

Hybrid Genetic Algorithms and Clustering

Francisco Mota Filho
DCA-FEEC-UNICAMP
State University of Campinas
Campinas – SP, Brazil
francisc@dca.fee.unicamp.br

Fernando Gomide
DCA-FEEC-UNICAMP
State University of Campinas
Campinas – SP, Brazil
gomide@dca.fee.unicamp.br

Abstract

This paper introduces a hybrid genetic algorithm that uses fuzzy c -means clustering technique as a mechanism to reduce fitness evaluations and to preserve solution quality. Population clustering provides a means to evaluate only the representative individual of each cluster instead of the whole population. The remaining individuals are indirectly evaluated. The aim is to maintain reasonable population size and to obtain near-optimal solutions. This is an important issue especially in large-scale, complex optimization and decision-making problems.

Keywords: clustering, genetic algorithm.

1 Introduction

Genetic algorithm (GA) was first proposed by John Holland in early 1970s. GA is inspired in some of the natural evolution mechanisms such as crossover, mutation and natural selection and is useful to solve combinatorial optimization and machine learning problems. GA provides an efficient search method and has been used in many practical instances of optimization and decision-making problems [2].

An important issue when using GA in many applications concerns the genetic drift. This means that the search may stick in local optima without any further progress towards the optimal solution. This happens because GA considers individuals of a population that usually sample only a part, instead of the whole search space. Therefore, it is desirable to maintain population size as large as possible to avoid such problem. Another important issue in this vein concerns population diversity. Several schemes have

been developed to maintain diversity, including migrations, local selection, minimal generation gap, and local search [8].

In general, the use of GA in large scale and complex applications requires high computational effort to evaluate individuals and this makes it difficult to maintain large populations. Numerous techniques have been suggested to estimate fitness of individuals instead of evaluating them directly [3][4][5][7]. One possibility is to assume that individuals are somehow genetically related with each other. In this case, large population size can be handled by clustering the population into groups of similar individuals [5].

Clustering techniques are widely applied in many real world problems such as image processing, pattern recognition, classifiers, machine learning. One important cluster technique is fuzzy c -means [1], a technique that recognizes the fact that clustering is in general imprecise and that an object may be compatible, with different clusters, with different degrees.

The hybrid genetic algorithm (HGA) addressed in this paper uses the fuzzy c -means clustering technique during the fitness evaluation phase to reduce direct evaluations. The idea is to improve the processing speed of the evolutionary process, but maintaining a satisfactory level of population diversity and solution quality, that is, to increase chances to obtain as good solutions as conventional GA. Previous experiments in actual practical circumstances, namely train scheduling and dispatch in freight railroads, have shown the usefulness of this approach as a strategy to solve complex, large scale problems [6]. In this paper we focus on the quality of solutions obtained when population clustering is adopted. For this purpose, we explore a

set of classical optimization problems suggested in the literature.

The paper is organized as follows. Next section details the HGA model. Section 3 summarizes the experimental results and analyzes the solutions quality provided by the HGA model. Section 4 concludes the work and summarizes perspectives for further research.

2 Hybrid Genetic Algorithm

Fig. 1 summarizes the HGA steps. The basic idea is to perform evaluation using a two-step procedure. The first step arranges all individuals of the population into groups using the fuzzy c-means clustering technique and chooses a representative individual for each cluster. The second step evaluates the representative individual of each group (cluster) directly and the remaining individuals indirectly. Clustering is performed in genotype space. In this paper we keep the basic GA operators, crossover, mutation, and selection the same as in conventional GA.

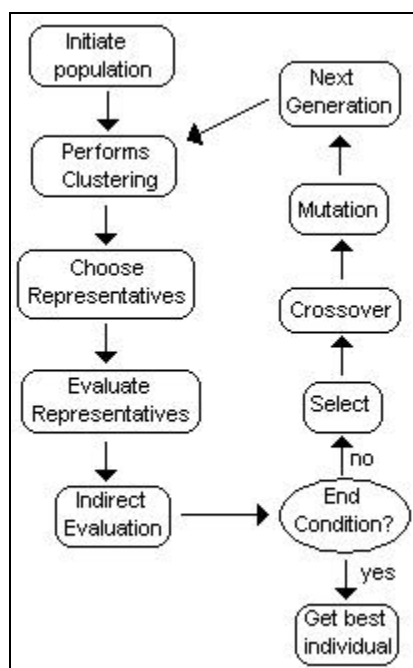


Figure 1: Main HGA steps

As one may realize, the key points in the HGA concern how to choose the representative individual for each cluster and how to do the indirect evaluation. During clustering, the similarity measure adopted to characterize similarity between individuals is an important step and must be well

