Schemata Tutorial

Rajneesh Pawar1, Dr. J.S. Saini2

[1]Assistant Professor, Deptt. of Electrical Engg., D.C.R. Univ. of Science & Tech.,
Murthal, Sonipat, Haryana (INDIA) – 131039; Email: rajneeshdpawar@gmail.com

[2]Professor, Deptt. of Electrical Engg., D.C.R. Univ. of Science & Tech.,
Murthal, Sonipat, Haryana (INDIA) – 131039; Email: jssain@rediffmail.com

Abstract

Genetic Algorithms are a powerful tool for solving optimization problems. This paper attempts to survey and analyze the existing theoretical work of schemata with the intent of paving a way for further theoretical advances like the unique schemata count, its relation with population size and its effect on convergence, etc.

***Key Words**: Genetic Algorithm (GA), schema, Similarity bits, strings, similarity bits count, schemata count, length of string, critiques, convergence, population size, fitness functions.*

1. INTRODUCTION

For the last 30 years, the growth of interest in heuristic search methods for optimization has been quite popular, fruitful and dramatic. One of these most interesting developments is in the application of genetic algorithms (GAs). In most GA's, the individuals are represented by fixed-length binary strings. The notion of Schema was introduced in the mid 70's so as to understand how genetic algorithms could so well direct the search towards high fitness regions of the search space.

This paper introduces the brief concepts of genetic algorithms, early development of schema theory, critiques, revival of schema theory, etc. The notion that we should look for is to try and exploit structures for Holland's schemata and schemata count. Their different theoretical perspectives have been presented to understand GAs better.

A substantial body of literature has grown out of this area of interest and the goal of this paper is to present schemata in such a way that the new researchers can grasp the basic concepts and utilize these for furtherance of advancements in the field of schemata and its usage. The paper attempts to cover the growth of this discipline with a focus towards analysis and synthesis of unique schemata count, its relation with similarity bits, etc. and points out a prominent direction of using its effect on convergence and determination of optimal population size.

To this end, the paper is organized into three broad sections, namely the brief description of Genetic Algorithms & theoretical concepts of schema theory in Section-2, critiques and resurgence of Schema Theory in Section-3 and the analysis & synthesis of schemata count in Section-4 and, of course, a concluding Section-5 with future directions pointed therein.

2. A BRIEF DESCRIPTION OF GA AND SCHEMATA

A Genetic Algorithm (GA) is a robust optimization technique based on natural selection (like reproduction or selection) and mixing or recombination (like crossover and mutation) operators. The basic goal of GAs is to optimize functions known as fitness functions or objective functions.

Its basis lies in natural selection which allows a GA to employ Darwin's theory of "survival of the fittest" when searching for optima. The use of a population of points helps the GA avoid converging to false peaks (local optima) in the search space.

The following subsections describe basics of GAs and schemata in brief. Most of the information presented here is based on [15, 19, 24-25]

2.1 GA Basics

A simple GA starts with a population of solutions (also called individuals or strings) encoded in one of the many ways. Binary encodings are quite common and are used in this paper. The GA determines each individual's strength based on an objective function (also called Performance Index or Fitness Function in which case the goal will be to maximize it, however, the goal for an objective function could be either minimization or maximization). The GA then performs one or more of the following three genetic operators on certain individuals in the population [7].

1. Reproduction (also called selection) is simply an orientation towards retaining fitter strings in the following generations, but coupled with a variety of low-fitness solutions too.
2. Crossover involves swapping partial strings of random length between two parent strings.
3. Mutation involves flipping a random bit in a string of binary encoding.

The crossover and mutation put together are called variation operators as they impart variety to the emerging populations; however the role of selection is reinforcement but with a slight inclination towards mediocre or poor fitness strings so as to explore their hidden potential, if any.

These three operations primarily involve random number generation, copying, and partial string exchange and random alterations [27]. Thus, GA's are simple to implement.

In each generation (GA iteration), the GA, through selection operator, probabilistically chooses strings based on their strengths, i.e., stronger strings are selected more frequently. After selection, the two chosen parents are probabilistically crossed, a process also called mating. Then, rarely one or more bits are changed; this mutation helps preserve the population's diversity, preventing the reaching of false peaks, or local optima in the solution space, which is a problem commonly encountered by standard hill climbing methods [16].

2.2 Theoretical Aspects of Schemata

This subsection explains the theoretical aspects of schemata as to why genetic algorithms are so successful in complex optimization problems in terms of the schemata and the effect of genetic operators on them.

Holland [18-19] invented the idea of schema (schemas or schemata for plural) to formally conceptualize the notion of 'building blocks'. The theoretical foundations of canonical genetic algorithms rely on using a fixed-length binary string representation of individuals (solutions), and on the notion of schemas, the symbol alphabet {0, 1, *} is considered where '*' is a special wild card or don't care symbol that matches both 0 and 1. A schema represents a hyper-plane in the search space with all strings which match it in all positions except in the positions with the don't care or wild card. For example, the schema [0*10*] is a template that matches the following four strings: [00100], [00101], [01100], and [01101].

If we interpret the set of binary strings of length n as hyper-cubes of dimension n, then schemata can be interpreted as hyper-planes, lines and corners in these hyper-cubes (see Figure 2 for an example with n = 3).

2.2.1 SETS AND SUBSETS

For a binary string of length-m, there are 2m possible strings and 2x; (with x = 2m) possible substrings (because of the wildcard symbol). Therefore, for a binary string with length 4-bits, there are 16 possible binary strings and an amazing 65536 (m = 4 so, x =16 and therefore, 216 = 65536) substrings, yet only 3m, i.e., 81 unique schemata to represent them.

