Pattern classification with genetic algorithms: Incorporation of chromosome differentiation

Sanghamitra Bandyopadhyay, Sankar K. Pal*

Machine Intelligence Unit, Indian Statistical Institute, 203, B.T. Road, Calcutta 700 035, India

Received 19 September 1996; revised 7 January 1997

Abstract

A new genetic search strategy involving chromosome differentiation into two classes and a restricted form of crossover operation is defined. Its application to multi-dimensional pattern recognition problems is studied. Superiority of the classifier is established for four sets of different artificial and real-life data. Schema analysis is provided.

Keywords: Nature's sexual differentiation; Optimization; Pattern recognition; Restricted crossover; Schema theorem; Speech recognition

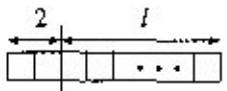
1. Introduction

Genetic Algorithms (GAs) (Goldberg, 1989) are randomized search and optimization techniques guided by the principles of evolution and natural genetics. They are efficient, adaptive and robust search processes, producing near-optimal solutions and have a large amount of implicit parallelism. Application of GA to various pattern recognition problems is described in (Pal and Wang, 1996; Gelsema, 1995). One such application for designing a classifier is to exploit the searching capability of GA for placement of a number of lines for approximating the decision boundaries (Bandyopadhyay et al., 1995). The method involves encoding the parameters of the lines in binary strings called *chromosomes*, in the feature space that yields minimum misclassification. The superiority of the GA based classifier over the k-NN rule (for different values of k) was demonstrated in (Bandyopadhyay et al., 1995) for both artificial and real-life data sets.

In this article, a new genetic search strategy called GACD (GA with chromosome differentiation) is proposed, where the chromosomes in GA are differentiated into two classes. Restricted crossover is applied to these chromosomes where mating is allowed only between individuals belonging to different classes. This is analogous to the biological mating system which involves individuals from two different sexes, male and female. It is proved that the schema theorem (Goldberg, 1989) holds for GACD as well; a short, low-order, above-average schema will receive an increasing number of trials in successive generations.

```
Begin
 1. t=0
 2. initialize population P(t)
 3. compute fitness P(t)
 4. t = t+1
 5. if termination criterion satisfied
    go to step 10
 6. select P(t) from P(t-1)
 7. crossover P(t)
 8. mutate P(t)
 9. go to step 3
10. output best and stop
End
```

Fig. 1. Basic steps in convential GA.



00 – F class bits
01, 10 – M class bits

Fig. 2. Structure of a chromosome in GACD.

Subsequently, the pattern classification problem in N dimensions is formulated using GACD, where the decision boundaries are now approximated by hyperplanes. It is shown that this classifier performs better than the previous GA-based classifier (Bandyopadhyay et al., 1995), for both artificial and real-life data sets.

2. Description of GACD

In GACD (GA with chromosome differentiation), the chromosomes are distinguished into 2 classes, M and F, thereby constituting two separate populations. These are initially generated in such a way that the Hamming distance between the two is maximized.

The basic steps of conventional GA or CGA are shown in Fig. 1. These steps are followed in GACD as well. However, the individual processes are modified in order to incorporate the notion of chromosome discrimination. The modified steps are now discussed in detail.

Population initialization. The structure of a chromosome of GACD is shown in Fig. 2. Here the l bits, termed *data bits*, encode the parameters of the problem. The initial two bits, termed the *class bits*, indicate the class (M or F) of the chromosome. Two separate populations, one containing the M chromosomes (M population) and the other containing the F chromosomes (F population), are maintained over the generations. The sizes of these two populations, p_m and p_f, respectively, may vary. Let $p_m + p_f = p$, where p is fixed (equivalent to the population size of CGA). Initially $p_m = p_f = p/2$. The data bits for each M chromosome are first generated randomly. One of the two *class bits*, chosen randomly, is initialized to 0 and the other to 1. The data bits of the F chromosomes are initially generated in such a way that the Hamming distance between the two populations (in terms of the data bits) is maximum. The Hamming distance between two chromosomes c_1 and c_2 ($c_1, c_2 \in C$), denoted by $h(c_1, c_2)$, is defined as the number of bit positions in which the two chromosomes differ. The Hamming distance between two populations, P_1 and P_2, denoted by $h(P_1, P_2)$, is defined as follows:

$$h(P_1, P_2) = \sum_i \sum_j h(c_i, c_j), \quad \forall c_i \in P_1, \ \forall c_j \in P_2.$$

Fitness computation. Only the l data bits are used to compute the fitness for the chromosomes in a problem-specific manner.

Selection. Selection is performed over all the p $(= p_m + p_f)$ chromosomes (i.e. disregarding the class information), using their fitness values. In other words, all the chromosomes compete with one another for survival. The selected chromosomes are placed in the mating pool.

Crossover. Crossover is applied probabilistically between an M and an F parent chromosome. Each parent contributes one class bit to the offspring. Since the F parent can only contribute a 0 (its class bits being 00), the class of the child is primarily determined by the M parent which can contribute a 1 (yielding an M child) or a 0 (yielding an F child) depending upon the bit position (among the two class bits) of the M parent chosen. This process is performed for both the offspring whereby either two M or two F or one M and one F offspring will be generated.