defined. It should take into account the map behavior between the genotype and phenotype spaces for the specific problem the HGA is handling. For smooth fitness landscapes, where close genotypes are mapped into close phenotypes, Euclidian and/or Hamming distances are appropriate measures. In more complex cases, such as in scheduling problems, where close genotypes do not necessarily mean close phenotypes, the similarity measure is problem dependant and its choice poses considerable challenges.

Another important step during a HGA run concerns the number of clusters adopted in each generation. This is also a problem of major concern and raises complex questions. Computational experiments suggest that the number of clusters in each generation of the evolutionary process is problem dependant. In general, it tends to increase for problems with large number of local optimal solutions, but decreases for problems with one single global optimal solution. For simplicity, in this work we maintain the number of clusters the same over the generations.

In this paper we focus on a combination of basic techniques that we suggest to evaluate the population. An analysis of the relevance of cluster-based genetic algorithms to solve complex, large-scale problems has been reported in [6].

2.1 Choosing Representative Individuals

In the HGA of Fig. 1, we need to choose a representative individual for each cluster after running the clustering algorithm. This is a key point for the HGA performance since all the remaining individuals have their fitness values estimated from the fitness values of the representative individual of each cluster. In this paper, we suggest two basic techniques to choose representatives.

The first, and eventually the most natural choice, is to consider as representative the individual closest to a cluster center. We can, for instance, define the closest individual as the one who has the highest membership value. In this case, the representative individuals can be found using the membership matrix U given by c -means clustering. More precisely, we define

$$I_k = \max_i (u_{ik}), \quad i = 1, \dots, N, \quad k = 1, \dots, c \quad (1)$$

where u_{ik} is the membership degree of the i -th individual in the k -th cluster and N is the population size. Thus, I_k is the representative individual of the k -th cluster.

A second choice would be to consider, as the representative individual, the cluster center itself. In this case we define as representative

$$I_k = v_k, \quad k = 1, \dots, c \quad (2)$$

where v_k is the cluster center of the k -th cluster and c is the number of clusters. This choice means that, if the population size is to remain the same, we must remove c individuals from the population every time representatives are chosen. To keep the best individual found so far in the population, we remove the c worse individuals. This choice, however, may affect the evolution performance by pressuring the selection over the fittest individuals, probably taking the HGA to local optima.

2.2 Indirect Evaluation Methods

The HGA of Fig. 1 needs to evaluate individuals indirectly after representative individuals are chosen. Therefore procedures to estimate the fitness of the remaining population individuals must be given. Here we suggest two basic techniques for indirect evaluation. They are summarized next.

The first procedure estimates the fitness values considering the u_{ik} value, the membership degree of the i -th individual in the k -th cluster, that is, we define

$$fitness(I_i) = \frac{\sum_{k=1}^c u_{ik} * fitness(I_k)}{\sum_{k=1}^c u_{ik}} \quad (3)$$

where $fitness(I_i)$ is the fitness value of the i -th individual whereas $fitness(I_k)$ is the fitness value of the k -th representative individual, and I_k is using (1) or (2) as indicated in the previous section.

The second procedure estimates the fitness values based on the correlation degrees between each representative individual and the remaining individuals. More precisely, let x be the genotype of an individual I_i under evaluation, and let y be the

genotype of the I_k representative individual. Then we compute

$$S_{ik} = \frac{\sum_{j=1}^l x_j y_j - l \bar{x} \bar{y}}{l - 1} \quad (4)$$

$$r_{ik} = \left| \frac{S_{ik}}{S_i S_k} \right| \quad (5)$$

We define

$$fitness(I_i) = \frac{\sum_{k=1}^c r_{ik} * fitness(I_k)}{\sum_{k=1}^c r_{ik}} \quad (6)$$

where l is the length of the genotype. We note that x_j and y_j are the j -th position values of the individual genetic code. In (5) S_i and S_k are the standard deviations of x and y respectively. From (5) we derive the correlation degree r_{ik} between I_i and I_k .

Next section summarizes the experiments conducted to evaluate the techniques suggested in this paper to choose the representative individuals and the method to indirectly evaluate individuals.

