Proceedings of the 6th World Congress on Intelligent Control
and Automation, June 21 - 23, 2006, Dalian, China

A Hybrid Genetic Algorithm with Fitness Sharing Based on Rough Sets Theory

Jihua FENG and Wenjuan LI
Department of Electronic Engineering, Information School
University of Yunnan
Kunming, 650091, China
fengjihua@avl.com.cn

Xinling SHI and Jianhua CHEN
Department of Electronic Engineering, Information School
University of Yunnan
Kunming, 650091, China

Abstract - **This paper presents a new method integrated sharing genetic algorithm (SGA), rough sets theory (RST) and bit-climbing algorithm for multimodal function optimization. We apply the SGA to complete the globe search and form niches which indicate the promising locations. Then, we utilize the strong qualitative analysis ability of rough sets to identify these niches. Finally, the bit-climbing algorithm is used to complete the local search and refine the solution in each of niches. The experiment has proved that using this approach to solve the multimodal function optimization is efficient both in robustness and in accuracy.**

Index Terms - Sharing genetic algorithm, Rough sets, bit-climbing algorithm, niche count.

I. INTRODUCTION

Real optimization problems often lead to multimodal domains and require the identification of multiple optima, either global or local. For this purpose, niching methods extend simple GA's by promoting the formation of stable subpopulations in the neighbourhood of optimal solutions. It was originally introduced by Holland [1] and improved by Goldberg and Richardson [2]. Niching methods have been developed to reduce the effect of genetic drift resulting from the selection operator in the standard GA. They maintain population diversity and permit the GA to explore all peaks in parallel. On the other hand, they prevent the GA from being trapped in local optima of the search space. However, a fundamental problem is: where are the niches? How do we decide if two individuals are in the same niche, and should therefore have their fitness values shared? To do this accurately, we need to know where each niche is, and how big it is.

The motivation of using rough sets [3] [4] to process the results of sharing GA is driven mainly by its usefulness in the context of vagueness and uncertainty. Particularly, RST has provided an array of tools which turned out to be especially adequate for conceptualization, organization, classification and analysis of the various types of data, when dealing with inexact, uncertain or vague knowledge and when discovering hidden patterns and regularities in applications related to information systems and Artificial Intelligence.

Based on rough sets, we develop a hybrid genetic algorithm with powerful searching ability and improve the accuracy and stability of algorithm by bit-climbing algorithm [5].

The paper is organized as follows. The background of the fitness sharing method is described in Section II. The preliminary principles of rough sets theory are reviewed in Section III, while the test problems are illustrated in Section IV. The experimental results are presented and discussed in Section V, followed by the conclusion in Section VI.

II. FITNESS SHARING THEORY

Fitness sharing accomplishes niching by reducing the fitness of an individual according to the presence of similar individuals within the population. The concept of similarity between two individuals is implemented by defining a metric on either the genotypic or the phenotypic space and by setting a threshold value σ which represents the maximal distance among individuals to be considered similar, i.e., belonging to the same species and, hence populating the same niche. The shared fitness of an individual at generation is given by

$$f_{sh}(a_i) = f(a_i)/m_i, i = 1,2,...,N \quad (1)$$

where $f(a_i)$ is the raw fitness of the individual a_i, and m_i is the niche count which measures the approximate number of individuals with whom the fitness is shared. The niche count is calculated by summing a sharing function over all members of the population. It is given by

$$m_i = \sum_{j=1}^{N} sh(d_{ij}), i = 1,2,...,N \quad (2)$$

where $sh(d_{ij})$ is the sharing function. Such a function measures the similarity between two individuals and has the following form:

$$sh(d_{ij}) = \begin{cases} 1 - (\frac{d_{ij}}{\sigma})^{\alpha}, & if\ d_{ij} < \sigma \\ 0, & otherwise \end{cases} \quad (3)$$

where d_{ij} is defined as distance function, which represents the distance between the individual i and the individual j, and σ denotes the threshold of dissimilarity (also the distance cutoff or the niche radius) and α is a constant parameter which regulates the shape of the sharing function. In general, d_{ij} is computed by using a metric in the phenotypic space (real values), while the choice of the values for the parameters σ depends on the fitness landscape.

