1 Why does a GA work?

1.1 Introduction

By what means is a GA an effective algorithm for search or optimisation? John Holland (1975) introduced the concept of hyperplane sampling to describe the search performed by a GA.

Starting with a trivial example case, imagine a simple 3-dimensional space encoded by a 3-bit bitstring. This may be illustrated by a cube with each vertex enumerated with its corresponding bitstring value, as shown in figure 1.

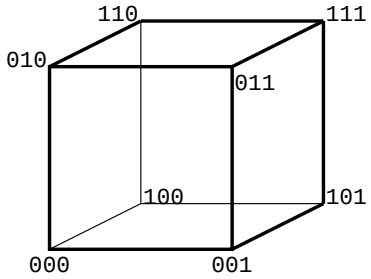

Figure 1: The 3-dimensional space which may be encoded with a 3-bit bitstring.

The labels for each adjacent pair of vertices differ by exactly 1-bit. The frontmost face has corners labelled with a "0" in the first bit. Using wild-card notation, we may express this as "0**". Bitstrings containing "*"s at loci which may contain any value are termed *schemata*. Each schema corresponds to a particular hyperplane in the search space. The number of fixed bits in a schema is referred to as that hyperplane's *order*. i.e. "*1*" indicates an order-1 hyperplane while "0*0" indicates an order-2 hyperplane.

All bitstrings which are consistent with a particular schemata are contained in the hyperplane represented by that schemata, e.g. any bitstring which has individual bits that match the fixed bits in the schemata and any values for the unspecified bits will lay on a single hyperplane.

If N is the length of the binary bitstring (the chromosome) which encodes the search space, then there will be $2^N - 1$ hyperplanes in that space. (At least one bit must be specified. The string of all "*"s does not represent a plane, it represents the entire space instead.)

Holland noted that the GAs implicit parallelism is derived from the fact that many hyperplanes are simultaneously sampled when a population of bitstrings is evaluated. Looking at a single point in the search space does not give much information, but looking at an entire population of points samples many more hyperplanes than bitstrings in that population, thus yielding the statistical information used in the optimisation of that population. This is the fundamental property which results in the efficiency of GAs.

Through the process of reproduction and recombination, the schemata of the various hyperplanes increase or decrease in propensity in the population according to the relative fitness of the strings representing points on those hyperplanes. Many hyperplanes are solved simultaneously, hence the term *implicit parallelism* is often used when describing GAs.

Even though a GA never explicitly evaluates any hyperplane partition, the distribution of bitstring values change over a series of generations as if it had.

As a consequence of the above considerations, it can be seen that the sampling of hyperplane partitions is not significantly affected by local optima. Likewise, increasing the sampling of hyperplane partitions exhibiting better than average fitness does not guarantee convergence on the global optimum. For example, when the global optimum is a relatively isolated peak then locating this will be difficult.

When bitstrings are simply duplicated, through the application of the selection operators, no new sampling of hyperplanes will occur. The mutation and crossover operators provide new sample points whilst partially preserving the distribution of strings belonging to the sampled hyperplanes.

1.2 Crossover Operators

Order-1 samples are not affected by crossover since the single significant bit is always inherited by one of the offspring. Order-2, and higher, samples do have their distribution modified by the crossover operation. The possibility that the bits in the schema "11****" will be separated during single point crossover is $\frac{1}{N-1}$ where $N = 6$ in this example. For the schema "1****1", the probability is $1.0 \left(\frac{N-1}{N-1}\right)$ because every possible crossover locus will disrupt the schema. This shows that all hyperplanes of the same order are not necessarily disrupted with the same probability. i.e. the position of the bits in a schema is an important factor in the probability of that schema surviving crossover.

For 2-point crossover, maximum disruption for order-2 schema occurs when the 2 significant bits are separated by $\frac{N-1}{2}$ bits.

Thus the probability of disruption caused by crossover is operator dependent.

A *compact representation* is a term used to describe a given representation (or ordering of bits) which minimises the probability of disruption of a given schema by a given operator.

Dependent on the representation and operators being used, particular sets of bits are more likely to be inherited as a group. This is known as the *linkage* phenomenon. For commonly used representations and operators, linkage usually equates with the distance between the bits on a string. This may be readily measured as the *defining length*, which is the index of the rightmost fixed bit minus the index of the leftmost fixed bit. This is the number of possible crossover loci which would disrupt the schema. The defining length of a schema representing a hyperplane, H, is often written as $\Delta(H)$. For single-point crossover, it is obvious that $\frac{\Delta(H)}{N-1}$ is the probability of that schema being disrupted.

1.3 Inversion

One further genetic operator introduced by Holland is inversion. Inversion alters the linkage of bits on the chromosome, possibly resulting in moving bits greater interdependence closer on the chromosome.

A typical inversion operator would reverse a randomly selected substring. This obviously requires a position independent encoding scheme. Position independent encoding may be easily implemented through the use of tagged bitstrings. Consider a bitstring with a mask containing an index for each bit. e.g.:

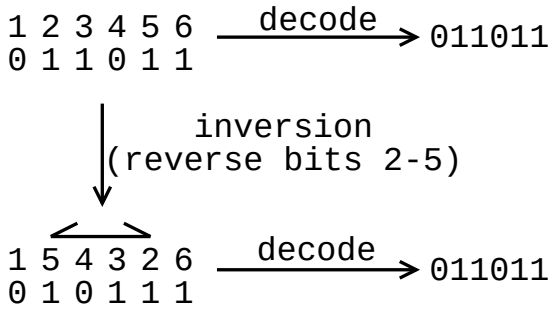

Figure 2: Example of a simple inversion operator. The representation has changed, but the resulting bitstring is not modified.

