DOI: http://dx.doi.org/10.7551/978-0-262-31709-2-ch179

How Fast Can We Evolve Something?

Inman Harvey

Evolutionary and Adaptive Systems Group
University of Sussex, Brighton, UK
inmanh@gmail.com

Abstract

When applying a Genetic Algorithm to a new problem, how many generations should one reasonably expect to wait before reaching an acceptable solution? Using arguments based on information flow into the genepool, mediated by selection each generation, we present a rough and ready heuristic.

The Problem

You have encoded a new problem for solution by Genetic Algorithm (GA) in what you hope is a sensible manner, using a population size and mutation rates that appear appropriate. Yet after 10,000 generations you are nowhere near reaching an acceptable solution; was that long enough to wait?

The textbooks and standard literature provide surprisingly little advice on this crucial matter. Using arguments inspired by information-theoretic approaches to evolution (Kimura, 1961; Worden, 1995), we present a crude Rule of Thumb. Primarily we are concerned with complex problems associated with a fixed high-dimensional fitness landscape, but we start by recasting the problem in terms of a childrens' game.

The Game of Twenty Questions

For convenience, suppose there is a pre-agreed list of 2^{20} objects from which Player 1 secretly picks one as the target. Player 2 may ask 20 binary questions ('animal or not?', 'bigger than a brick or not?') to try and identify the target; what is the best strategy? Ideally, each successive question should divide the currently remaining pool of possibilities exactly in half, so that Player 1's response reduces the remaining uncertainty of Player 2 by the maximum possible 1 bit of Shannon information. Any question that fails to divide the pool into halves will provide less information and may leave some remaining uncertainty after the 20^{th} question.

This can be reformulated into a simplistic asexual GA. With a population size of 2^{20}, the initial generation consists of one example of every possible target (that could be represented by different binary genotypes length 20). Each question corresponds to a generation with a round of truncation selection where the 'fitter' half of the population is selected, and then duplicated (without mutation) to maintain the same population size. After 20 such generations, if the selective 'questions' have been ideally posed, the final population should consist of 2^{20} copies of the desired target. Clearly the very maximum amount of information, passed through such 50% truncation selection into the population (or its genepool), is 20 bits in 20 generations.

A more realistic asexual GA will achieve significantly less than this 1 bit information flow per generation. Selection is unlikely to produce the ideal 50/50 split; also the population will typically be far smaller than the complete genotype space. Thus mutation must partially compensate by expanding the population each generation beyond the limited set of possibilities within the initial small population; this is at the expense of some information being lost from the genepool each generation. Hence, as Worden (1995) points out, the figure of 1 bit per generation counts as a speed limit, like the speed of light, that is an idealized upper bound; in practice, significantly fewer bits per generation are achievable.

Higher rates of selection such as 25% truncation selection would result in 2 bits per generation as this upper speed limit. But since there are other reasons (such as avoiding getting trapped on local optima) that tend to make higher rates undesirable, we focus here on the paradigm case of 50% truncation selection. This is equivalent to the most basic forms of tournament selection (Harvey, 2011).

Redundancy. The example of binary genotypes of length 20 does indeed imply a search-space of size 2^{20}. However in evolutionary search we are often not searching for just one target genotype, but any one of an equivalence class that count as equally fit. In the game of 20 Questions, if the 2^{20} possible targets actually fell into say 2^{10} different equivalence classes, and the second Player can win by identifying the equivalence class rather than the unique target, then the initial uncertainty can be counted as 10 bits rather than 20 bits. This clearly reduces the number of Questions (or generations) needed.

Does Recombination Change the Picture?

The implication so far is that an evolutionary problem phrased in terms of binary genotypes of length L should expect to need a minimum of L generations, adjusted by two estimated fudge factors. The first such factor significantly increases the generations needed; the role of a relatively small population and the loss of information due to mutations incurs considerable costs. The second such factor, arising from an assessment of the likely degree of genetic redundancy, may partially compensate by tending to decrease the generations needed. So far this assessment is based on an asexual GA.

But GAs are typically sexual, and many will expect the addition of recombination to provide a significant speed-up. Indeed, under some circumstances it definitely will.

Mackay (1999) argues that, under particular assumptions, recombination speeds up the asexual rate of 1 bit per generation (where he agrees with the argument above) to something of order $\sqrt{L}$ bits per generation, where L is the length of a binary genotype. This is a massive speedup, and simulations based on those assumptions broadly confirm experimentally this factor that Mackay derives analytically.

But these assumptions are completely unrealistic for any problem where GAs are likely to be used. He assumes that fitness is a strictly additive trait; i.e. no epistasis with each locus on the genotype contributing fitness independently of the values at other loci. This can be considered a 'Mt. Fuji' fitness landscape; for such simple scenarios there are much more effective search algorithms than any standard GA.