Any binary string is a member of 2m different schemata. The string '01' with length-2 bit is an instance of **, 0*, *1, and 01. Therefore, if all strings are identical, then there are 2m schemata and if all strings are totally different, then the schemata lie between 2m & n. 2m, depending upon the diversity. For clarity about 3m and n. 2m schemata, one may refer to [38].

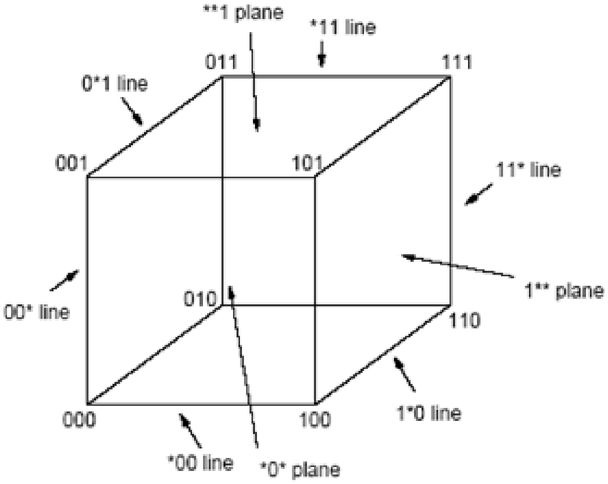


Figure 2: A hyper-plane interpretation of schemata for n = 3.

3. HISTORICAL DEVELOPMENTS, CRITICISMS & RESURGENCE OF SCHEMA THEORY

This section dwells up the developments in the field of schemata in the context of GA's. In the backdrop of the extensive criticisms by several researchers, this paper then looks for resurgence of the field by focusing on the new developments that seem to end the stalemate.

3.1 Historical Developments

The genetic algorithm refers to a model introduced and investigated by Holland [19] and his research students, namely, De Jong [8]. Most of the schema theories subsequently reported and related to traditional genetic algorithms solely depended on this model of Holland.

The search heuristics of GA are based upon Holland's schema theorem. Researchers working in the field of GAs have put in a lot of effort into the analysis of the genetic operators (crossover, mutation, selection). In order to achieve better analysis and understanding of how GA successfully searches for fitter solutions, Holland has introduced a construct called schema [19] as described in the above section.

A simple way to view the Genetic Algorithm is provided by the schema theorem known as the "The Fundamental Theorem of Genetic Algorithms" put forth in 1975 by John Holland. He first introduced the concept of "implicit parallelism", which explains the processes of genetic algorithms. Implicit or intrinsic parallelism is the fundamental foundation for genetic algorithms which is analyzed as: For a population size of 'n' binary strings, we actually obtain much larger number of useful processing of n3 schemata in each generation.

Holland then further analyzed the influence of selection, crossover, and mutation on the expected number of schemata, when going from one generation to the next and gave the Schema Theorem [19] to describe quantitatively the change of expected number of schema instances over time. A detailed discussion of related analysis can be found in Goldberg [15]; in the context of the present work, however, we only outline the main results and their significance as below:

The dynamics of Schema Theorem can be mathematically characterized by an expression given as:

$$E\{v(H,t+1)\} \geq [v(H,t).ff–(H,t)/f–(t)].[1–pc\{\delta(H)/(m–1)\}].[(1–pm) O(H)]; \quad (1)$$

where:

E{v(H, t+1)} is the expected value of number of instances of schema, H, at time t+1;

v(H, t)} is the number of instances of H at the time t;

f– (H, t) is the average fitness of H at time t (i.e., average fitness of instances of H in the population at time t);

f– (t) is the average fitness of the population at time t;

m is the string length.

Probability of crossover or crossover rate pc: probability that two parents will crossover in a single point.

Probability of mutation pm: probability that mutation can occur at each bit position in a string.

Schema order O(H): of a schema H is the no. of fixed positions (in binary alphabet case, no. of 1's & 0's) in the template, i.e., the no. of defined bits (non-asterisks, non-don't care positions) in schema, e.g., H1 = (011*1**), so O(H1) = 4, H2 = (0******), so O(H2) = 1. Order defines specificity of a schema. In this example, H1 is more specific than H2. The schema order is useful in computing survival probabilities of schema for mutations.

Defining length (H): of a schema, H, is the distance between the first and the last specific string positions, i.e., distance between its outermost defined bits, e.g., (011*1**) = 5 1 = 4; and (0******) = 1 1 = 0. It defines compactness of information contained in a schema, and is useful in calculating survival probabilities of schema for crossovers.

Basically, schemata provide basic means for analyzing the net effect of reproduction & genetic operators on building blocks in a population. The credibility of the GA does not rest solely on the schema theorem. It also rests on the so-called Building-Block Hypothesis (BBH). The building block hypothesis is stated in Goldberg's book [15] as: "A genetic algorithm achieves high performance through the juxtaposition of short, low order, highly fit schemata, or building blocks". The meaning of "highly fit schemata" is not completely clear. The most obvious interpretation is that a schema is highly fit if its average fitness is considerably higher than the average fitness of all strings in the search space. In fact, the 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" given by Forrest and Mitchell [14] justify the statement given by Goldberg in [15] towards BBH .

From the above discussion, it is clear that the schema theory is a very important result in Genetic Algorithms. However, it is obtained under idealized conditions and found reliable in many cases and does not hold good for most practical GA applications since the individual representation and the genetic operators used in practical applications may often be different from those used by Holland. Therefore, it is justified to criticize the problems for which it does not hold true. The critiques of schema theory by various researchers from time to time are given in the next section.