1.4 The Schema Theorem

Holland introduced the schema theorem, a mathematical framework for consideration of the above ideas. In essence the schema theorem provides a lower bound for the change of sampling rate for a hyperplane during a generation.

When selection occurs, an intermediate population is generated from the initial population. (In GAUL, no intermediate population is explicitly generated since selection occurs 'on-the-fly', but the overall outcome is exactly equivalent). For a hyperplane, H, at generation t:

$$M(H,t^*) = M(H,t)\frac{f(H,t)}{< f >} \tag{1}$$

where t^* indicates that selection has occurred based on the population at generation t, M is the set of solutions in the population belonging to hyperplane H at generation t. f is the mean fitness of the set of solutions, M, and $< f >$ is the mean fitness of the entire population.

This selected population is acted upon by the crossover and mutation operators to produce the population $M(H, t+1)$. First considering just crossover

which is applied to a portion of the population, with the remainder of the population unmodified:

$$M(H,t+1) = (1-P_c)M(H,t)\frac{f(H,t)}{<f>} + P_c\left[M(H,t)\frac{f(H,t)}{<f>}(1-l)+g\right] \quad (2)$$

where P_c is the probability for crossover. l and g are the losses and gains due to the schema disrupted by the crossover. It is assumed that crossover within the defining length of the schema defining H is always disruptive to that schema. In truth, this is not always the case. For simplicity, it is also assumed that gains do not occur. These are both very conservative estimates which will overestimate reduction in sampling of a given hyperplane. So,

$$M(H,t+1) \geqslant (1-P_c)M(H,t)\frac{f(H,t)}{<f>} + P_c\left[M(H,t)\frac{f(H,t)}{<f>}(1-d)\right] \quad (3)$$

where d represents these disruptions.

We can express this in terms of P, the proportional representation of H in the entire population. Also, disruption can be written in terms of the defining length, thus:

$$P(H,t+1) \geqslant P(H,t)\frac{f(H,t)}{<f>} + \left[1 - P_c\frac{\Delta(H)}{N-1}(1-P(H,t))\right] \quad (4)$$

In the case that both parents, rather than just one, is selected based on fitness then the schema theorem needs to be modified:

$$P(H,t+1) \geqslant P(H,t)\frac{f(H,t)}{<f>} + \left[1 - P_c\frac{\Delta(H)}{N-1}(1-P(H,t)\frac{f(H,t)}{<f>})\right] \quad (5)$$

For bit-flip mutation with probability P_m, the probability that a given schema is mutated will be dependent on the order of that schema:

$$P(H,t+1) \geqslant P(H,t)\frac{f(H,t)}{<f>} + \left[1 - P_c\frac{\Delta(H)}{N-1}(1-P(H,t)\frac{f(H,t)}{<f>})\right](1-P_m)^{order(H)} \quad (6)$$

1.5 Premature Convergence

After several generation, it is possible that selection will eliminate certain values from bits in the chromosome. This is termed *premature convergence* if the population has not yet discovered a satisfactory solution. The mutation operator reduces this occurrence, however, mutation is a primarily disruptive effect and therefore reduces the efficiency of the search.

Many researchers point out that the idea that mutation should be used at very low levels, if at all, is due to a deficiency of the schema theorem. The schema theorem ignores the hill-climbing mechanism of mutations. Try a GAUL simulation with mutation only and you will see a fairly robust search behaviour, contrary to the prediction from the schema theorem.

As the overall fitness of the population is improved, the variance will tend to be reduced. This can be a problem since the selective pressure will also be proportionally reduced. This is another problem of premature convergence. There are two common solutions to this problem. The first is to use a rank-based selection operator. The second is to scale the fitness. there are numerous techniques for performing this scaling, for example subtracting the fitness of the least fit solution from every fitness score and then dividing every score by the most fit solution would give scaled scores which always covers the range from zero to one.

1.6 Limitations of the Schema Theorem

Although the schema theorem may be used to make many potentially useful prediction, simple experiments can demonstrate its inadequacy for representing particular GAs. For example, consider a mutation operator which randomises more than one of the bits in a bitstring. The schema theorem would suggest that this would be less efficient than a operator which randomises a single bit. However, it is easy to derive simple problems for which the reverse is observed.

1.7 Genetic Alphabets

The original justification for using a binary alphabet in preference to alphabets with higher cardinality can be described in terms of schema processing. The binary alphabet maximises the number of hyperplane partitions available, while higher cardinality alphabet's sampling of each partition will be reduced. With binary alphabets, order-1 partitions will be sampled by 50% of the population, asumming a random sampling. Likewise, each order-i hyperplane is sampled by $\left(\frac{1}{2}\right)^i$ of the population. For a base n alphabet, the sampling ratio will be $\left(\frac{1}{n}\right)^i$, thus reducing the representation of hyperplanes by a given population size. Advantages of non-binary encoding include the greater range of available operators which allow domain-specific properties to be exploited. It has also been suggested that higher order alphabets implicitly do sample the equivalent binary hyperplanes.

1.8 Further Limitations of the Schema Theorem

It is clear that by ignoring schemata gains and underestimating schemata losses, much information is lost. This is indicated by expressing the equations as an inequality, so in essence information is only provided about the limiting case. Another shortcoming is the absence of consideration of the fact that the observed fitness of a hyperplane H will change from generation to generation as the population concentrates the samples in reduced regions of the hyperspace. This results in the schema theorem only being exact at the first generation. This is confirmed by the inability of the schema theorem to adequately predict computational behaviour.