For instance, consider a population of L clones of some randomly chosen genotype. Produce L new offspring, where the i^{th} one differs from the parent at the i^{th} locus, and compare the fitness of each one with its parent. This immediately identifies the fittest allele at each locus, and allows you to identify the peak of Mt. Fuji in a single generation. No real world GA application comes anywhere close to corresponding to such a simplistic fitness landscape, yet such assumptions of additive fitness are quite often used when biologists do theoretical population genetics. Why is this?

Micro-evolution and Macro-evolution. I believe the answer is that biologists very commonly consider issues of *micro-evolution*: they consider an already-evolved species of complex organisms and its ongoing 'struggle' to maintain its fitness through natural selection in the context of continual environmental and competitive challenges that tend to decrease that fitness. Loosely speaking, the fitness of any species, whether single celled amoeba or far more complex mammal, must be around unity if it neither goes extinct nor increases without limit. But this is like maintaining height through constantly climbing up (via natural selection) a downward escalator driven down by 'deterioration of the environment' in its broadest sense. This is the picture given by Fisher's Fundamental Theorem (Fisher, 1958), especially as clarified by Price (1972).

In such a scenario, where any evolved phenotypic trait probably depends on the sum of a number of genetic contributions, then the micro-evolutionary recovery from environmental deterioration may well justify the biologists' assumption of additive fitness. But the scenario faced by those using evolutionary algorithms is usually very different.

They are typically interested instead in *macro-evolution*: the equivalent of mammal species developing from unicellular species over billion of years. If this is characterized by extended periods of stasis, punctuated by individual innovations, for instance in the context of Neutral Networks (Harvey, 2001), then the additive fitness assumption ceases to make sense. One can agree (Ochoa and Harvey, 1999) that the more correlated a fitness landscape is, the more advantageous it is to add recombination. One can agree that recombination is still worth adding to a GA; its role appears largely to be mitigating the negative effects of mutation. But unless one buys into the wholly unrealistic assumptions of Mackay (1999) and others, in the absence of compelling evidence to the contrary it seems prudent to assume that macro-evolution has the speed limit associated with an asexual GA.

The Rule of Thumb

Let us first assume binary genotypes, and some standard GA with the equivalent of 50% truncation selection, recombination and appropriate rates (Harvey, 2001) of mutation. If your search-space corresponds to a 20-bit genotype, then 20 generations (at 1 bit per generation) is your starting point. This may be reduced by an estimate of the redundancy; if one plausibly thought that phenotype space partitioned into say 2^{10} equivalence classes, this reduces the figure to 10 generations.

But then very significant allowance must be made for the relatively small population you are likely to be using, and the loss of information due to mutations. One order of magnitude would seem conservative for this, and I tentatively allow 2 orders of magnitude. I.e., for a 20-bit problem with 50% redundancy, I would recommend 20*0.5*100 = 1000 generations. For a steady-state GA with population size P, this is equivalent to 1000*P new individuals.

Real-valued genotypes. When evolving (e.g.) neural networks, the weights may be encoded in doubles or floats rather than bits on the genotype. As a first approximation, decide whether e.g. 4-bit or 6-bit precision is likely to be adequate for the job, and this gives a direct conversion to the binary case. With a feed-forward network having, say, 3 nodes in the hidden layer, one should note that any perfect network is functionally equivalent with any relabeling of the hidden nodes, and this would give an immediate redundancy factor of 3! = 6, and all such redundancy is helpful.

Caution. This heuristic suffers from many imperfections, from lack of knowledge of the details of any particular case, and from the need to make imprecise estimates. It should be treated with caution. Nevertheless, until more principled estimates are proposed it seems better than nothing.

References

Fisher, R. A. (1958). *The Genetical Theory of Natural Selection.* 2nd ed. New York: Dover Publications.

Harvey, I. (2001). Artificial evolution: a continuing SAGA. In Gomi, T., editor, *Evolutionary Robotics: From Intelligent Robots to Artificial Life,* LNCS 2217, pages 94-109. Springer.

Harvey, I. (2011). The microbial genetic algorithm. In Kampis, G., Karsai, I. and Szathmáry, E., editors, *Advances in Artificial Life: ECAL 2009,* Part II, LNCS 5778, pages 126-133. Springer.

Kimura, M. (1961). Natural selection as the process of accumulating genetic information in adaptive evolution. *Genetical Research*, 2(1):127-140.

Mackay, D. J. C. (1999). Rate of information acquisition by a species subjected to natural selection. Available online at www.inference.phy.cam.ac.uk/mackay/gene.pdf

Ochoa, G. and Harvey, I. (1999). Recombination and error thresholds in finite populations. In Banzhaf, W. and Reeves, C., editors, *Foundations of Genetic Algorithms* 5, pages 245-264. Morgan Kauffman, San Francisco, CA.

Price, G. R. (1972). Fisher's 'fundamental theorem' made clear. *Ann. Hum. Genet. Lond.,* 36:129-140.

Worden, R. P. (1995). A speed limit for evolution. *J. Theor. Biol.*, 176(1):137-152.