3.2 Criticisms and Resurgence

Schema theorem has been criticized by researchers from various directions. This section gives the examples of a wider criticism in literature.

One of the criticisms is that schema theorem only gives lower bounds on the expected value of the number of individuals sampling a given schema at the next generation. Therefore, they cannot be used to make predictions over multiple generations. For these reasons, many researchers like Vose [44] during the mid 90's believed that schema theorems were nothing more than trivial tautologies of no use whatsoever. Mühlenbein [23] has also argued about the behavior of schema theorem as: "…the schema theorem is almost a tautology, only describing proportionate selection…".

Radcliffe [33-34] argues that for many practical problems, conventional linear chromosomes and recombination operators are inadequate for effective genetic search. His critique of intrinsic parallelism is especially noteworthy. Bertoni and Dorigo [3] argue that intrinsic parallelism is of limited use and point out that O(N3) lower bound relies upon a particular relationship between N – population size and p — the maximum schema order considered in the derivation with the assumption being that N ~ 2p. By considering the case N ~ 2βp for various ranges of β values, they extend the argument to deal with an arbitrary population size and show that the relationship can be anything at all.

Liepins and Vose [21] discuss various representational issues in regards to genetic optimization. Their paper elaborates the failure modes of a GA and eventually represents the existence of an affine transformation that would convert a deceptive objective function to an easily optimizable objective function.

Vose [47] examines the function of mutation in a GA, and shows that a tiny change in the value of the mutation rate, μ, can cause a profound change in the GA's trajectory — a change which no schema analysis can possibly predict.

Vose [49] further points out that that the number of instances of a schema that survive is misplaced, thus one can't predict about which instances of a schema appear in the next population. The subset of strings represented by a schema usually has a good amount of fitness variability, so the new set of strings in the next population is not necessarily as fit as the previous population.

Macready and Wolpert [22], also point out a flaw in Holland's mathematics, "that there is no reason to believe that Holland's strategy is an optimal one. It follows that, even if we accept that the GA adopts 'exponentially increasing trials', it is, in any case, not the best way of solving a competition between two competing schemata”.

Antonisse [2] points out that what is meant by a 'schema' for higher cardinality alphabets is not necessarily the same as it is in the case of binary strings where each '*' can only be replaced either by '0' or '1'. With non-binary encodings, he argues, we should consider all subsets that replace a '*', counting each of them as a possible schema. This perspective, leads to many more schemata available for higher-cardinality alphabets.

In addition to above, some other researchers also criticize the schema theorem in one or the other aspect. Altenberg [1] argues that the schema theorem has very little to say about the recombination operators. Fogel and Ghozeil [12-13] criticize it for not being able to estimate correctly the expected proportion of a schema in the population when fitness proportionate selection is used in the presence of stochastic effects and schema fitness varies according to the specific random variables involved. Chung and Perez [6] describe how the schema theorem affects the behavior of a GA and uncover the relationship between building blocks and increasing schemata and thus consider that the schema theorem is insufficient and it needs the interpretation carefully.

Thus, the schema theory does not deliver the explanation for GA behavior that it originally appeared to promise.

While some criticisms mentioned in the above section are really not justified as discussed in [30, 35-36], others are reasonable and apply to many schema theories.

However, this does not mean that the situation cannot be changed and, therefore, the acknowledged criticisms and inadequacies of the Schema Theorem stimulate new lines of enquiry.

This section shows that schemata exist and can be further researched to advance the state-of-the-art.

It should be appreciated that Holland's original account of the Schema Theorem was certainly rather more refined than that provided by some of the later researchers for GAs. Nevertheless, the implications of the above points have often been neglected by later researchers, so one may seem somewhat dissatisfied to find that over the course of the 1990s, Holland's GA theory has been so comprehensively discarded, for example, Vose states “... schema theory tells us almost nothing about GA behaviors…” [50], a judgment which clearly dismays the work of John Holland [19-20].

However, the appropriate way science and technology develops is as shown by works [5, 29-32, 41-43] that consider that the schema theories have not been fully exploited, neither fully analyzed nor fully developed.

De Jong, who did a great deal of work towards optimization and was credited with initializing the optimizing aspects of GAs, cautions in a few of his papers [9-10] in which he contends that the Holland's original vision of a GA are not really function optimizers – preferring the acronym 'GAFO' for the application that has become very popular and widely used [11].

By the end of the last millennium, Stephens and Waelbroeck [42] developed an exact GA schema theory which gives an exact formulation (rather than a lower bound) for the expected number of instances of a schema at the next generation in terms of macroscopic quantities.

One is to take account of both 'gains' and 'losses' in the process of recombination, so that the Schema Theorem can be written as an equation, instead of a lower bound. The version [5] is an attempt to deal with this particular problem, although such ideas were originated way back in 1987 [4].

Heckendorn and Whitley [17] do extensive analysis of epistasis using the Walsh transforms. Poli gives his idea of recursively partitioning schemata [32] in order to derive a probability distribution for the number of instances of a schema that survive.

Perhaps, the most satisfying and elegant development stemming from dissatisfaction with schema-processing arguments is the formal executable dynamical system model developed by Vose [48] and Whitley [52] that takes into account that the construction of strings (or schemata) as well as the destruction needs to consider all possible ways of performing crossover and mutation.

