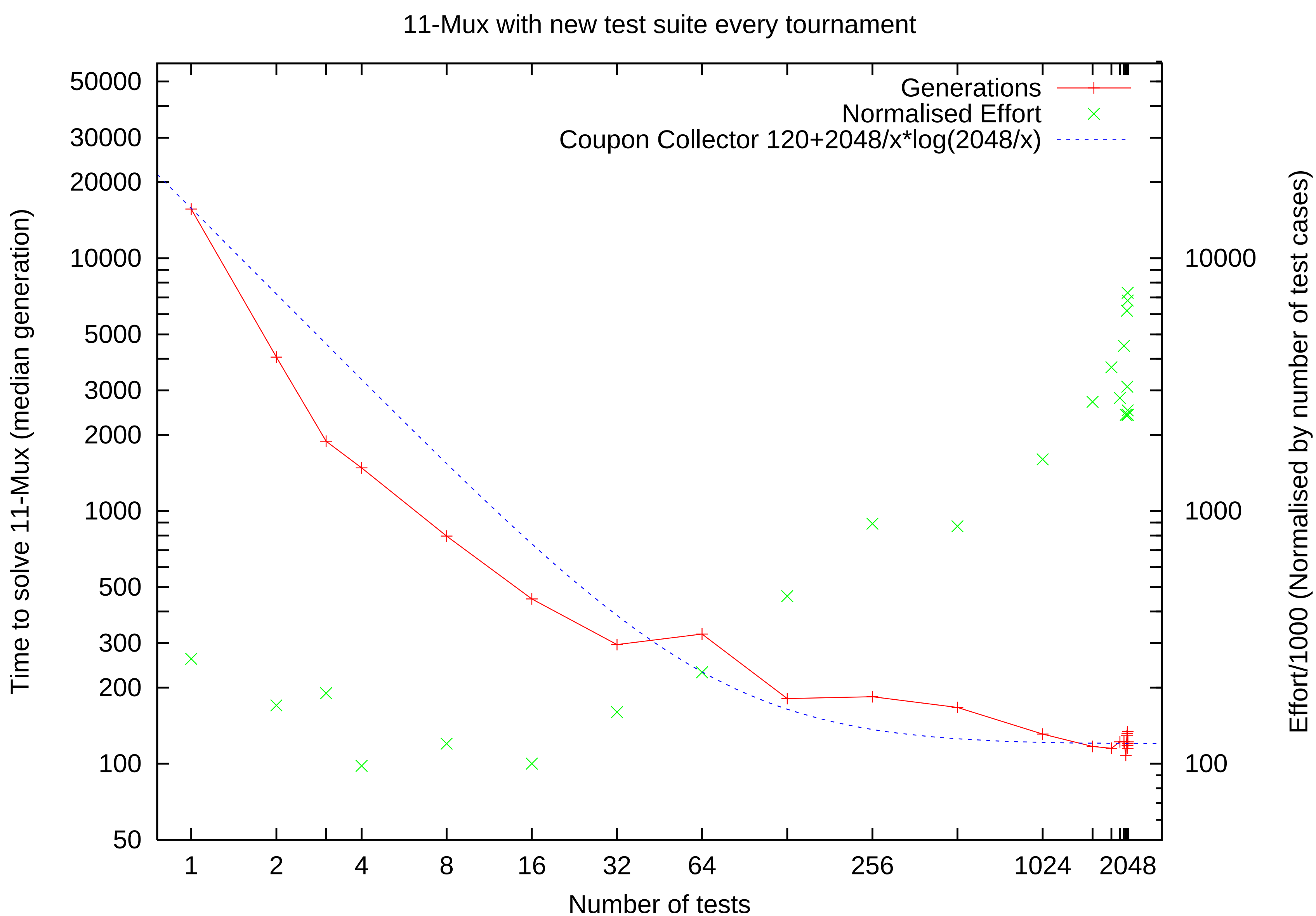
Terminals:	Boolean inputs A0 A1 A2, D0 D1 D2 D3 D4 D5 D6 D7
Functions:	AND, OR, NAND, NOR
Fitness:	Pseudo random samples of from 0 to 2048 fitness cases.
Fit 20-Mux:	Samples of 32 of 1 048 576 fitness cases.
Selection:	Generational, with 4 members tournaments. Tournaments run on same random sample. 1) Fixed sample, 2) new samples for each generation and 3) new samples for each tournament.
Population:	16 384 (20-Mux: 262 144)
Initial pop:	Ramped half-and-half 4:5
Parameters:	50% subtree crossover, 5% subtree mutation, 45% point mutation. Max depth 15, max size 511.
Termination:	first solution or 2 000 gens [extended runs 50 000 gens or pop is only leafs]

