tends to increase. According to the Darwinian theory, this process of **natural selection** is responsible for the development of all life forms on earth. Researchers hope to harness its power for computational purposes by implementing simulations of the process. In such simulations, the individuals are candidate solutions to some problem and fitness is a measure of solution quality. The aim is thus to 'evolve' high-quality solutions through simulated natural selection.

A common way of pursuing this approach involves use of the **crossover-based genetic algorithm** or **C-GA** [Goldberg, 1989]. In this approach, candidate solutions are represented as strings of characters or **genotypes**. Reproduction involves the production of a new individual through the splicing together of genotypes from two 'parents'. In the usual approach, parent genotypes are split at a certain point, forming a left part and a right part. The right part from one parent is then joined to left part from other, and vice versa. This produces two offspring genotypes which then replace relatively unfit individuals from the population.

2 Schema analysis

At first sight, the C-GA appears to be a way of randomly exploring the space of possible genotypes. However, Holland's **schema analysis** [Holland, 1975] provides an alternative picture. In this analysis we assume that the GA is a way of processing genotype *features* rather then genotypes themselves — a feature being simply a set of values in specific positions. A particular feature is defined in terms of a **schema**. This is a genotype-like string with specific values in some positions and 'don't care' values (asterisks) in others. An example is

```
*10**0****
```

This schema has ten characters in all, including seven 'don't care' values. It will match any 10-character genotype with a 1 in the second position, a 0 in the third position and a 0 in the sixth position. The genotypes which match the schema are referred to as its **instances**.

Note that we can construct a schema which will match some specific genotype by replacing any one of its characters with asterisks. There are thus 2^l schemas matching to any genotype of length l. There are potentially $n \cdot 2^l$ schemas for a population of n genotypes of length l, although in practice there will usually be fewer due to duplication. (There are always at least 2^l.) The **defining length** of a schema is the number of positions between its first and last specific position. Its **order** is the number of specified bits. Thus the defining length of '`*10**0****`' is 5 and its order is 3.

3 Schema growth under pure reproduction

The schema concept allows us to adopt a new perspective on the GA. Rather than thinking of it as a simulation of an evolutionary process we can think of it as carrying out 'schema processing'. We view every schema as having a fitness value which is defined as the average fitness of its instances. In Figure 1 we see five genotypes (column 1) each of which has a certain fitness value (column 2). The table also shows five example schemata (column 3), each of which has zero or more instances (column 4) among the listed genotypes. The mean fitness of each schema (column 5) is just the average of the fitnesses of its instances.

We view the goal of the GA as the multiplication of highly-fit schemas in the population.[1]

Genotype	Fitness	Schemata	Instances	Mean fitness
01100 (1)	103	*1*0**	1,2	63.5
01001 (2)	24	0**0**	1,2,4	83
10000 (3)				

4 Schema growth under reproduction with crossover

In an evolutionary process involving pure reproduction (i.e., copying), the population is always made up from copies of members of the original population. (In fact, if the process continues long enough we expect the population to be

greater than its 'vulnerability' — the ratio of its defining length to the genotype length. How easily is this 'shortness' requirement satisfied?

If fitness values range between 80 and 120 with the average value being 100, the maximum fitness ratio is 120/100 and the maximum fitness advantage is

fitness. Where GAs are used to evolve genotypes which encode for systems or mechanisms, the low-epistasis assumption will thus typically be violated.

The scenario in which fitness is affected by 'epistatic interactions' between parts of the genotype has, of course, been intensively investigated by the GA community. In fact, the construction of the so-called 'GA-deceptive' problem is typically a matter of deliberately nurturing epistasis ('nonlinearity') in an encoding [cf. Goldberg, 1989, ch. 2]. However, the observation that the schema theorem effectively assumes low epistasis, may help to explain the problems that some researchers have encountered in the use of C-GAs in **genetic programming** [Koza, 1992], cf. [Lang, 1995] and [O'Reilly, Forthcoming]. In this application, high-fitness genotypes are *programs* for a given task and the genotype is thus literally an encoding of a mechanism. Fitness is not independently attributable to individual parts of the genotype, but only to their interactions.

6 The building block hypothesis

The credibility of the C-GA does not rest solely on the schema theorem. It also rests on the so-called **building-block hypothesis**. This states that the crossover GA works well when short, low-order, highly fit schemas recombine to form even more highly fit, higher-order schemas. In fact, as Forrest and Mitchell [1996] note, 'the ability to produce fitter and fitter partial solutions by combining blocks is believed to be the primary source of the GA's search power.' Unfortunately, when we come to examine the assumptions introduced by the building-block hypothesis, we find that they contradict those introduced by the schema theorem.

The building-block hypothesis assumes that the fitness of any one block is typically affected by the other blocks on the genotype. If this were not the case it would be meaningless to talk about a 'building-block process' operating over and above the usual evolutionary process. Thus the building-block hypothesis implicitly assumes only a positive effect of epistasis on fitness and thus contradicts the low-epistasis assumption introduced by the schema theorem.'

When we come to consider the length implications of the building-block hypothesis we uncover a further contradiction. During the building-block process, the schemas that require processing at any given stage are actually the blocks that have been put together by the prior building-block process. Except at the initial stage, the defining length of these schemas is related to the defining lengths of the *components* of the blocks. Consider a block made up of just two schemas. One schema may be nested inside the other. In this case the defining length of the block is simply the defining length of the longer of the two schemas. At the other extreme the two schemas might be situated at opposite ends of the genotype. In this case the defining length of the new block will be close to the genotype length l.

If we make the conservative assumption that, on average, the defining lengths

As Forrest and Mitchell [1996] have commented there is a 'need for a deeper theory of how low-order building blocks are discovered and combined into higher-order schemas.'

7 Summary

As Forrest and Mitchell have noted, confidence in the efficacy of the GA is still largely based on the building-block hypothesis and the schema theorem. The schema theorem shows that schemas with high fitness are given exponentially increasing numbers of trials through reproduction but only if we assume that their fitness contributions are context-free and their defining lengths are sufficiently short. In reality, as we have seen, neither of these assumptions is easily satisfied.

When we come to consider the assumptions implicitly introduced by the building-block hypothesis we find that they implicitly contradict the assumptions underpinning the schema theorem. Thus, if we assume that the viability of the GA process is established by the schema theorem but that its 'power' is accounted for by the building-block hypothesis, we have to conclude that the GA

[3] Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley.

[4] Holland, J. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press.

[5] Koza, J. (1992). *Genetic Programming: on the Programming of Computers by Means of Natural Selection*