Neither is it to say that schemata must be completely thrown to the dustbin while we search around for newer and better ways of describing the internal processes of the genetic algorithm. The suggestion that schema theorem tells us 'almost nothing' might be considered excessive and inordinate, as the schema theory is somehow still useful to us [37]. It has been seen that with revival of interest in schema theorems [1, 5, 29-32, 41-43, 46, 48-51] and with the contributions of new researchers in the field of Genetic algorithms, the concept of a schema is rather more limited than it at first appeared, which means it has although been intensively studied, yet further contributions can still be made.

The notion that we should look for is to try and exploit structures for Holland's schemata, although the same may not necessarily fulfill the complete objective. This is further clarified by the work of Saini [38-39] and Pawar [26] in putting the schemata in a wider and newer context. They show how the exploitable similarities (for which a new term 'similarity bits' was coined in [38]) could be related to the schemata, thus opening up new dimensions of research on schemata count and its usage in practical / theoretical applications.

4. ANALYSIS OF SCHEMATA COUNT

Towards analyzing the interrelations between different similarity metrics, a few terms are first defined and in the sequel thereto, the illustrations and mathematical expression evolved are put forth.

4.1 Definitions in the Context of Schemata Count

This section comprises of a few definitions needed in the context of schemata count. Saini & Pawar [26, 40] coin and define the terms needed for the analysis and synthesis of schemata count as:

SIMILARITY BITS COUNT

The number of matching alleles between any two individual strings is called the similarity bits count. It is denoted by csb. For example, consider a population of 4 individual strings with length three as shown in Figure 3. From this Figure 3, the strings taken are: S1: (0 1 0) and S2: (0 1 1); the similarity bits count between the strings S1 & S2 is csb = 2.

TOTAL SIMILARITY BITS COUNT

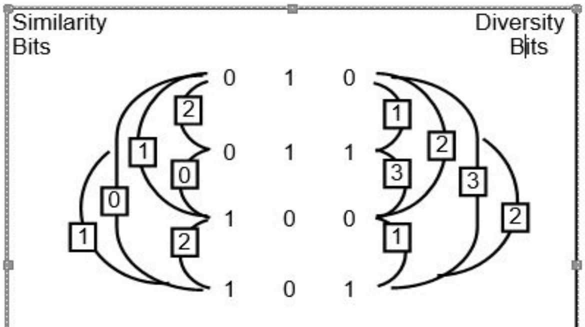
The total of all such similarity bits counts in a population of 'n' strings constitutes the total similarity bits count. It is denoted by ctsb. In the same example as shown in Figure 3, the total similarity bits count, ctsb = 2 + 0 + 2 + 1 + 0 + 1 = 6.

DIVERSITY BITS COUNT

The number of non-matching alleles between any two individual strings is called the diversity bits count. It is denoted by cdb. For example, for the case in Figure 3, the diversity bits count between the strings S1 & S2, cdb =1.

TOTAL DIVERSITY BITS COUNT

The total of all the diversity bits counts in a population of 'n' strings constitutes the total diversity bits count. It is denoted by ctdb. For example, for the case in Figure 3, the total diversity bits count, ctdb = 12.



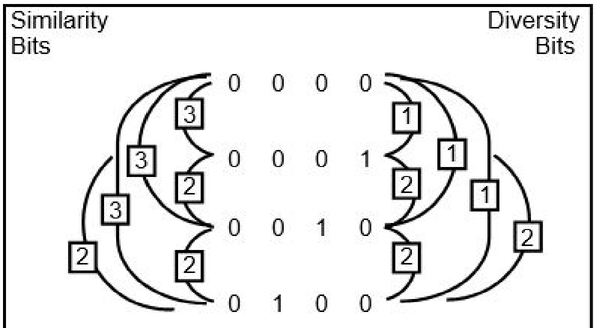
Total similarity bits count, C_{tsb} =2+0+2+1+0+1 = 6.
Total diversity bits count, C_{tdb} = 1+3+1+2+3+2=12.

Figure 3: Four strings each of length 3 that depicts breakup of similarity and diversity bits count.

4.2 Illustrations Showing Correlation between Schemata and Similarity Bits

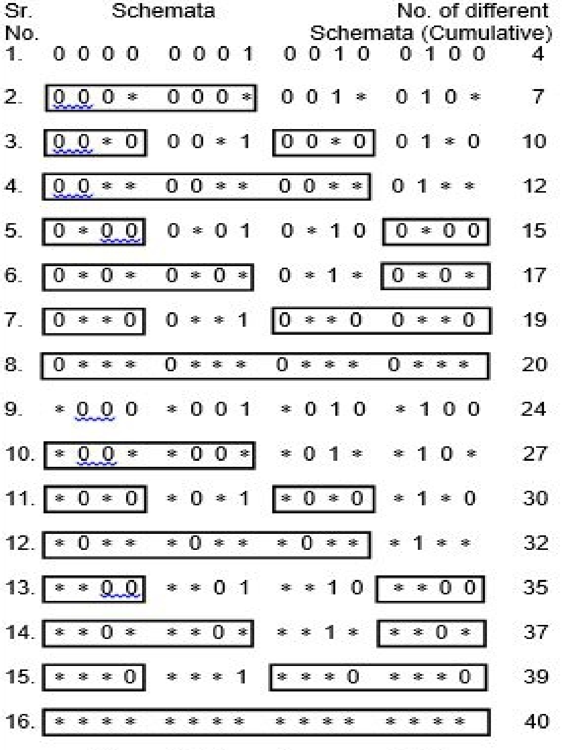
Let us take a population of n-strings, each of length 'm', the number of schemata lies between 2m & n.

2m depending upon population diversity. In [38-39], it was pointed out that the schemata count is correlated with similarity bits.