Code at FTP `cs.ucl.ac.uk` `genetic/gp-code/gp32cuda.tar.gz`

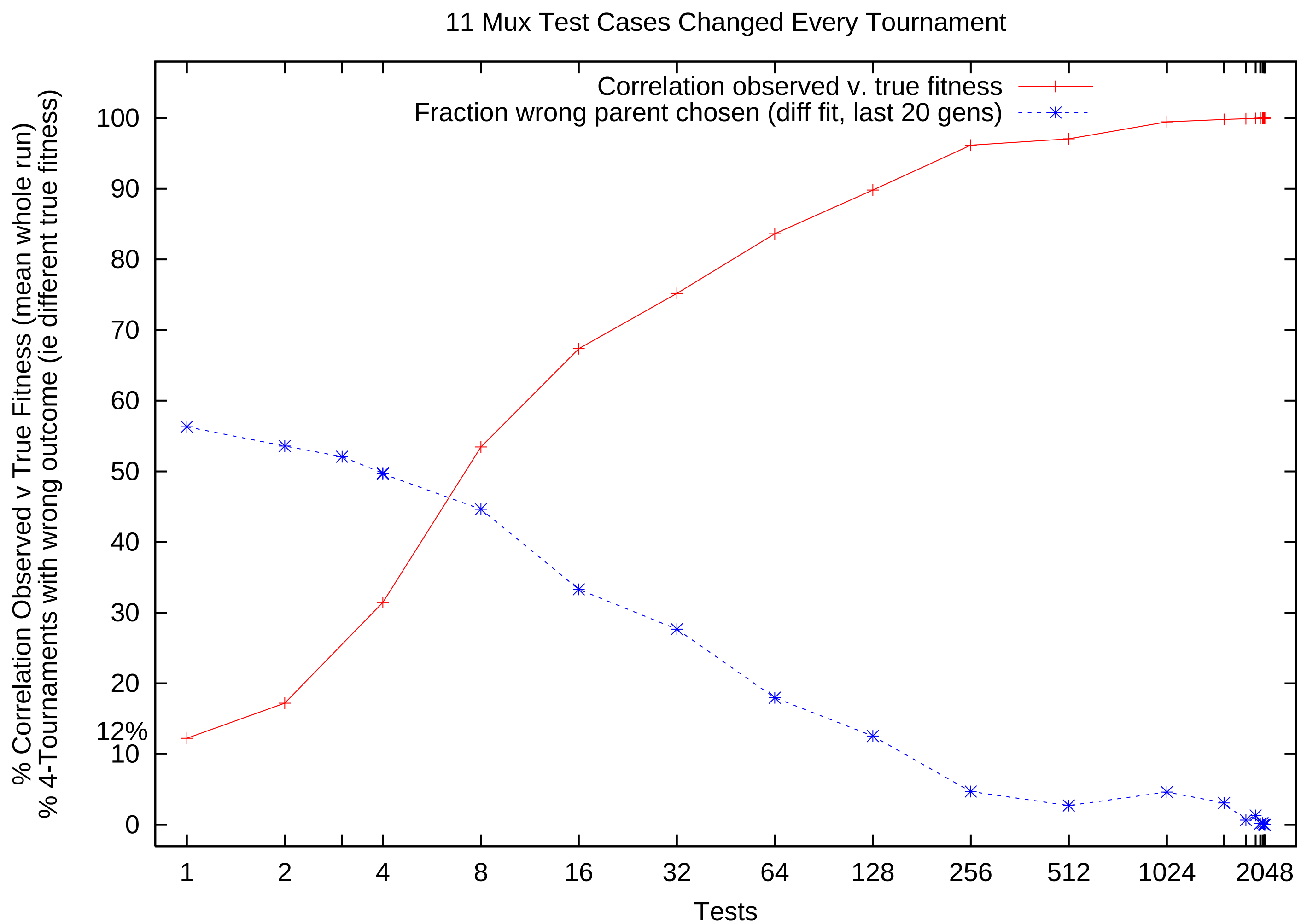
Example Successful Runs



Varying number of Test Cases



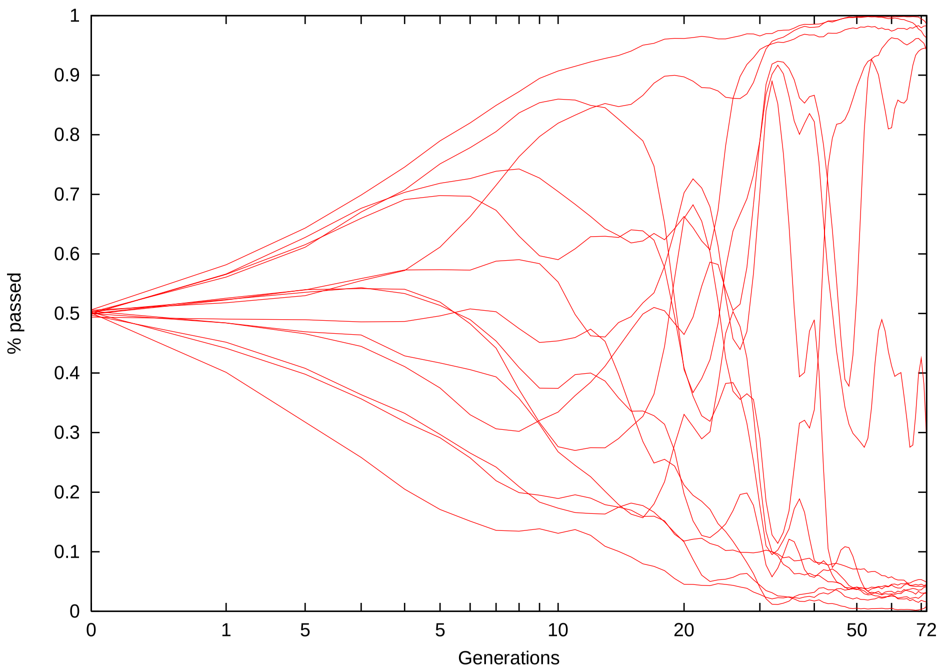
Correlation with True Fitness



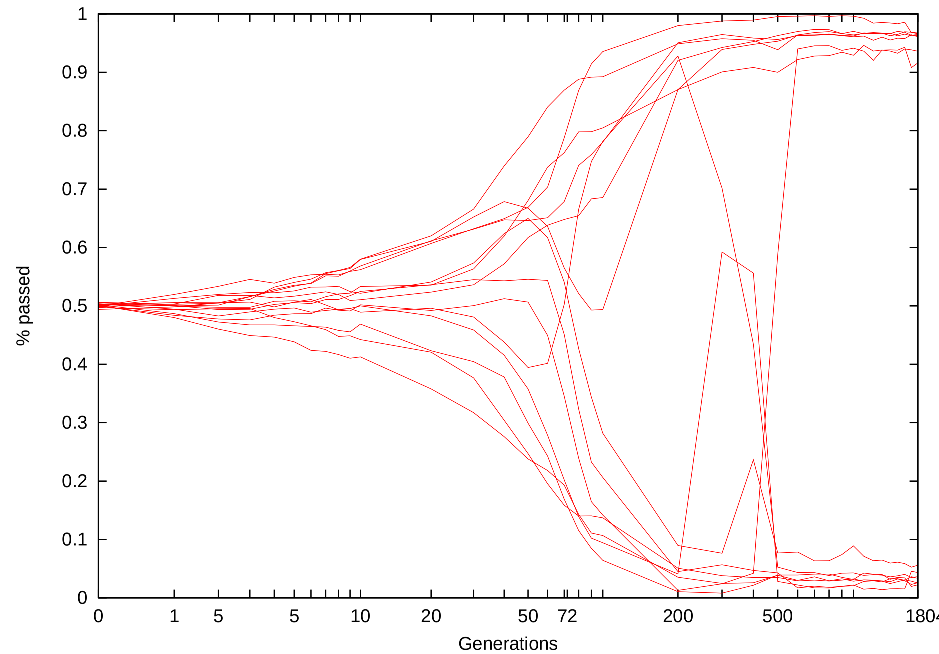
Reducing testing below 256 causes more than 5% errors. But evolution proceeds even with many selection errors.

Phenotypic Convergence

11-Mux All tests



11-Mux 8 of 2048 tests



Apart from speed of change, phenotypic behaviour of population with all tests much like those with random samples of tests.

Solution found despite many tests being failed by most of the population

Conclusions

- Evolution can proceed even when many selection tournaments choose wrongly.
- The population evolves more slowly (Price's Theorem) but mostly in a similar way to when the full test suite is used.
- Multiplexor family are *regular* problems,

  where passing x% of a sample suggests code will pass x% of all tests.

  Not all problems are regular.
- In software engineering when can we assume the software under test is regular?

  I.e. when can we extrapolate from the tests actually run to all possible tests?