Crossover is carried on until (a) there are no chromosomes in the mating pool, or (b) there are no M (or F) chromosomes in the mating pool. In the former case the crossover process terminates. In the latter case, the remaining F (or M) chromosomes are mated with the best M (or F) chromosome. Note that if at the start of the crossover process, it is found that the mating pool contains chromosomes of only one class, then the crossover process is discontinued.

Mutation. Bit by bit mutation is performed over the data bits only with probability μ_m. The *class bits* are not mutated.

Note. Elitism is incorporated by preserving the best chromosome, among both the M and F chromosomes, seen till the current generation either inside or outside the population.

3. Schema theorem for GACD

A schema, of same length as the chromosomes being considered, is a string over the alphabet $\{0, 1, \#\}$ (where # represents the don't care symbol). It represents the subset of all the binary strings that match the schema at positions where the latter has 1s and 0s. For example, 11######### is a schema of length 10. It represents all the binary strings of length 10, which have 1s in the first two most significant positions. The *defining position* of a schema is a position where it has either a 1 or a 0. The *defining length* of a schema is the distance between the last defining position and the first defining position, and is obtained by subtracting the first defining position from the last defining position. The *order* of a schema is the number of defining positions in the schema. Thus the defining length and order of the above-mentioned schema are 1 and 2, respectively. The schema theorem (Goldberg, 1989), which is of fundamental importance in GA, states that short, low-order, above-average (characterized by fitness value) schemata will receive an exponentially increasing number of trials in subsequent generations.

In this section we prove that the schema theorem holds for GACD. It is also shown that in certain cases the lower bound of the number of schemata sampled by GACD is larger than or equal to that of CGA. Enumeration of the different terms and terminology are given first, followed by an analysis of GACD with respect to schema sampling. In general the M and F parameters are denoted by subscripts m and f, respectively, while parameters with no subscript denote that these are applicable over both the M and F populations.

3.1. Terminology

p: the total population size which is assumed to be constant.

$p_m(t)$: the M population size at time t.

$p_f(t)$: the F population size at time t; note that $p = p_m(t) + p_f(t)$.

$\bar{f}$: the average fitness of the entire population.

h: a schema.

$\overline{f_h}$: the average fitness of instances of schema h over the entire population.

$\overline{f_m}$: the average fitness of the M population.

$\overline{f_f}$: the average fitness of the F population.

l: the length of a string.

$m(h,t)$: number of instances of schema h in the population at time t.

$m_m(h,t)$: number of instances of schema h in the M population at time t.

$m_f(h,t)$: number of instances of schema h in the F population at time t.

$\delta(h)$: the defining length of the schema h

$O(h)$: the order of the schema h

μ_c: the probability of crossover.

μ_m: the probability of mutation.

Superscripts s and c with any of the above-mentioned symbols indicate the corresponding values after selection and crossover, respectively. It is to be noted that the following equalities will hold for GACD for any value of t.

3.2. Analysis of GACD

Let us consider the effects of each operation, *selection*, *crossover* and *mutation* separately on the number of instances of a schema h.

Selection. Since proportional selection is performed, the probability of selecting a string with fitness f_i is given by $f_i/\sum_{j=1}^{p} f_j$. Hence the number of instances of the schema h after selection will be given by

$$m^s(h,t+1) = m(h,t) * p * \frac{\overline{f_h}}{\sum_{j=1}^{p} f_j} = m(h,t) * \frac{\overline{f_h}}{\bar{f}}. \tag{1}$$

The number of instances of the schema h that will be present in the M and F populations, respectively, must obviously be proportional to the fraction present in the two populations before selection takes place. Or in other words,

$$m_m^s(h,t-1) = m^s(h,t+1) * \frac{m_m(h,t)}{m(h,t)}. \tag{2}$$

Similarly,

$$m_f^s(h,t+1) = m^s(h,t+1) * \frac{m_f(h,t)}{m(h,t)}. \tag{3}$$

Crossover. To analyze the effect of crossover (assuming single-point crossover) on the instances of the schema h, its probability of disruption is first calculated. Instances of the schema that are members of the M population are considered first. The analysis for the F population is analogous. For the present, let us assume that an instance of the schema from the M population, if not disrupted by crossover, is placed in the M population again.

Schema h will most likely be disrupted due to crossover if (i) crossover occurs (with probability μ_c); (ii) crossover site falls within the first and the last defining positions (with probability $\delta(h)/(l-1)$); and (iii) crossover occurs with an instance of some schema h^* in the F population such that h^* is not contained in h (with probability $1 - m_f^s(h, t+1)/p_f^s(t+1)$). (Note that if h^* is contained in h, then crossover can never disrupt h, i.e., schema h will survive in both the offspring. Schema h^*, on the other hand, may not survive crossover at all.)