Similarity bits count, C_{tsb} = 3+2+2+3+3+2= 15.
Diversity bits count, C_{tdb} = 1+2+2+1+1+2 = 9.

Figure 4: Four strings with length four that depicts similarity and diversity bits count.



Sr. No.	Schemata				No. of different Schemata (Cumulative)
1.	0 0 0 0	0 0 0 1	0 0 1 0	0 1 0 0	4
2.	0 0 0 *	0 0 0 *	0 0 1 *	0 1 0 *	7
3.	0 0 * 0	0 0 * 1	0 0 * 0	0 1 * 0	10
4.	0 0 * *	0 0 * *	0 0 * *	0 1 * *	12
5.	0 * 0 0	0 * 0 1	0 * 1 0	0 * 0 0	15
6.	0 * 0 *	0 * 0 *	0 * 1 *	0 * 0 *	17
7.	0 * * 0	0 * * 1	0 * * 0	0 * * 0	19
8.	0 * * *	0 * * *	0 * * *	0 * * *	20
9.	* 0 0 0	* 0 0 1	* 0 1 0	* 1 0 0	24
10.	* 0 0 *	* 0 0 *	* 0 1 *	* 1 0 *	27
11.	* 0 * 0	* 0 * 1	* 0 * 0	* 1 * 0	30
12.	* 0 * *	* 0 * *	* 0 * *	* 1 * *	32
13.	* * 0 0	* * 0 1	* * 1 0	* * 0 0	35
14.	* * 0 *	* * 0 *	* * 1 *	* * 0 *	37
15.	* * * 0	* * * 1	* * * 0	* * * 0	39
16.	* * * *	* * * *	* * * *	* * * *	40

Figure 5: Schemata represented by population of Figure 4

The more diverse a population, the closer will be the number of unique schemata to the number n. 2m. An illustration from is taken that shows a population of 4 strings of length four each, having the total similarity bits count as 15 (or, for that matter, total diversity bits count as 9) as shown in Figure 4.

One can now see in Figure 5 how many unique schemata are represented by the population of Figure 4. The common schemata are enclosed in rectangles on the same line. The count of unique (different) schemata in Figure 5 comes out to be 40.

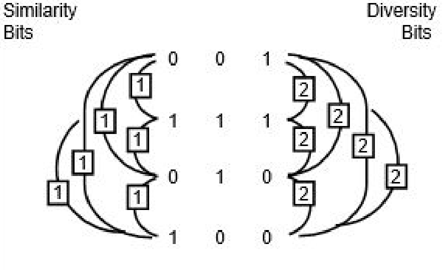
All this goes to show that there is a definitive correlation between the number of total diversity bits in a population and the number of different schemata represented by it. It was therefore, pointed out in [38-39] that total diversity bits could be used as an alternative to the schemata count for the GAs.

4.3 Illustrations Showing computation of Unique Schemata Count

In [38-39], it is pointed out that the number of unique schemata count in a population depends upon the total similarity bits count (or, for that matter, total diversity bits count). However, on analysis and investigation by taking two or more different populations having same total similarity bits count but different breakup of similarity bits count, it is found that unique schemata count is correlated with the breakup of similarity bits count as shown in the following illustration [40].

Illustration:

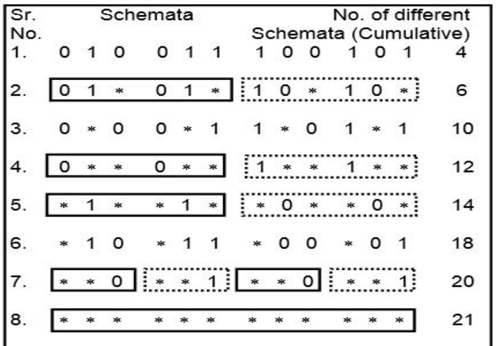
Figures 3 & 6 show two different populations of 4 strings of length three each, having the same total similarity bits count (or, for that matter, total diversity bits count) but different breakup of similarity bits count.



Total similarity bits count, C_{tsb} =1+1+1+1+1+1 = 6.
Total diversity bits count, C_{tdb} = 2+2+2+2+2+2=12.

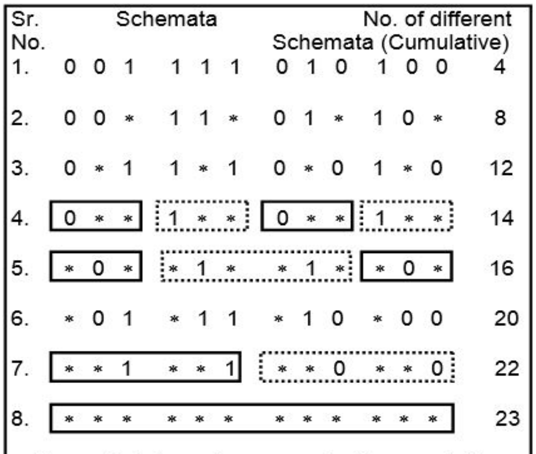
Figure 6: Four strings each of length three with different breakup of similarity bits count but same C_{tsb} = 6 as in Figure 3

number of unique schemata in a population does not depend upon the total similarity bits count (or, for that matter, total diversity bits count) but instead depends upon the breakup of similarity bits count in the population.