III. ROUGH SETS THEORY

A. Information system

Formally, an information system can be seen as a system S=<U,A,V,f>.Where U is the universe (a finite set of objects, $U = \{x_1,..., x_n\}$); and A is the set of attributes (features, variables). Each attribute $a \in A$ (attribute a belonging to the considered set of attributes A) defines an information function $f : U \to V$, where V is the set of values of a, called the domain of attribute a. An information system can be represented with a decision table. In a decision table, the columns are the values of each attribute, and the rows are the objects.

Definition 1**:** For an information system S=<U, A, V, f>, let $B \subseteq A$**,** an indiscernibility relation is defined as follows:

$$\text{ind(B)}=\{(x_1, x_2) \in \text{U} \times \text{U} \mid \text{b}(x_1)=\text{b}(x_2), \forall \text{ b} \in \text{B}\} = \bigcap_{b \in \mathbf{B}} \text{ind} \quad (4)$$

Definition 2**:** For any element x ∈ **U**, B ∈ A**,** the equivalence class of x in relation B represented as $\lfloor x \rfloor_B$**,** and is defined as follows:

$$[x]_B = \{y \in U \mid (x, y) \in ind\,(B)\} \quad (5)$$

Definition 3**:** Let X ⊂ **U***,* the lower approximation of X in S for B is defined as:

$$\underline{B}(X) = \bigcup \{Y \in U \mid ind\,(B) \mid Y \subseteq X\} \quad (6)$$

and the upper approximation of *X* in S for B is defined as:

$$\overline{B}(X) = \bigcup \{Y \in U \mid ind\,(B) \mid Y \cap X \neq \phi\} \quad (7)$$

Definition 4: Further more, we define the positive region and the negative region of X with respect to B*.*

$$Pos_B(X) = \underline{B}(X), Neg_B(X) = U - \overline{B}(X) \quad (8)$$

B. Core and Reducts of attributes

Definition 5: Let R∈ A be an equivalence relation, r∈ R, the attribute r is superfluous if ind(R) = ind(R - { r }) ,otherwise r is indispensable in R .

Definition 6: For a subset of attributes P∈R , if there exists Q = P - { r } , Q⊆ P , satisfying ind(Q) = ind(P), and Q is independent, then Q is called a reduct of P , denoted as RED(P).

Definition 7: The set of all indispensable attributes in P contained by all reducts is called the core of P*,* denoted as:

$$CORE(P) = \bigcap RED(P) \quad (9)$$

C. Rule extraction from GA

The aim of the rule extraction is to get rules in the following form from input-output data historians:

$R^{(k)}$: If x_1 is , **is** A_2**,** … x_m is A_m, then **Y** is Y_k.

When the sharing GA outputs the end population, which includes the structure of the niches and the values of current optima, we need find out this information**.** Hence, rough set theory is used here to extract reductive rules from individual set. The steps of rule extraction are as follows:

a. Get initial information system from sharing GA's outputting population

An information system is considered as a decision table, which can be represented as a finite data table. The columns are labelled by attributes, the rows by objects.

Considering the end population in the GA, we suppose there are N pieces of individual, in the form {C1, C2, and C3, D}, where C1 is the fitness values of individual, C2 is the niching count (m_i) of individual and C3 is the individual's real value. Finally, D represents which niche the individual is belonged to. So the initial decision system can be represented with Table1 in which the condition attributes set is C= {C1, C2, C3} and decision attributes D= {niche}.

b. Discretization of continuous attribute values in rough set

To discrete a continuous valued attribute, the domain of the attribute should be partitioned into several intervals. A boundary between intervals is called a cutpoint. Each interval is treated as a discrete value. (Table2)

In this paper, we apply equal-width-interval method to discrete the attribute.