Taking the above-mentioned three conditions into account, the probability of disruption of h in one instance of the schema may be written as

$$\mu_c * \frac{\delta(h)}{l-1} * \left(1 - \frac{m_f^s(h, t+1)}{p_f^s(t-1)}\right). \tag{4}$$

Hence the probability of survival of one instance of the schema in the M population is given by

$$1 - \mu_c * \frac{\delta(h)}{l-1} * \left(1 - \frac{m_f^s(h, t+1)}{p_f^s(t-1)}\right). \tag{5}$$

Consequently, considering $m_m^s(h, t+1)$ instances (after selection), after crossover we get

$$m_m^c(h, t+1) \geqslant m_m^s(h, t-1)\left(1 - \mu_c * \frac{\delta(h)}{l-1} * \left[1 - \frac{m_f^s(h, t+1)}{p_f^s(t+1)}\right]\right). \tag{6}$$

The "greater than" sign comes because even after disruptive crossover, the schema h may survive.

Similarly the number of instances of h that will survive crossover in the F population is given by the relation

$$m_f^s(h, t+1) \geqslant m_f^s(h, t-1)\left(1 - \mu_c * \frac{\delta(h)}{l-1} * \left[1 - \frac{m_m^s(h, t+1)}{p_m^s(t-1)}\right]\right). \tag{7}$$

It had previously been assumed that if an instance of h is present in the M (or F) population, and if h is not disrupted due to crossover, then it survives in the M (or F) population. In reality the situation may not be so. Let P_1 be the probability that h survives in the M population, when it is originally present in the M population. Hence $(1 - P_1)$ is the probability that h goes to the F population after crossover. Similarly let P_2 and $(1 - P_2)$ be the probabilities that h survives in the M and F populations, respectively, when it is originally present in the F population.

Thus the modified equation for schema survival due to crossover is

$$m_m^{*c}(h, t+1) = P_1 m_m^c(h, t+1) + P_2 m_f^c(h, t+1). \tag{8}$$

The second term is introduced on considering the instances of h that are present in the F population, which survive crossover but are placed in the M population. Similarly,

$$m_f^{*c}(h, t+1) = (1 - P_2) m_f^c(h, t+1) + (1 - P_1) m_m^c(h, t+1). \tag{9}$$

Adding Eqs. (8) and (9) yields the number of instances of h present in the entire population after crossover:

$$m^c(h, t-1) = m_m^c(h, t+1) + m_f^c(h, t+1),$$

or

$$\begin{aligned} m^c(h, t+1) \geqslant\ & m_m^s(h, t+1)\left\{1 - \frac{\mu_c \delta(h)}{l-1}\left[1 - \frac{m_f^s(h, t+1)}{p_f^s(t+1)}\right]\right\} \\ & - m_f^s(h, t+1)\left\{1 - \frac{\mu_c \delta(h)}{l-1}\left[1 - \frac{m_m^s(h, t+1)}{p_m^s(t+1)}\right]\right\}. \end{aligned} \tag{10}$$

Using Eqs. (2) and (3), and the fact that $m(h,t) = m_m(h,t) + m_f(h,t)$, the r.h.s. of inequality (10) may be written as

$$m^s(h,t-1)\left(1-\frac{\mu_c\delta(h)}{(l-1)}\left[1-\left\{\frac{m_m(h,t)m_f^s(h,t-1)}{p_f^s(t-1)m(h,t)}+\frac{m_f(h,t)m_m^s(h,t+1)}{p_m^s(t+1)m(h,t)}\right\}\right]\right).$$

Let us denote the term in the curly brackets by α. Thus we may write

$$m^s_{GACD}(h,t+1) \geqslant m^s(h,t+1)\left(1-\mu_c\frac{\delta(h)}{l-1}[1-\alpha]\right). \tag{11}$$

In this context a slight modification of the schema theorem (Goldberg, 1989) is called for which provides a better lower bound of the number of instances of h that survive after selection and crossover. An instance of schema h may be disrupted due to crossover *iff* it is crossed with an instance of another schema h^* such that h^* is not contained in h and the other conditions for disruptive crossover hold. Accounting for this detail, the disruption probability should be recalculated as

$$\mu_c\frac{\delta(h)}{l-1}\left(1-\frac{m^s(h,t+1)}{p}\right).$$

Hence after selection and crossover

$$m^s_{CGA}(h,t+1) \geqslant m^s(h,t+1)\left\{1-\mu_c\frac{\delta(h)}{l-1}\left[1-\frac{m^s(h,t+1)}{p}\right]\right\}. \tag{12}$$

Let us denote the term $m^s(h,t+1)/p$ by β. Thus

$$m^s_{CGA}(h,t+1) \geqslant m^s(h,t+1)\left\{1-\mu_c\frac{\delta(h)}{l-1}[1-\beta]\right\}.$$

Mutation. Since the conventional bit by bit mutation is applied on the strings with a probability μ_m, the probability of disruption of one bit of the schema is μ_m. Probability of its survival is $1-\mu_m$. Hence the probability of survival of the schema is $(1-\mu_m)^{O(h)}$. Thus the number of instances of the schema h that are present in the population at time $t+1$ (after selection, crossover and mutation) is given by

$$m_{GACD}(h,t-1) \geqslant m^s(h,t+1)\left\{1-\frac{\mu_c\delta(h)}{(l-1)}(1-\alpha)\right\}\left\{(1-\mu_m)^{O(h)}\right\}.$$

Approximating the r.h.s., the inequality may be written as

$$m_{GACD}(h,t+1) \geqslant m^s(h,t-1)\left\{1-\frac{\mu_c\delta(h)}{l-1}(1-\alpha)-\mu_m O(h)\right\}. \tag{13}$$

Similarly, the equation for CGA is given by

$$m_{CGA}(h,t+1) \geqslant m^s(h,t+1)\left\{1-\frac{\mu_c\delta(h)}{l-1}(1-\beta)-\mu_m O(h)\right\}. \tag{14}$$

In order to compare $m_{GACD}(h,t+1)$ and $m_{CGA}(h,t-1)$, we consider the following cases.