Sr. No.	Schemata				No. of different Schemata (Cumulative)
1.	0 1 0	0 1 1	1 0 0	1 0 1	4
2.	0 1 *	0 1 *	1 0 *	1 0 *	6
3.	0 * 0	0 * 1	1 * 0	1 * 1	10
4.	0 * *	0 * *	1 * *	1 * *	12
5.	* 1 *	* 1 *	* 0 *	* 0 *	14
6.	* 1 0	* 1 1	* 0 0	* 0 1	18
7.	* * 0	* * 1	* * 0	* * 1	20
8.	* * *	* * *	* * *	* * *	21

Figure 7: Schemata represented by population of Figure 3



Sr. No.	Schemata				No. of different Schemata (Cumulative)
1.	0 0 1	1 1 1	0 1 0	1 0 0	4
2.	0 0 *	1 1 *	0 1 *	1 0 *	8
3.	0 * 1	1 * 1	0 * 0	1 * 0	12
4.	0 * *	1 * *	0 * *	1 * *	14
5.	* 0 *	* 1 *	* 1 *	* 0 *	16
6.	* 0 1	* 1 1	* 1 0	* 0 0	20
7.	* * 1	* * 1	* * 0	* * 0	22
8.	* * *	* * *	* * *	* * *	23

Figure 8: Schemata represented by population of Figure 7

Saini & Pawar [26, 40] come a step forward in this direction by using this analysis of correlation between schemata and breakup of similarity bits count to develop an empirical expression for computation of unique schemata count. However, initially, the empirical relation so developed is applicable for a restricted case of maximum similarity bits count being just one less than the length, m, of the string (i.e., csb = m–1) which inter-alia means that the strings are non-repeated and further that the count of such maximum similarity bit counts for a population of non-repeated strings is (n-1), where n is the number of binary strings in the population.

Empirical relation for Unique Schemata Count, USC = n.2m – (n–1)2m–1 (2)

Applying this developed relationship for the population of Figure 4, we find the number of unique schemata in a population of four strings, each of length 4, by applying the above empirical relation as:

USC = 4.24 – 3.23 = 64 – 24 = 40,

which is corroborated by Figure 5.

Thus, as also partly observed in paper [40], while the tedium of computing the unique schemata in a population is obvious from Figures 5, 7 and 8, on the other hand, the developed empirical relation gives the same information via a single line computation of Eqn. (2).

4.4 Mathematical Expressions for Schemata Count

The above relation, however, represents only a restricted case of only one type. This section extends this specific case to a more generalized one for obtaining the unique schemata count for non-repeated strings of different populations, but without imposing the restriction as placed in the section 4.3.

Some spade work has been done towards the generalized mathematical expressions of Schemata Count by Pawar and Saini [28] that directly computes the Schemata Count. Pawar & Saini [28] developed the part of the generalized expression taking into consideration the cases with no missing similarity bits for directly computing the number of unique schemata.

Based upon different similarity metrics (such as breakup of similarity bits and total of similarity bits count in the population) and the population size, we develop the following generalized mathematical expression with no missing similarity bits (with same and different lengths) for directly computing the number of unique schemata as given below:

$$\text{Unique schemata count} = n.2^m - \{(s_{b(m-1)})2^{m-1} + (s_{b(m-2)})2^{m-2} + \ldots + (s_{b1})2^1 + (s_{b0})2^0\} + \{(s_{b(m-1)} - 1)2^{m-2} + (s_{b(m-2)} - 1)2^{m-3} + \ldots + (s_{b2} - 1)2^1 + (s_{b1} - 1)2^0\} \quad (3)$$

where n is the number of strings,

m is the length of the string,

sb(m-1) is the total of maximum similarity bits count, (e.g., if m = 4, then sb(m-1) is the total of 3-similarity-bits in the population).

sb(m-2) is the count (total) of just-one-less-than maximum-similarity-bits count,…

sb1 is the total of second-least-similarity-bits-count, (i.e., total of 1-similarity-bits- count)), and

sb0 is the total of least-similarity-bits-count, (i.e., total of 0-similarity-bits-count)

The above generalized mathematical expression is the first step towards the generalized mathematical expressions for computation of unique schemata count, i.e., cases with no missing similarity bits of non-repeated, strings, however, again it represents a few cases (approx. 10-15 cases in a population), although there exist a large number of cases for computation of schemata count; the more general case that is applicable to all the cases is under exploration. It is also evident from these caveats that there is a definitive basis which can further be evolved to examine how the schemata count is correlated to population size and convergence.

5. CONCLUSION & FUTURE DIRECTIONS

In this paper, the traditional schema processing arguments for GAs, as developed initially by Holland, have been described. We have also considered some of the extensions in case of schemata that have been derived from this basic framework.

While the idea of a schema has some value in explaining the GA performance, we are far from convinced that it holds the key to a complete understanding about all aspects of GA. There are many other aspects of traditional genetic algorithms like the generalized expression for unique schemata count, its relation with population size and its effect on convergence, etc. that can be exploited in future based on

empirical, intuitive and analytical explorations. We hope that this paper will increase interest in researchers towards schemata analysis as well as genetic algorithmic theoretical and practical applications developments.

REFERENCES

[1] Altenberg, L. (1995). The schema theorem and Price's theorem. In D.Whitley and M.Vose (Eds.) Foundations of Genetic Algorithms 3, Morgan Kaufmann, San Mateo, CA, 23-49.

[2] Antonisse, J. (1989). A new interpretation of schema notation that overturns the binary encoding constraint". In J.D. Schaffer (Ed.), Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 86-91.

[3] Bertoni, A., and Dorigo, M. (1993). Implicit parallelism in genetic algorithms. Artificial Intelligence, 61, 307-314.

[4] Bridges, C. L., and Goldberg, D. E. (1987). An analysis of reproduction and crossover in a binary-coded genetic algorithm. In J. J. Grefenstette (Ed.) Proceedings of the 2nd International Conference on Genetic Algorithms. Lawrence Erlbaum Associates, Hillsdale, NJ, 9-13.

