

# The Building Block Fallacy

*Chris Thornton*

Cognitive and Computing Sciences  
University of Sussex  
Brighton  
BN1 9QH  
UK

Email: Christopher.Thornton@firenet.uk.com

Tel: (44)1273 678856

May 21, 2003

## Abstract

Genetic Algorithms (GAs) are increasingly used for such purposes as deriving programs [1] and producing designs for robots [2]. According to the building-block hypothesis and schema analysis of Holland [3] the GA is an efficient search method. However, empirical work has shown that in some cases the method is outperformed by simpler processes such as random-permutation hill climbing [4] and [5]. The present paper reexamines Holland's framework (as formulated by Goldberg [6]) and finds that such in-practice failures are predictable given the implications of the schema analysis. The high efficiency of the GA method is commonly attributed to its 'implicit parallelism', i.e., its ability to develop candidate solutions in parallel, without focussing on any particular solution at any one time. However, this efficiency is hard to realise because there is a deep contradiction between the building-block hypothesis and the schema theorem.

## 1 Introduction: natural and simulated evolution

In natural evolution, populations of individuals compete to survive and reproduce. Relatively fit individuals survive longer and thus reproduce more. Over time, the fitter examples of random variations accumulate and average fitness 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** [6]. 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** [3] provides an alternative picture. In this analysis we assume that the GA is a way of processing genotype *features* rather than 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.<sup>1</sup>

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)	87	*****1	2,4	72
00011 (4)	122	11****	none	0
01010 (5)	90	**01**	4,5	106

Table 1: Relationship between genotypes, schemas and fitness.

As Holland has shown, this goal is achieved by ordinary reproduction (i.e., genotype-copying) without the need for crossover. Under this scenario, the evolutionary process involves repeatedly selecting an individual and making a copy of it. Provided that genotypes are selected with a probability proportional to their fitness, high-fitness schemas are bound to become more prevalent in the population.

Highly fit schemas will obviously tend to have many, highly fit instances. Each of these has a high chance of being selected for reproduction. Thus, highly fit schemas will tend to multiply. In fact under pure reproduction, schemas can be expected to grow ‘as the ratio of average fitness of the schema to the average fitness of the population’ [6]. Unfit schemas gradually get squeezed out of the population because their instances tend not to be reproduced.

The growth formula for schemas in a pure-reproduction scenario is as follows [6].

$$m(H, t + 1) \geq m(H, t) \cdot \frac{f(H)}{\bar{f}}$$

Here  $H$  is a schema,  $m(H, t)$  is the number of instances of  $H$  in the population at time  $t$ .  $f(H)$  is the fitness of  $H$  and  $\bar{f}$  is the average fitness.

## 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 made up entirely of copies of the fittest genotype from the original population.) If the original population does not contain a good solution then the method obviously cannot succeed in finding it. Some source of novelty is required. To

---

<sup>1</sup>It is of course the instances rather than the schemas which are ‘in’ the population.

address this requirement the reproduction regime is typically modified to use crossover rather than copying. This ensures that novel genotypes can be generated. Unfortunately, the crossover method involves cutting parental genotypes into two parts and thus runs the risk of slicing up and *destroying* useful schemas. We therefore need to determine the extent to which the destructive properties of the crossover operator undermine schema-processing.

In the schema analysis, this is dealt with by introducing a schema conservation probability to the growth formula. In the original growth formula, the multiplication of a schema in each round is determined solely by the ratio of its fitness advantage (the difference between fitness under consideration and mean fitness) to the mean fitness. The probability of schema destruction depends on the defining length of the schema: longer schemas have a higher probability of being destroyed under crossover. The probability of destruction is thus the ratio of defining length to genotype length.<sup>2</sup> and the probability of conservation is the complement of this. The amended formula thus becomes

$$m(H, t + 1) \geq m(H, t) \cdot \frac{f(H)}{\bar{f}} \left[ 1 - \frac{\delta(H)}{l} \right]$$

Here  $\delta(H)$  is the defining length of  $H$  and  $l$  is the total genotype length.<sup>3</sup>

The destructive properties of the crossover operation are assumed to be negligible with schemas of short defining length since, in this case, the bracketed part of the growth formula evaluates to a value close to 1. This leads directly to the so-called **schema theorem** which states that ‘short, low-order, above-average schemata receive exponentially increasing trials in subsequent generations.’ [6]

## 5 The shortness and low-epistasis assumptions

According to the schema theorem, schemas will only be correctly preserved if they are of ‘short’ defining length. In fact a schema will only be preserved correctly if its fitness advantage (the excess of its fitness over the average fitness) is 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 20/100. Any schema for which the ratio of defining length to genotype length exceeds this will not be preserved correctly. If the schema is of above average fitness, it will receive *decreasing* rather than increasing trials in future populations (and vice versa). With a genotype length of 100 bits, the defining length of a schema must thus be less than 20 bits. The number of schemas with a defining length of less than 20 bits form less than  $10^{-22}$  percent of the total number

<sup>2</sup>In Goldberg’s [6] book, it is the ratio of defining length to genotype length - 1.