Case i. Let $m_m(h,t) = m_f(h,t) = m_1$. In that case $m_m^s(h,t+1) = m_f^s(h,t+1) = m_2$. Note that $m(h,t) = 2m_1$, $m^s(h,t+1) = 2m_2$ and $\beta = 2m_2/p$. Then

$$\alpha = \frac{1}{2m_1}\left(\frac{m_1 m_2}{p_f^s(t-1)} + \frac{m_1 m_2}{p_m^s(t+1)}\right) = \frac{m_2}{2}\left(\frac{p}{p_f^s(t+1)p_m^s(t+1)}\right) = \beta\left[\frac{p^2}{4p_f^s(t+1)p_m^s(t+1)}\right].$$

The minimum value of the term in square brackets is 1 when $p_m^s(t+1) = p_f^s(t+1)$. Hence $\alpha \geqslant \beta$. This in turn indicates that lower_bound($m_{GACD}(h,t+1)$) $\geqslant$ lower_bound($m_{CGA}(h,t+1)$), i.e., the lower bound of the number of instances of some schema h sampled by GACD is better than that of CGA.

Case ii. Let $m_m(h,t) \neq m_f(h,t)$. Let $m_m(h,t) = \gamma m_f(h,t)$ where $\gamma \neq 1$. Then $m_m^s(h,t-1) = \gamma m_f^s(h,t+1)$. Note that $m(h,t) = m_f(h,t)(1-\gamma)$, $m^s(h,t+1) = m_f^s(h,t+1)(1+\gamma)$ and $\beta = m_f^s(h,t+1)(1+\gamma)/p$. Thus, we may write

$$\alpha = \frac{1}{m_f(h,t)(1+\gamma)}\left(\frac{\gamma m_f(h,t) m_f^s(h,t+1)}{p_f^s(t+1)} + \frac{m_f(h,t)\gamma m_f^s(h,t-1)}{p_m^s(t+1)}\right)$$

$$= \frac{m_f^s(h,t+1)\gamma}{(1-\gamma)}\left(\frac{p}{p_f^s(t-1)p_m^s(t+1)}\right) = \frac{m_f^s(h,t+1)(1+\gamma)}{p}\frac{\gamma}{(1+\gamma)^2}\frac{p^2}{p_f^s(t+1)p_m^s(t+1)}$$

$$= \beta\frac{\gamma}{(1+\gamma)^2}\frac{p^2}{p_f^s(t+1)p_m^s(t+1)}.$$

Now, in this case $\alpha \geqslant \beta$ if the following holds:

$$\frac{\gamma}{(1+\gamma)^2}\frac{p^2}{p_f^s(t+1)p_m^s(t+1)} \geqslant 1, \quad \text{or} \quad \frac{p}{\sqrt{p_f^s(t+1)p_m^s(t+1)}} \geqslant \frac{1-\gamma}{\sqrt{\gamma}}. \tag{15}$$

Since the above-mentioned condition (inequality (15)) cannot be always ensured, we cannot conclude that lower_bound($m_{GACD}(h,t-1)$) $\geqslant$ lower_bound($m_{CGA}(h,t+1)$). (Note also that both the functions $(1+\gamma)/\sqrt{\gamma}$ and $p/\sqrt{p_f^s(t+1)p_m^s(t+1)}$ have minimum value 2.)

In order to experimentally compare the values of m_{CGA} and m_{GACD}, an optimization problem is considered. Let $f(x) = x^2$ be the function to be optimized. A population size of 30 (initially 15 M and 15 F chromosomes are considered for GACD) and chromosome length of 10 are taken, with $\mu_c = 0.8$ and $\mu_m = 0.01$. The variation of the number of instances of four schemata with different characteristics is presented over the first five generations in Fig. 3(a)–(d). Note that any instance of the schema 1######### will have x values greater than or equal to 512, whereas, that of 0########## will have values less than or equal to 511. In other words, the resulting strings corresponding to the former one have higher fitness values than those of the latter one.

From the nature of the problem it is obvious that the number of instances of the first three schemata should increase while that of the last schema should decrease in subsequent generations. As expected, this is the case for both CGA and GACD, except a minor temporary experimental deviation in each case (e.g., 2nd generation in Fig. 3(a), 5th generation in Fig. 3(b), 2nd generation in Fig. 3(c), and 2nd generation in Fig. 3(d)). However, the growth (decay) rates for schemata with high (low) fitness values are greater for GACD as compared to CGA, although both could find the optimal value of x (=1023).

In the following sections we demonstrate that GACD is superior to CGA for the pattern classification problem described in (Bandyopadhyay et al., 1995). The classifier was designed using CGA for a two-dimensional feature space ($N = 2$), involving the placement of a number of straight lines for approximating the class boundaries. This conventional GA based classifier was found to be superior to the k-NN rule (for different values of k).