D. The simplification of decision table

a. Dealing with inconsistent rules

If there are two rules with the same condition attribute values, but different decision attribute value, then these two rules *are* inconsistent. Keep the one with higher confidence, and delete the other.

b. Dealing with superfluous attributes

For attribute set *C*, *C*= $\{C_1, C_2,\ldots,C_n\}$, $\bigcap (C - \{C_i\}) = \bigcap C$, the C_i is superfluous in *C*. Namely, in order to judge whether C_i is superfluous, we will remove C_i from the set. If no inconsistent rule appears, C_i is called superfluous.

TABLE I
The initial information system

Individual	C1	C2	C3	niche
1	fitness(1)	m(1)	x(1)	1
2	fitness(2)	m(2)	x(2)	1
3	fitness(3)	m(3)	x(3)	1
4	fitness(4)	m(4)	x(4)	2
5	fitness(5)	m(5)	x(5)	2
6	fitness(6)	m(6)	x(6)	2
7	fitness(6)	m(6)	x(6)	2
8	fitness(8)	m(8)	x(8)	2.
…	…	…	…	…
i	fitness(i)	m(i)	x(i)	3
…	…	…	…	…
N	fitness(N)	m(N)	x(N)	5

TABLE2
The example of discrete table (50 populations)

Individual	C1	C2	C3	niche
1	1	3	1	1
2	10	13	1	1
3	11	13	1	1
4	10	13	1	2
5	10	13	1	2
6	10	13	1	2
7	1	5	2	2
8	10	5	2	2
…	…	…	…	…
50	3	4	6	5

IV. TEST PROBLEMS

A. *Test Functions*

The set of functions used in our experiment is made of four test functions, namely F1, F2, F3, and F4 [6], whose fitness landscapes exhibit an increasing complexity. The advantage of using these functions is twofold.

a. Except for, this set of functions is largely adopted in the literature as test-bed for GA with niching and, therefore, facilitate the analysis of the obtained results.

b. For each function the fitness landscape is *a priori* known and, therefore, also the values of optima and can be estimated and used as ground truth to assess the effectiveness of our method.

Functions are defined as follows:

$$F\,1 = \sin^6(5\pi x),\ x \in [0,1] \tag{10}$$

This function defined on [0, 1] consists of five equally spaced peaks of uniform height. Maxima are located at approximate x values of 0.1, 0.3, 0.5,0.7, and 0.9. All peaks are of height 1.0.

$$F2 = e^{-2\times\ln 2\times((x-0.1)/0.8)^2} \times \sin^6(5\pi x),\ x \in [0,1] \tag{11}$$

F2 is also defined on [0, 1] and consists of five equally spaced peaks of nonuniform height. Maxima are located at approximate x values of 0.1, 0.3, 0.5, 0.7, and 0.9. Maxima are of approximate height 1.000, 0.9170, 0.7071, 0.4585 and 0.2500 respectively.

$$F3 = \sin^6(5\pi(x^{3/4} - 0.05)),\ x \in [0,1] \tag{12}$$

F3 consists of five unequally spaced peaks of uniform height. Maxima are located at approximate x values of 0.0797, 0.2467, 0.4506, 0.6814, and 0.9339. All peaks are of height 1.0.

$$F4 = e^{-2\times\ln 2\times((x-0.1)/0.8)^2} \sin^6(5\pi(x^{3/4} - 0.05)), x \in [0,1] \tag{13}$$

F4 is defined on [0, 1] and consists of five unequally spaced peaks of nonuniform height. Maxima are located at approximate x values of 0.0797, 0.2467, 0.4506, 0.6814, and 0.9339. Maxima are of approximate height 0.9991, 0.9545, 0.7662, 0.4809 and 0.2217 respectively.