<sup>3</sup>Accounting for the destructive effects of a mutation operation required a further change to the formula, see [6, p. 33].



## 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 [4] 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 of blocks will be at least the *sum* of the defining lengths of the components, then we see that real schema length will increase at least exponentially during the building-block process. Assume all original schemas have defining length 1 and that all blocks have two components. Then a first-level block has defining length 2. A second level block has defining length 4. A third level block has defining length 8, and so on.

The schemas that must be processed at any given stage are the blocks that have been created by previous processing. Thus, strictly speaking, the schema growth formula — which defines growth in terms of a *particular* schema — cannot be meaningfully applied to a C-GA in a building-block scenario. If it is to be applied, we should at least use a time-fixed schema identifier and a definition of defining length which explicitly captures the time-related growth. For example, we might let  $H_t$  denote a schema processed at time  $t$ , and define the schema length as

$$\delta(H_t) = x^t$$

with  $x$  being the defining length of an original schema.

Once, the implicit growth in schema lengths is made explicit, we see that the building-block hypothesis (which depends on negligible schema length) is very likely to be violated in all but the initial stage of processing. It demands increasing schema length and thus violates the shortness assumption introduced by the schema theorem.

Thus we find that both of the assumptions introduced by building-block hypothesis directly contradict assumptions introduced by the schema theorem. The situation is illustrated schematically in Figure 1. Given the importance of

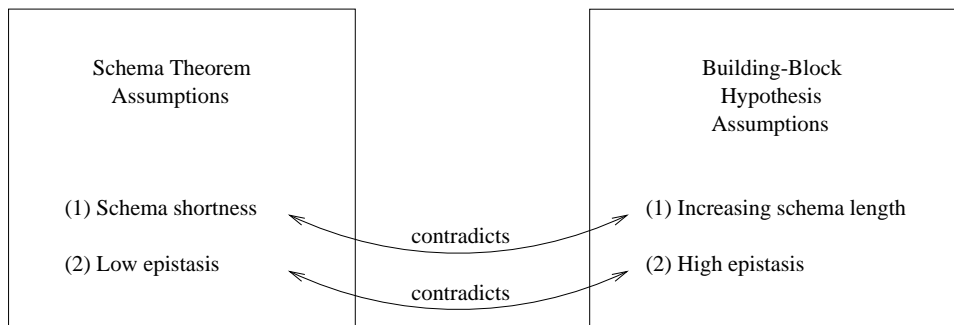


Figure 1: Inherent contradictions in the schema/building-block framework.

the building-block hypothesis within the GA paradigm this clash of assumptions occurring at the most fundamental level of the analysis is of particular interest. As Forrest and Mitchell [4] 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 only works well in the situation in which it is guaranteed not to work effectively. The practical failures of the method, such as those reported by For-

rest and Mitchell [4] for the standard C-GA, and by Lang [5] and O'Reilly [7] for the GP variant, might thus be viewed as a predictable consequence of the contradictory assumptions upon which the method is based.

## 8 Acknowledgements

I am indebted to Inman Harvey for helping me to understand the way in which the schema theorem introduces the assumption of low-epistasis. I am also indebted to the anonymous reviewers for a number of helpful suggestions.

## References

- [1] Koza, J. (1992). *Genetic Programming: on the Programming of Computers by Means of Natural Selection*. Cambridge, Massachusetts: MIT Press.
- [2] Cliff, D., Harvey, I. and Husbands, P. (1993). General visual robot controller networks via artificial evolution. In D. Casement (Ed.), *Proceedings of the Society of Photo-optical Instrumentation Engineers Conference (SPIE-93), Session on Intelligent Robots and Computer Vision XII: Algorithms and Techniques*. Boston MA.
- [3] Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press.
- [4] Forrest, S. and Mitchell, M. (1996). Relative building-block fitness and the building-block hypothesis. In D. Whitley (Ed.), *Foundations of Genetic Algorithms 2*. San Mateo, CA: Morgan Kaufmann.
- [5] Lang, K. (1995). Hill climbing beats genetic search on a boolean circuit synthesis task of koza's. *Proceedings of The Twelfth International Conference on Machine Learning* (pp. 340-343).
- [6] Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.
- [7] O'Reilly, U. (Forthcoming). An analysis of genetic programming. PhD Thesis.