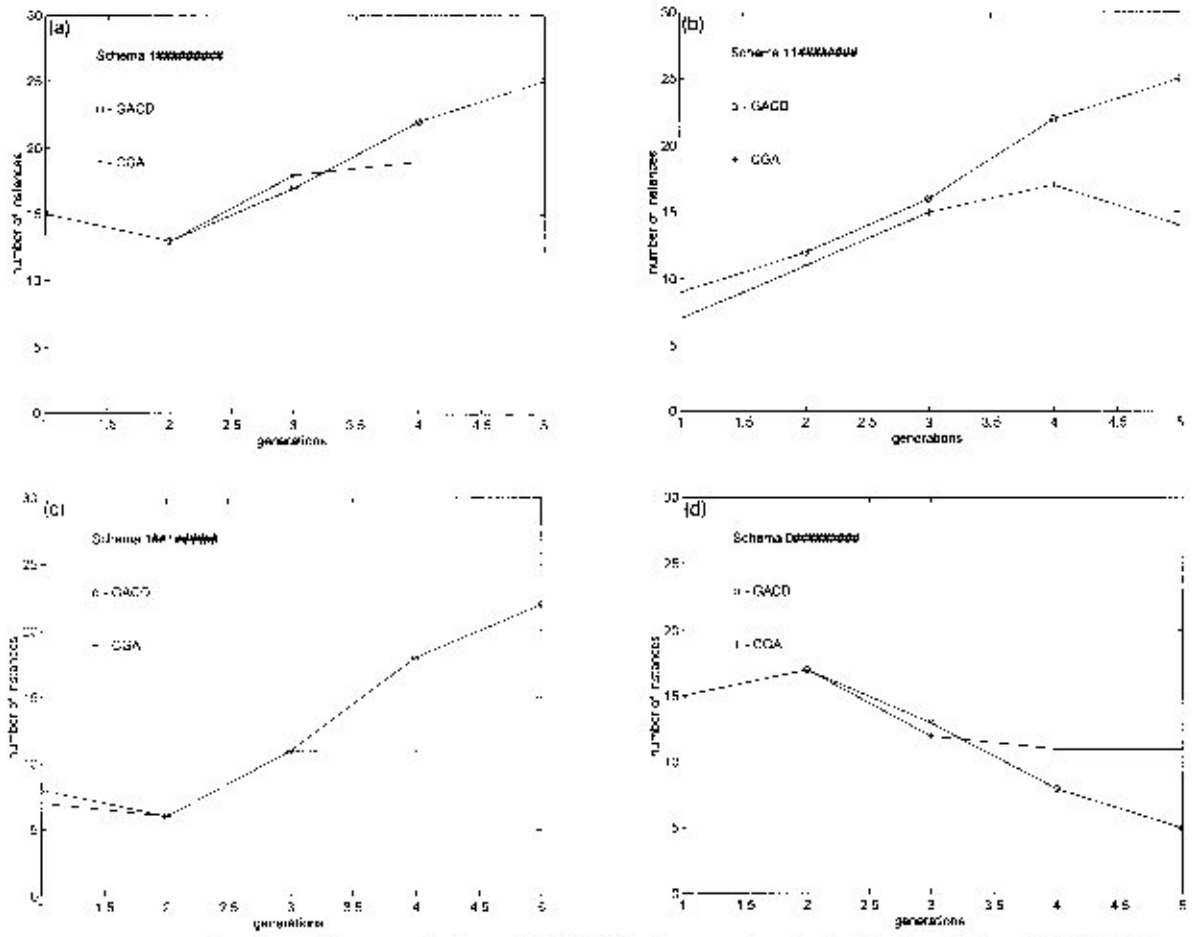


Fig. 3. (a) Variation of number of instances of schema 1######### with generations for the function $f(x) = x^2$. (b) Variation of number of instances of schema 11######## with generations for the function $f(x) = x^2$. (c) Variation of number of instances of schema 1##1###### with generations for the function $f(x) = x^2$. (d) Variation of number of instances of schema 0######### with generations for the function $f(x) = x^2$.

First of all, we extend the said problem of classification for any value of N (Section 4) where a fixed number H of hyperplanes is considered to approximate the decision boundary. (Note that the extension of the scheme of line representation in (Bandyopadhyay et al., 1995) to higher dimensions is not a straightforward one because it would involve a constraint equation that must be satisfied. This, in turn, leads to the complicated issue of getting unacceptable solutions or chromosomes. However, the trigonometric form of representation described in Section 4.1 avoids this problem by being unconstrained in nature.) Finally, a detailed comparison of GACD-based classifier with that of CGA is shown in Section 5.

4. Pattern classification problem

Here we describe the hyperplane representation and fitness computation schemes required for formulating the classification problem in higher dimensions.