3 Experimental Results

Six different instances of genetic algorithms and HGA are considered. They are listed in Table 1.

Table 1: Characteristics of GA and HGA instances

	Number of individuals	Number of clusters	Representative Individual	Indirect Evaluation
GA1	50	---	---	---
GA2	5	---	---	---
HGA1	50	5	Eq. (1)	Eq. (3)
HGA2	50	5	Eq. (1)	Eq. (4)
HGA3	50	5	Eq. (2)	Eq. (3)
HGA4	50	5	Eq. (2)	Eq. (4)

All GA and HGA instances use common representation and operators. More specifically, they use floating-point values in the representation of their genotypes, that is, each chromosome position is a float value. They use arithmetic crossover and Gaussian mutation as operators, and a four-round tournament as selection operator. The crossover rate was kept fixed at 0.6, the mutation rate at 0.04, and the number of generations chosen was 1000.

Analytical functions of the test cases considered in the experiments reported below are easy to compute and offer smooth fitness landscapes. Thus we use HGA as detailed in the previous sections. Complex functions and discrete landscapes will be treated in a future work. An example in this problem instance and preliminary experiments were reported in [6]. Here our aim is to get insights on how the HGA behaves against the classic GA, emphasizing the quality of solutions. Future work shall address the use of alternative fitness evaluation strategies.

3.1 Case 1 – De Jong Function

$$f(x) = \sum_{j=1}^5 x(j)^2, -100 \leq x(j) \leq 100$$

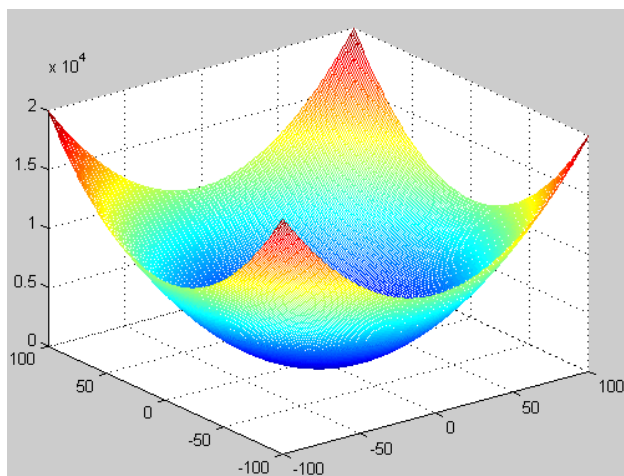


Figure 2: De Jong function

De Jong function is a quadratic function with a single global optimum as shown in Fig. 2. In Fig. 3, we note that GA1 rapidly converges to a value corresponding to 90% of the optimal, but remains stuck at this value even after several direct evaluations. HGA2, HGA3 and HGA4 converge to the optimal solution faster. In this case, all HGA instances (1,2,3 and 4) give better solutions than GA2. HGA2, GA3 and HGA4 achieved even a better solution than GA1 despite evaluating only 5 individuals directly (5 clusters) in each generation.

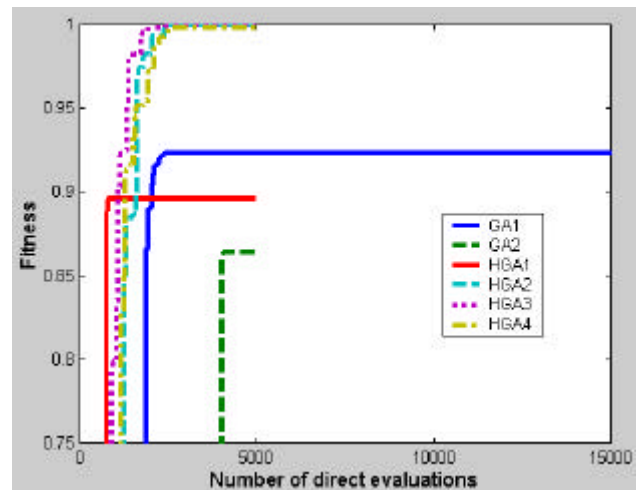


Figure 3: Performances for De Jong function

3.2 Case 2 – Griewangk Function

$$f(x) = 1 + \sum_{j=1}^{10} \frac{x(j)^2}{4000} - \prod_{j=1}^{10} \cos\left(\frac{x(j)}{\sqrt{j}}\right), -500 \leq x(j) \leq 500$$