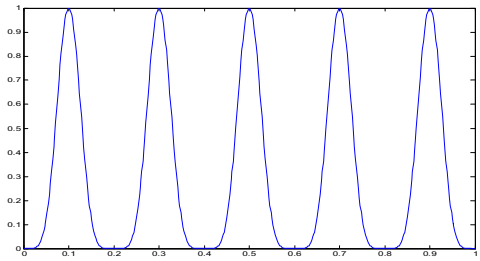

Fig1. Function F1

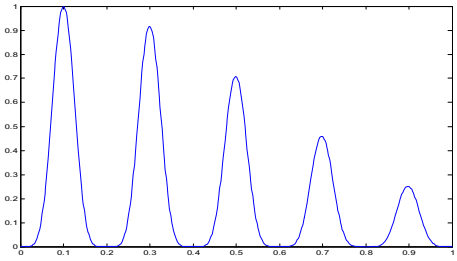

Fig2. Function F2

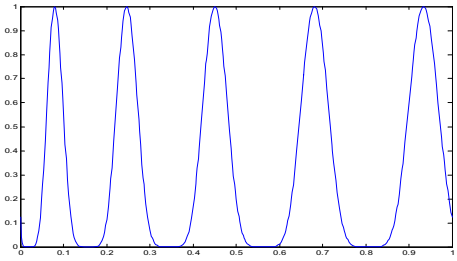

Fig3. Function F3

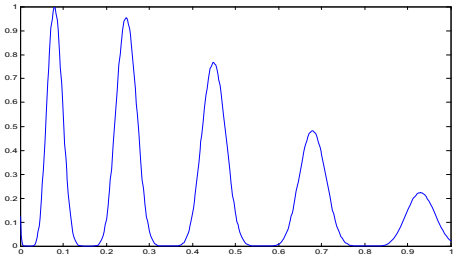

Fig4. Function F4

B. *Performance Criteria*

a. Maximum Peak Ratio[7]:

The maximum peak ratio is the sum of the fitness of the local optima identified by the niching technique divided by the sum of the fitness of the actual optima in the search space. An optimum is considered to be detected if it is within a niche radius of the real optimum and if its fitness value is at least 80% of the real optimum. When an optimum is not identified, the local optimum value is set to zero.

b. Effective Number of Peaks Maintained:

In order to compare standard sharing GA and our hybrid GA, we also consider the effective number of optima maintained at the end of the search according to the previous assumptions.

c. Number of Fitness Function Evaluations:

In many applications, the computational cost of fitness functions can be very expensive. Therefore, we are interested in evaluating the efficiency of algorithms at limited numbers of fitness evaluations.

d. The best solution (global optimum) in each generation:

For comparison with the performance of pure sharing GA and hybrid GA, we also provide the curve of fitness vs. generation (see Fig.6, Fig.7). It expresses the algorithms' refining and searching ability.

V. EXPERIMENTAL STUDY

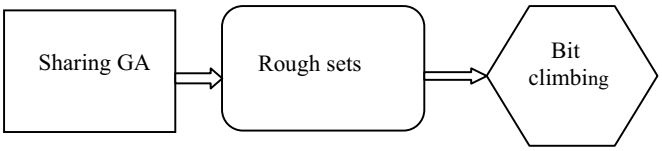

Fig5: Hybrid GA's Program flow chart

A. Genetic algorithm's parameters and operators

The common binary encoding with codelength 30 bits is used for the coding of genotype in our experiments (using elitism strategy). The parameter of the niche radius is set to 0.1 (the threshold of dissimilarity), and the constant parameter α is set to 2 (according to [8]).

The common binary encoding with codelength 30 bits is used for the coding of genotype in our experiments (using elitism strategy). The parameter of the niche radius is set to 0.1 (the threshold of dissimilarity), and the constant parameter α is set to 2 (according to [8]).

a. Selection operator: We apply Normalized Geometric Ranking Selection to investigate the efficiency of the sharing GA. Normalized Geometric Ranking Selection defines p_i for each individual by:

$$p_i = q'(1-q)^{r-1} \tag{10}$$

where q is the probability of selecting the best individual, r is the rank of the individual, $r=1$ is the best. q' is defined as

$$q' = \frac{q}{1-(1-q)^{Pop}} \tag{11}$$

where Pop is the population size.