4.1. Hyperplane representation

From elementary geometry, the equation of a hyperplane in N-dimensional space $(X_1\text{–}X_2\text{–}\cdots\text{–}X_N)$ is given by

$$x_N \cos\alpha_{N-1} - \beta_{N-1} \sin\alpha_{N-1} = d, \tag{16}$$

where

$$\beta_{N-1} = x_{N-1} \cos\alpha_{N-2} + \beta_{N-2} \sin\alpha_{N-2},$$
$$\beta_{N-2} = x_{N-2} \cos\alpha_{N-3} + \beta_{N-3} \sin\alpha_{N-3},$$
$$\vdots$$
$$\beta_1 = x_1 \cos\alpha_0 + \beta_0 \sin\alpha_0.$$

The various parameters are as follows:

X_i: the ith feature of the sample points.
$(x_1, x_2, \ldots, x_N)$: a point on the hyperplane.
α_{N-1}: the angle that the unit normal to the hyperplane makes with the X_N axis.
α_{N-2}: the angle that the projection of the normal in the $(X_1\text{–}X_2\text{–}\cdots\text{–}X_{N-1})$ space makes with the X_{N-1} axis.
$\vdots$
α_1: the angle that the projection of the normal in the $(X_1\text{–}X_2)$ plane makes with the X_2 axis.
α_0: the angle that the projection of the normal in the (X_1) plane makes with the X_1 axis = 0.
d: the perpendicular distance of the hyperplane from the origin.

Thus the N tuple $(\alpha_1, \alpha_2, \ldots, \alpha_{N-1}, d)$ specifies a hyperplane in N-dimensional space.

Each angle α_j, $j = 1, 2, \ldots, N-1$, is allowed to vary in the range of 0 to 2π. If b_1 bits are used to represent an angle, then the possible values of α_j are

$$0, \delta * 2\pi, 2\delta * 2\pi, 3\delta * 2\pi, \ldots, (2^{b_1} - 1)\delta * 2\pi,$$

where $\delta = 1/2^{b_1}$.

Once the angles are fixed, the orientation of the hyperplane becomes fixed. Now only d must be specified in order to specify the hyperplane. For this purpose the hyper rectangle enclosing the sample points is considered. The hyperplane passing through one of the vertices of the hyperrectangle and having a minimum distance, $d_{\min}$, from the origin is specified as the base hyperplane. Let *diag* be the length of the diagonal of the hyperrectangle and let the bits used for specifying d be capable of generating values, say *offset*, in the range $[0, diag]$. (The number of bits again controls the precision of d.) Then $d = d_{\min} + offset$.

4.2. Fitness computation

The hyperplanes encoded in the chromosomes partition the feature space into several regions. The class associated with each region is determined in the following manner: for each region, the class of the maximum number of training data points that lie in this region is identified. The region is then assumed to be associated with this class. Points of other classes that lie in this region are considered to be misclassified. The number of misclassified samples in each region is computed in this manner, yielding the total number of misclassified samples, *miss*, associated with the particular arrangement of the hyperplanes. The fitness of the corresponding chromosome is then computed as $(n - miss)$, where n is the size of the training data.

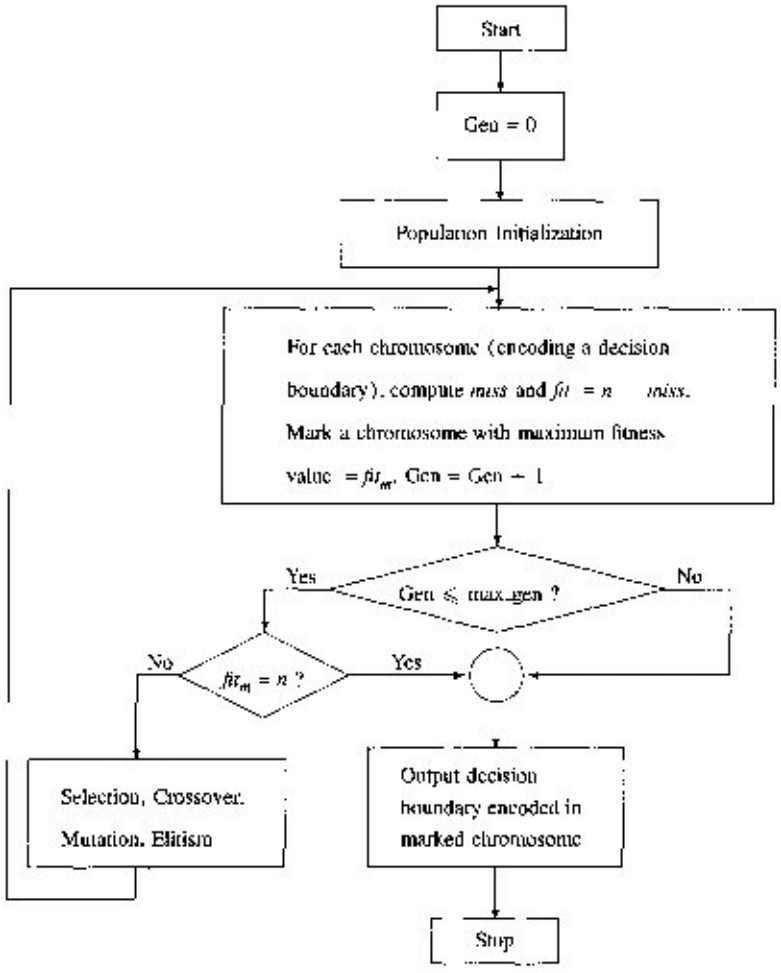


Fig. 4. Block diagram of genetic pattern classifier.