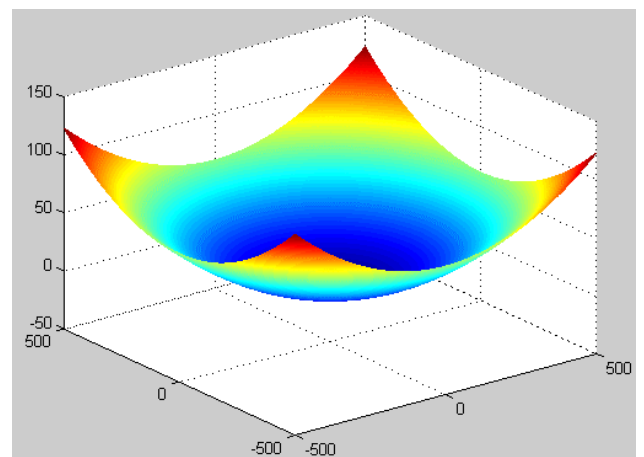


Figure 4: Griewangk function

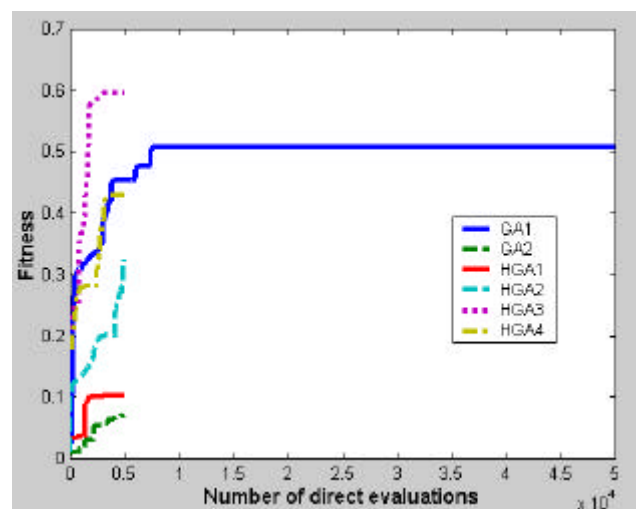


Figure 5: Performances for Griewangk function

Griewangk is also a quadratic function as shown in Fig. 4, but it has a product term in it. This may lead some search methods to undesirable solutions. We note in Fig. 5 that this is the case with GA2, but most of HGA has escaped from the *trap*. Once again GA1 rapidly converges to 50% of the optimal solution, but remains stuck afterwards whereas HGA3 converges to 60% of the optimal solution. HGA3 again presents the best solution amongst all HGA and GA.

3.3 Case 3 – Schwefel Function

$$f(x) = 4189829 * l + \sum_{j=1}^2 x(j) \sin(\sqrt{|x(j)|}), -500 \leq x(j) \leq 500$$