b. Crossover operator: In this study, we choose simple crossover for binary code. Simple crossover generates a random number r which is uniformly distributed on [1,M] and creates two new individuals (x_i' and y_i') according to the following equations:

$$x_i' = \begin{cases} x_i, & if \quad i < r \\ y_i & otherwise \end{cases} \tag{12}$$

$$y_i' = \begin{cases} y_i & if \quad i < r \\ x_i & otherwise \end{cases} \tag{13}$$

c. Mutation operator: Binary mutation is used to flips each bit in every individual in the population with a probability 0.01 according to the following equation:

$$x_i' = \begin{cases} 1-x_i, & if \quad p_m < 0.01 \\ x_i, & otherwise \end{cases} \tag{14}$$

where p_m is a random variable uniformly distributed on [0,1].

B. Bit-climbing procedure:

Begin
  t=0
  initialize a=($a_1, a_2, \ldots, a_L$)
  L:= length(a)
  calculate f(a)
  repeat
  generate random sequence: bit_list={ $j_1, \ldots, j_l, \ldots, j_L$ }, ($j_l \in \{1,2,\ldots L\}$)
  for l=1 to L
    begin
    a'_{jl} :=1- a_{jl}
    $a' =: \{a_1, \ldots, a_{jl-1}, a'_{jl}, a_{jl+1}, \ldots, a_L\}$
    calculate f(a')
    if f(a')>f(a)
    a:= a'
  end
  t:=t+1
  until (stop_criteria) or t=MAX
end

C. Experimental results

Ten round of tests are made with different populations generated randomly for each scheme to take into account the stochastic nature of GA. The average results of these tests are presented.

a. The performance of hybrid GA:

The Table3 presents the results of the hybrid GA calculated on F1, F2, F3, and F4.

TABLE3: the searching results of hybrid GA
(50 individuals and 100 generations)

F1	Solutions	fitness
Peak1	0.099936	1
Peak2	0.29995	1
Peak3	0.49972	1
Peak4	0.70006	1
Peak5	0.89993	1
F2	Solutions	fitness
Peak1	0.099722	0.99994
Peak2	0.29958	0.91722
Peak3	0.49902	0.7078
Peak4	0.6985	0.45953
Peak5	0.89771	0.25101
F3	Solutions	fitness
Peak1	0.079888	0.99995
Peak2	0.24709	0.99984
Peak3	0.45078	0.99999
Peak4	0.68139	1
Peak5	0.93377	0.99999

F4	solutions	fitness
Peak1	0.079738	0.99911
Peak2	0.24633	0.95459
Peak3	0.44987	0.76682
Peak4	0.67894	0.48234
Peak5	0.92982	0.22341

b. Number of peaks Maintained and Maximum Peaks ratio:

Table4 presents the results on function F1, F2, F3, and F4 for sharing GA and hybrid GA. The experiment is performed using 100 individuals and 100 generations.

TABLE 4(100 individuals and 100 generations)

Test functions	Number of Peaks Maintained		Maximum Peaks Ratio	
	sharing	hybrid	sharing	hybrid
F1	4.9	5	0.96299	0.99998
F2	4.8	4.9	0.97211	0.99161
F3	4.8	4.9	0.92597	0.97998
F4	4.7	4.9	0.96808	0.98151

The experiment results on 50 individuals and 100 generations are presented inTable5.

TABLE 5(50 individuals and 100 generations)

Test functions	Number of Peaks Maintained		Maximum Peaks Ratio	
	sharing	hybrid	sharing	hybrid
F1	4.9	5	0.92238	0.99997
F2	4.6	4.8	0.95423	0.98703
F3	4.9	5	0.97002	0.99983
F4	4.6	4.8	0.95543	0.98807

The results in Table4 and Table5 clearly show that the hybrid algorithm can produce good results. Our algorithm can locate the peaks emerging at different locations of the landscape. Especially, F4 function is difficult for standard sharing GA due to its unequally spaced peaks and their nonuniform heights. In this function, sharing mechanism can not be satisfied and the formation of stable subpopulation is very difficult. Therefore, sharing GA often fails to find all solutions. However, hybrid algorithm locates optima without difficulty due to its better search ability.