Once the hyperplane representation and fitness computation schemes are fixed, the GACD algorithm can be applied to the pattern classification problem directly. The block diagram of a genetic pattern classifier is shown in Fig. 4 for the convenience of the readers. Here "Gen" and "max_gen" indicate the current generation number and maximum number of generations, respectively.

The following section provides a comparative study of the GACD-based classifier (*GACD-classifier*) and the previous GA-based classifier (*GA-classifier*) for several data sets.

5. Implementation and experimental results

A population size of 20 is chosen. Two different values, namely 0.8 and 0.5, are used for the crossover probability for two sets of experiments. The mutation probability is varied in the range $[0.01, 0.333]$ over every 100 generations for a maximum of 1500 generations. Initially μ_m has a high value, thereby ensuring sufficient diversity in the population. Subsequently it is decreased gradually to the minimum value, when the algorithm is allowed to make a detailed search in the solution space. The mutation probability is again increased

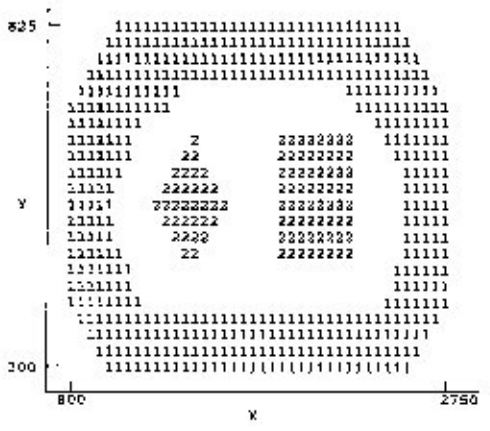


Fig. 5. Artificial data set ADS1.

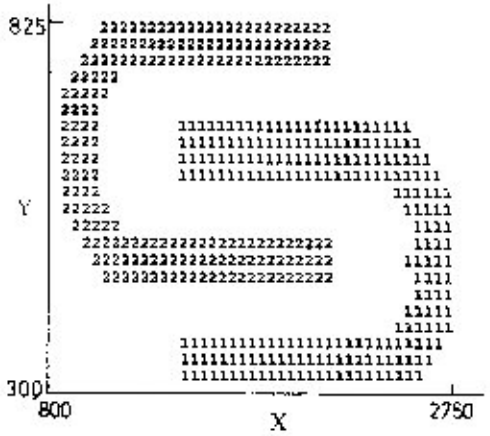


Fig. 6. Artificial data set ADS2.

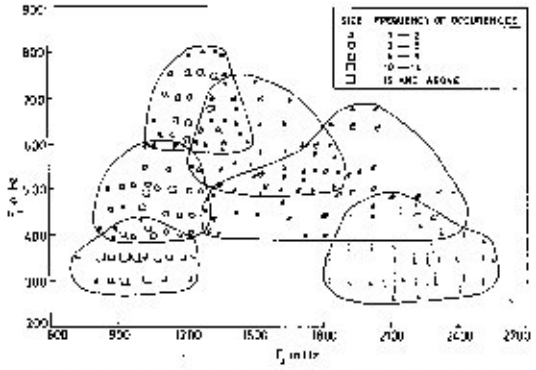


Fig. 7. Real-life speech data, *Vowel data*, in the first and second formant frequency planes.

indicating an increase in the randomness of the search. In case the optimal string was already obtained, elitism ensures that it is not lost.

Results on four data sets are presented. The 2-dimensional artificial data sets, ADS1 (Fig. 5) and ADS2 (Fig. 6), consist of 557 and 417 points respectively belonging to two classes. The real-life speech data, *Vowel data* (Pal and Mitra, 1992), consists of three feature values (corresponding to the three formant frequencies) and six classes $\{\delta, a, i, u, e, o\}$. Fig. 7 shows the class structures in the first and second formant frequency plane. *Iris data* comprises 150 samples having four features and belonging to three classes with 50 points in each class.

The results shown (Tables 1 to 3) are the average values obtained over 10 simulations of *GACD-classifier* (abbreviated GACD) and *GA-classifier* (abbreviated CGA). 10% of the data set is used for training. The remaining 90% data points are used for testing.

Note. (i) The tables show the average number of data points that are misclassified by the classifiers during training (*miss*), and the percentage of the correctly recognized test data (*recogn. score*).

(ii) The values of H (number of hyperplanes) are chosen intuitively based on previous experiments (Bandyopadhyay et al., 1995) with the data sets.