[5] Chen, S., and Smith, S. (1999). Putting the "genetics" back into genetic algorithms. In W. Banzhaf and C.R. Reeves (Eds.) Foundations of Genetic Algorithms 5, Morgan Kaufmann, San Francisco, CA, 103-116.

[6] Chung, S. W., and Perez, R.A. (1994). The schema theorem considered insufficient. In Proceedings of the Sixth IEEE International Conference on Tools with Artificial Intelligence, pages 748-751, New Orleans.

[7] Davis, L. (1989). Handbook of Genetic Algorithms. New York: Van Nostrand Reinhold.

[8] De Jong, K. A. (1975). An Analysis of the Behaviour of a Class of Genetic Adaptive Systems. Ph.D. thesis, University of Michigan.

[9] De Jong, K. A. (1992). Are genetic algorithms function optimizers? In

R.Männer and B.Manderick (Eds.) Parallel Problem solving from Nature, 2, Elsevier Science Publishers, Amsterdam, 3-13.

[10] De Jong, K. A. (1993). Genetic algorithms are NOT function optimizers. In D.Whitley (Ed.) Foundations of Genetic Algorithms 2, Morgan Kaufmann, San Mateo, CA, 5-18.

[11] De Jong, K. A., Spears, W. M., and Gordon, D.F. (1995). Using Markov chains to analyze GAFOs. In L.D. Whitley and M.D. Vose, editors, Foundations of Genetic Algorithms 3, pp. 115-137. Morgan Kaufmann, San Francisco,CA.

[12] Fogel, D.B., and Ghozeil, A. (1997). Schema processing under proportional selection in the presence of random effects. IEEE Transactions on Evolutionary Computation, 1(4):290-293.

[13] Fogel, D.B., and Ghozeil, A. (1998). The schema theorem and the misallocation of trials in the presence of stochastic effects. In V.W. Porto, N. Saravanan, D. Waagen, and A.E. Eiben, editors, Evolutionary Programming VII: Proc. of the 7th Ann. Conf. on Evolutionary Programming, pp. 313-321, Berlin, Springer.

[14] Forrest, S., and Mitchell, M. (1993). What makes a problem hard for a genetic algorithm? Some anomalous results and their explanation. Machine Learning, 13, 285-319.

[15] Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, MA.

[16] Haupt, R. L., and Haupt, S. E. (2004). Practical Genetic Algorithms. USA: Wiley-Interscience, A John Wiley and sons inc.

[17] Heckendorn, R.B., and Whitley, D. (1998). Predicting epistasis directly from mathematical models. Technical Report, Dept of Computer Science, Colorado State University, USA.

[18] Holland, J. H. (1962). Outline for a logical theory of adaptive systems. J. Assoc. Comput. Mach., vol. 3. pp.297-314.

[19] Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. Cambridge, MA: First Edition, The MIT Press.

[20] Holland, J. H. (2000). Building blocks, cohort genetic algorithms, and hyperplane defined functions. Evolutionary Computation, 8, 373-391.

[21] Liepins, G. E., & Vose, M. D. (1990). Representational issues in genetic optimization. J. Expt. Theor. Artif. Intell., 2:101-115.

[22] Macready, W.G., and Wolpert, D.H. (1998). Bandit problems and the exploration/exploitation tradeoff. IEEE Transactions on Evolutionary Computation, 2, 2-13.

[23] Mühlenbein, H. (1991). Evolution in time and space - the parallel genetic algorithm. In G.J.E. Rawlins (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 316-33.

[24] Michalewicz, Z. (1992). Genetic Algorithms+ Data Structures =Evolution Programs. Berlin: Springer-Verlag.

[25] Mitchell, M. (1998). An Introduction to Genetic Algorithms. New Delhi: PHI Pvt. Ltd., 1998.

[26] Pawar, R. (2006) Recent advances in genetic algorithms, M.E. dissertation, MDU, Rohtak, Haryana, India.

[27] Pawar, R., and Sharma, A. (2007). Designing of PID Controller using genetic algorithms. Third National Conf. on Computing, INDICOM, New Delhi.

[28] Pawar, R., Saini, J. S., Gopal, M., and Mittal, A. P. (2011). Towards generalized expression for schemata count. Applied Soft Computing, Elsevier, 11, 2134-2141.

[29] Poli, R. (1999). Schema theorems without expectations. In W.Banzhaf, J.Daida, A.E.Eiben, M.H.Garzon, V.Hanavar, M.Jakiela and R.E.Smith (Eds.) Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann, San Francisco, CA.

[30] Poli, R. (2000a). Why the schema theorem is correct also in the presence of stochastic effects. In Proceedings of the Congress on Evolutionary Computation, pages 487-492, San Diego, USA.

[31] Poli, R. (2000b). Exact schema theorem and effective fitness for GP with one-point crossover. In D. Whitley, D. Goldberg, E. Cantu-Paz, L. Spector, I. Parmee, and H. G. Beyer, editors, Proceedings of the Genetic and Evolutionary Computation Conference, pages 469-476, Las Vegas. Morgan Kaufmann.

[32] Poli, R. (2001). Recursive conditional schema theorem, convergence and population sizing in genetic algorithms. In W.N. Martin and W.M. Spears (Eds.) Foundations of Genetic Algorithms 6, Morgan Kaufmann, San Francisco, CA, 143-163.

[33] Radcliffe, N.J. (1991a). Forma analysis and random respectful recombination. In R.K. Belew and L.B. Booker (Eds.) Proceedings of fourth International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 222-229.

