Abstract
Information theoretic analysis of large evolved programs produced by running genetic programming for up to a million generations has shown even functions as smooth and well behaved as floating point addition and multiplication loose entropy and consequently are robust and fail to propagate disruption to their outputs. This means, while dependent upon fitness tests, many genetic changes deep within trees are silent. For evolution to proceed at reasonable rate it must be possible to measure the impact of most code changes, yet in large trees most crossover sites are distant from the root node. We suggest to evolve very large very complex programs, it will be necessary to adopt an open architecture where most mutation site are within 10–100 levels of the organism’s environment.

1 Background
Recently we have been investigating the long term evolution of genetic programming. Firstly, using Poli’s submachine code GP [24, 25], evolving large binary Boolean trees [7] and more recently exploiting SIMD Intel AVX and multi-core parallelism to evolve floating point GP [9, 14, 12, 5]. Running for up to a million generations without size limits has generated, at more than a billion nodes, the biggest programs yet evolved and forced the development [10, 11] of, at the equivalent of more than a trillion GP operations per second, the fastest GP system. It has also prompted information theoretic analysis of programs [16]. (Of course information theory has long been used with evolutionary computing, e.g. [2].)

One immediately applicable result has been the realisation that in deep GP trees most changes have no impact on fitness and once this has been proved, for a given child, its fitness evaluation can be cut short and fitness simply copied from the parent. This can lead to enormous speed ups [13].

We have also considered traditional imperative (human written) programs and shown these too are much more robust than is often assumed [15, 6, 8]. Indeed we suggest that information theory provides a unified view of the difficulty of testing software [17, 11, 22].

The question of why fitness is so often exactly inherited [20, 21, 23] despite brutal genetic change is answered by the realisation that without side effect the disruption caused by the mutation must be propagated up the tree through a long chain of irreversible operations to the root node. Each function in the chain can loose entropy. In many cases deeply nested functions progressively loose information about the perturbation as the disruption fails to propagate to the program’s output. Thus the mutation becomes invisible to fitness testing and its utility cannot be measured. Without fitness selective pressure, evolution degenerates into an undirected random walk.

In bloated structures information loss leads, from an evolutionary point of view, to extremely high resilience, robustness and so stasis. From the engineering point of view this is problematic, as then almost all genetic changes have no impact and evolutionary progress slows to a dawdle.

Since all digital computing is irreversible [18] it inherently losses information and so without short cuts, must lead to failed disruption propagation (FDP) [22]. We suggest in order to evolve large complex systems it must be possible to measure the impact of genetic changes, therefore we must control FDP and suggest in the next section that to evolve large systems, they be composed of many programs of limited nesting depth and structured to allow rapid communication of both inputs and outputs to the (fitness determining) environment.
2 Open Complex System

Can we evolve systems more like cell interiors, with high surface area membranes composed of very many small adjacent programs each of limited depth placed side by side. The membranes forming an open structure with many gaps between them. The gaps themselves supporting rapid communication (which might be implemented using global memory or enhanced communication links or buses) with no, or little, processing ability and consequently little information loss.

Figure 1 shows 1300 programs arranged in an open structure (based on fit Sextic polynomial [4] trees with average height 9.22, quartile range 7–11).

Rich Lenski, in his long-term evolution experiment (LTEE) [19, 3] has demonstrated that Nature can continue to innovate in a static environment for more than 75000 generations. We have shown GP can do similarly for at least 100000 generations. However when evolving deep structures, progress slows dramatically and therefore we feel monolithic deep structures will not be sufficient to automatically evolve complex systems. Instead an open structure like Figure 1 may be needed.
References