For all the data sets it is seen from Tables 1 and 2 that in terms of the recognition scores, *GACD-classifier* performs better than *GA-classifier* both for the training and test data. A point to be mentioned here is that

Table 1
Comparative results for the pattern classification problems for $\mu_c = 0.8$

Data set	H	Avg. number of generations		Training (*avg. miss*)		Testing (*avg. recogn. score*)	
		GACD	CGA	GACD	CGA	GACD	CGA
ADS1	5	573.2	854.3	0	0.7	92.23	91.89
ADS2	5	620.5	911.5	0	0.3	90.53	86.65
Iris	5	50.8	49.9	0	0	93.41	89.99
Vowel	5	1500	1500	7.4	9.4	75.98	71.32

Table 2
Comparative results for the pattern classification problems for $\mu_c = 0.5$

Data set	H	Avg. number of generations		Training (*avg. miss*)		Testing (*avg. recogn. score*)	
		GACD	CGA	GACD	CGA	GACD	CGA
ADS1	5	578.33	825.5	0	0.4	93.09	92.89
ADS2	5	611.25	765.5	0	0.3	90.23	85.32
Iris	5	43.2	44.4	0	0	93.78	90.21
Vowel	5	1500	1500	8.0	10.7	74.43	73.66

Table 3
Variation of the avg. test recognition score for *Vowel data* with H for $\mu_c = 0.8$

H	Training (avg. *miss*)		Testing (avg. *recogn. score*)	
	GACD-classifier	*CGA-classifier*	*GACD-classifier*	*CGA-classifier*
4	12.5	13.4	72.90	73.88
5	7.4	9.4	75.98	71.32
6	7.0	9.2	71.25	71.25
7	5.5	7.5	69.80	69.33

GACD-classifier attained zero misclassification in all the 10 simulations, whereas this is not true for the *GA-classifier*. For example, it took 7 and 6 simulations corresponding to $\mu_c = 0.8$ and $\mu_c = 0.5$ in the case of ADS1, and 8 and 7 simulations corresponding to $\mu_c = 0.8$ and $\mu_c = 0.5$ for ADS2 when zero misclassification was obtained.

Since the *Vowel data* has a considerable amount of overlap, both *GACD-classifier* and *GA-classifier* fail to attain zero misclassification even until 1500 generations for both values of μ_c. Note that for ADS1 and ADS2, *GACD-classifier* performs considerably better than *GA-classifier* in terms of the average number of generations required to obtain zero misclassification. For *Iris data*, it is comparable.

To demonstrate the variation of the recognition score with H, we present the results for only *Vowel data*, since it is a real-life, overlapping data. Table 3 shows the results for $\mu_c = 0.8$. As expected, it is found that the misclassification during training gradually decreases as H increases for both *GACD-classifier* and *GA-classifier*. However, the results on the test data indicate that increasing the number of hyperplanes does not necessarily increase the generalization capability of the classifiers.

6. Discussion and conclusions

A new methodology GACD, based on chromosome differentiation in genetic algorithms, has been described, and its application to the problem of pattern classification in N-dimensional space has been studied. It is found that GACD results in an improvement of performance of the GA-based classifier in (Bandyopadhyay et al., 1995).

The schema theorem is proved to hold for GACD. It has also been theoretically established that in many cases the lower bound of the number of instances of a schema h sampled by GACD is greater than or equal to that of CGA. Because of this, GACD is better able to exploit the information gained so far. Again, initializing the M and F populations in such a way so as to maximize the Hamming distance between them, and allowing mating between individuals from these two dissimilar populations, enhances the exploration capability of GACD. As a result, GACD appears to strike a better balance between exploration and exploitation, which is crucial for any adaptive optimization technique, thereby giving it an edge over the conventional GA.

An analogy to the GACD methodology developed here can be found in sexual differentiation in nature. Nature, like in GACD, generally does not permit unrestricted mating in living beings. In human genetics, X and Y chromosomes are used to distinguish between the male and female classes. Each parent contributes one chromosome to the offspring; the sex of the offspring is primarily determined by the male parent. Similarly, in GACD, two bits are utilized for differentiating the chromosomes into two classes, with the M parent being primarily responsible for the class of the offspring. However, note that the similarity between the natural and the artificial genetic system is very naive.

It has been proved in (Bhandari et al., 1996) that any elitist model of GAs will definitely converge to the optimal string as the number of iterations tends to infinity provided the probability of going from any population to the one containing the optimal string is greater than zero. Note that the conventional mutation operation alone ensures that this probability is greater than zero. Since GACD utilizes the conventional mutation operation and incorporates elitism, the above-mentioned criteria are fulfilled. Thus GACD is also guaranteed to provide the optimal string as the number of iterations goes to infinity.

Acknowledgements

This work was carried out when Ms. Sanghamitra Bandyopadhyay held a fellowship awarded by the Department of Atomic Energy, Govt. of India.

References

Bandyopadhyay, S., C.A. Murthy and S.K. Pal (1995). Pattern classification using genetic algorithms. *Pattern Recognition Letters* 16 (8), 801–808.

Bhandari, D., C.A. Murthy and S.K. Pal (1996). Genetic algorithm with elitist model and its convergence. *Internat. J. Pattern Recognition and Artificial Intelligence*. Accepted.

Gelsema, E.S. (Ed.) (1995). *Pattern Recognition Letters* 16 (8). Special Issue on Genetic Algorithms.

Goldberg, D.E. (1989). *Genetic Algorithms: Search, Optimization and Machine Learning*. Addison-Wesley, New York.

Pal, S.K. and S. Mitra (1992). Multilayer perceptron, fuzzy sets and classification. *IEEE Trans. Neural Networks* 3 (5), 683–697.

Pal, S.K. and P.P. Wang (Eds.) (1996). *Genetic Algorithms for Pattern Recognition*. CRC Press, Boca Raton.