[34] Radcliffe, N.J. (1991b). Equivalence class analysis of genetic algorithms. Complex Systems, 5, 183-205.

[35] Radcliffe, N.J. (1994). The algebra of genetic algorithms. Annals of Maths, and Artificial Intelligence, 10, 339-384.

[36] Radcliffe, N.J. (1997). Schema processing. In Th.Bäck, D.B. Fogel and Z.Michalewicz (Eds.) Handbook of Evolutionary Computation, Oxford, UK: Oxford University Press.

[37] Reeves, C. R., & Rowe, J. E. (2002). Genetic Algorithms: Principles And Perspectives A Guide to GA Theory. New York: Kluwer Academic Publishers.

[38] Saini, J. S. (2002). Intelligent Control Techniques & Applications. Ph.D. Thesis, M. D. University, Rohtak, Haryana (India).

[39] Saini, J. S., Gopal, M. and Mittal, A. P. (2003). An alternative to schemata Count for Genetic Algorithms. All-India Seminar on Power & Energy for Sustainable Growth organized by IE(I) Hr. Centre & CRSCE, Murthal, Sonepat, Haryana (India), Pp. 41-48.

[40] Saini, J. S., Gopal, M., Mittal, A. P., Pawar, R., & Sharma, S. (2006) 'A New Technique for Schemata Count', Asian Conference on Intelligent Systems and Networks, Centre for advanced technology, Jagadhari, Haryana, India, vol. 3, pp. 37-44.

[41] Stephens, C. R., and Waelbroeck, H. (1997). Effective degrees of freedom in genetic algorithms and the block hypothesis. In Th. Bäck (Ed.) Proceedings of 7th International Conference on Genetic Algorithms, Morgan Kaufmann, San Francisco, CA, 34-40.

[42] Stephens, C.R., and Waelbroeck, H. (1999). Schemata evolution and building blocks. Evolutionary Computation, 7, 109-124.

[43] Stephens, C. R., Waelbroeck, H., and Aguirre, R. (1999). Schema as building blocks: does size matter? In W.Banzhaf and C.R. reeves (eds.) Foundations of Genetic Algorithms 5, Morgan Kaufmann , San Francisco, CA, 117-134.

[44] Vose, M. D. (1991). Generalizing the notion of a schema in genetic algorithms. Artificial Intelligence, 50, 385-396.

[45] Vose, M. D., and Liepins, G. E. (1991). Schema disruption. In R.K. Belew and L.B. Booker (Eds.) Proceedings of 4th International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 237-242.

[46] Vose, M. D. (1993). Modeling simple genetic algorithms. In D.Whitley (Ed.) Foundations of Genetic Algorithms 2, Morgan Kaufmann, San Mateo, CA, 63-74.

[47] Vose, M. D. (1994). A closer look at mutation in genetic algorithms. Annals of Maths. and Artificial Intelligence, 10, 423-434.

[48] Vose, M. D. (1995). Modeling simple genetic algorithms". Evolutionary Computation, 3, 453-472.

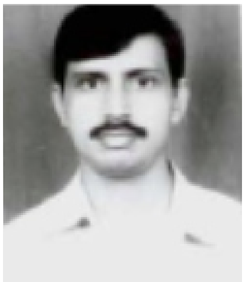
[49] Vose, M. D. (1999). What are genetic algorithms? A mathematical perspective. In L.D. Davis, K.A. De Jong, M.D. Vose and L.D. Whitley (Eds.) Evolutionary Algorithms: IMA Volumes in Mathematics and its Applications, Vol 111, Springer-Verlag, New York, 251-276.

[50] Vose, M. D. (2004). The Simple Genetic Algorithm. Delhi: Prentice Hall of India Pvt. Ltd.

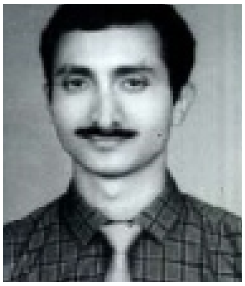
[51] Vose, M. D., and Wright, A. H. (2001). Form invariance and implicit parallelism. Evolutionary Computation, 9, 355-370.

[52] Whitley, D. (1993). An executable model of a simple genetic algorithm. In L.D. Whitley (Ed.) Foundations of Genetic Algorithms 2, Morgan Kaufmann, San Mateo, CA, 45-62.

BIOGRAPHIES



Rajneesh Pawar received his B.E. and M.E. degree in Electrical Engg. from C.R. State college of Engg., Murthal, Sonepat, India in 1999 and 2006. He is presently, Assistant Professor in Dept. of Electrical Engg., D. C. R. University of Science & Technology, Murthal, Sonipat, India. His industrial, teaching and research experience stints at organizations such as BSES Rajdhani Power Limited Delhi, BVCOE Delhi and DCRUST Murthal. His current research interests are in the areas of Soft Computing, Energy Management System, Power Distribution Audit and planning



Dr. J. S. Saini received his B.Sc. (Engg.) degree (Hons.) in Electrical Engg. from Punjab Univ., Chandigarh, India in 1983. He did his M.Tech. in Control Engg. & Instrumentation from I.I.T., New Delhi (India) in 1989. He was awarded Ph.D. degree in Elect. Engg. by M.D. University, Rohtak (Haryana, India) in 2003. He is currently Professor in Dept. of Electrical Engg., D. C. R. University of Science & Technology, Murthal, Sonipat, India. He has published about 50 research papers in journals & Proc. of National & International conferences. His current interests include GAs, Chaotic Systems, Fuzzy Logic, & ANN applications in Control, Instrumentation & Optimization.