For the accuracy of results, the hybrid GA is also better than standard sharing GA. The quality of solutions is evidently enhanced by employing bit-climbing algorithm.

In our experiments we notice that increasing the number of fitness function evaluations have less influence on hybrid GA than on sharing GA after certain generations.

c. The best individual fitness (global optima) in each generation:

In this experiment, both of the algorithms are tested using 50 individuals and 50 generations on F2 and F4 respectively (Both functions only have a maximum peak).

As shown in Fig.6 and Fig.7, at the beginning of evolution, the fitness of sharing GA is averagely larger best than that of hybrid GA. However, the hybrid GA's performance is rapidly increasing and its fitness finally surpasses the sharing GA's after the approximately fifth generation.

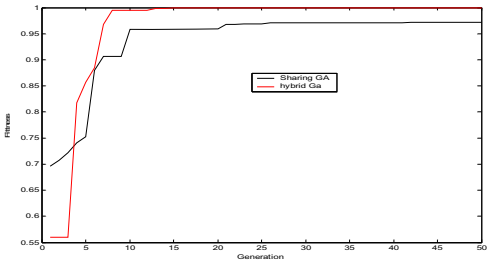

Fig.6 Generation vs. fitness (run on F2)

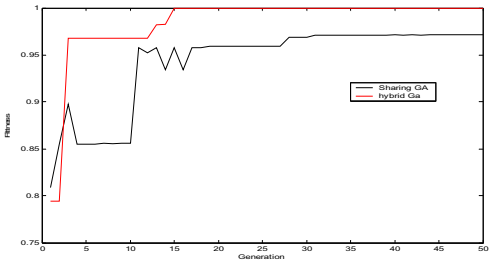

Fig.7 Generation vs. fitness (run onF4)

In general (see Fig.6, Fig.7), when the evolution time increases, the hybrid GA's search becomes faster and faster, so we can use limited generations to get higher precision.

VI. CONCLUSION

In this paper, a hybrid GA algorithm based on rough sets is proposed to identify and search multiple niches (peaks) efficiently in a multimodal domain. It is found that hybrid algorithm performs better than standard sharing methods for multiple optima identification when applied to several numerical functions. The experiments have also proved that the proposed method can overcome some difficulties in multimodal optimization problems. The application of new hybrid GA to optimization of multidimensional functions will be our future work.

REFERENCES

[1] J. H. Holland, "Adaptation in Natural and Artificial Systems," Ann Arbor, MI: Univ. of Michigan Press, 1975.

[2] D. E. Goldberg and J. Richardson. "Genetic algorithms with sharing for multimodal function optimization," In Proc. 2nd Int. Conf. Genetic Algorithms, J. J. Grefensette, Ed. Hillsdale, NJ: Lawrence Erlbaum, 1987, pp. 41–49.

[3] Pawlak Z. "Rough classification,"Int.J. Human-computer studies (1999) 81,369-383, ©1999 Academic Press

[4] B. Walczak, D.L. Massart. "Rough sets theory," 1999 Elsevier Science B.V.

[5] Davis,L. "Bit-climbing, representational bias, and test suite design," In Proceedings of Fourth International Conference on Genetic Algorithms (ICGA 4), Belew, R.K., and Booker, L. B.(eds.), San Mateo, CA: Morgan Kaufmann Publishers,1991:18-23

[6] S. W. Mahfoud. "Niching methods for genetic algorithms,"Ph.D. dissertation, Univ. of Illinois, Urbana-Champaign, 1995.

[7] B. L. Miller and M. J. Shaw, "Genetic algorithms with dynamic niche sharing for multimodal function optimization," in *Proc. 1996 IEEE Int. Conf. Evolutionary Computation*, 1996. Piscataway, NJ: IEEE Press, pp. 786–791.

[8] Beasly,d. Bull, D.r., and Martin, R.R. "A sequential niche technique for multimodal function optimization," Evolutionary Computation, 1993,1(2): 101-125