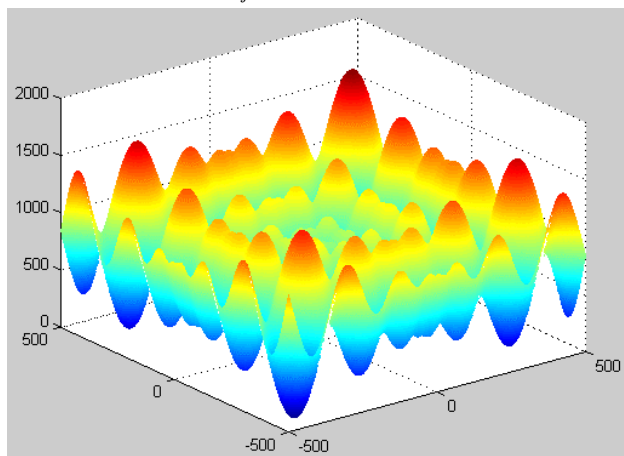


Figure 6: Schwefel function

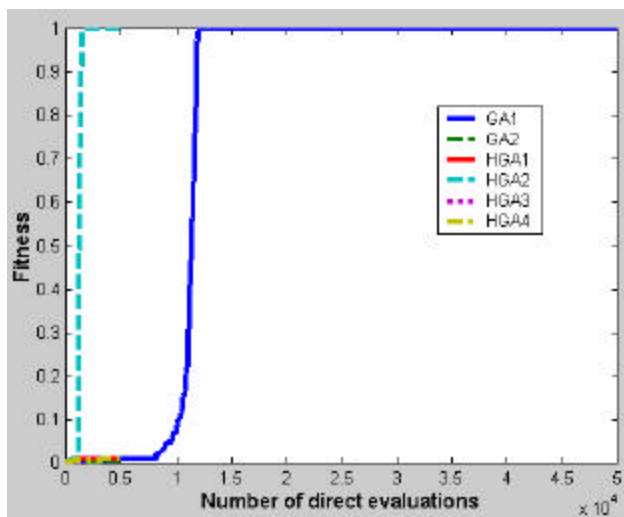


Figure 7: Performances for Schwefel function

Schwefel function has many local optimal solutions, as Fig. 6 shows. We note in Fig. 7 that GA2 and almost all HGA converge to a local optimum. However, HGA2 reaches the global optimal solution evaluating directly far less individuals than GA1 did.

This is an instance that highlights the key idea behind HGA. Another point to note is, since only one HGA converges to the optimal solution, it seems advisable to consider different combination of representative individuals and indirect evaluation methods when running HGA algorithms.

3.4 Case 4 – Rastrigin Function

$$f(x) = 3 * l + \sum_{j=1}^5 x(j)^2 - 3 \cos(2\pi x(j)), -10 \leq x(j) \leq 10$$

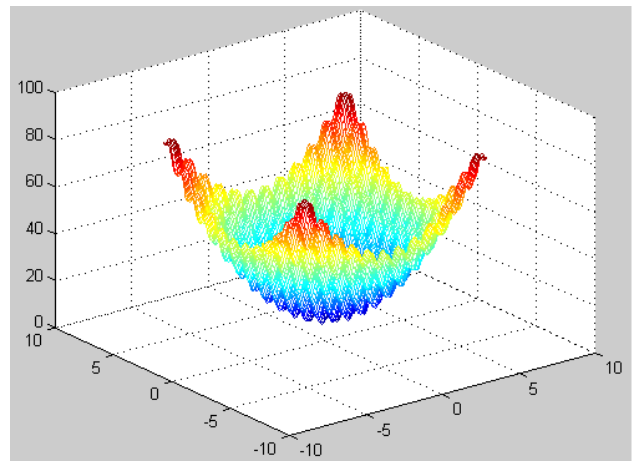


Figure 8: Rastrigin function

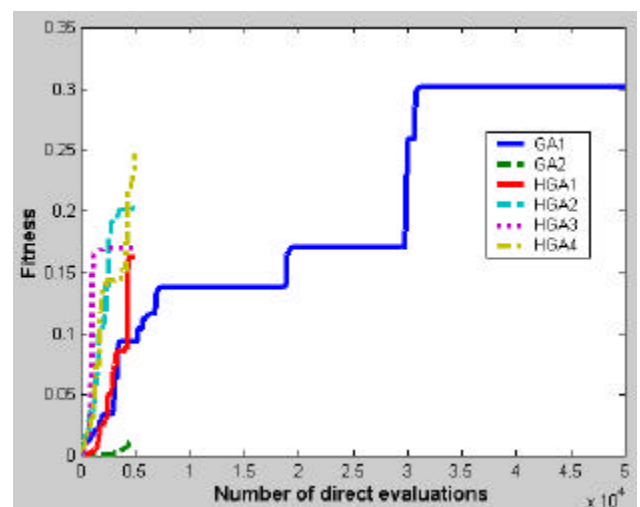


Figure 9: Performances for Rastrigin function

Similarly as the Schwefel function, Rastrigin function, depicted in Fig. 8, also has many local optimal solutions. As Fig. 9 shows, GA1 slowly converges to a local solution, 30% of the optimum. HGA avoid local solutions, as HGA4 shows. In the case of HGA4, it reaches 15% of optimal solution for a considerable number of direct evaluations, and next it proceeds to reach about 25% of the optimum.

From the processing speed point of view, GA2 is obviously the fastest instance, but always gives the poorest solution amongst all GA and HGA, in all test cases. GA1 runs faster than all HGA and provides solutions as good as HGA, eventually performing better than HGA. From the computational performance point of view, this is expected since, for all test cases considered in this paper, the cost to evaluate one individual directly is low when compared with the cost to perform population clustering and indirect evaluation. The HGA performs better when the cost to evaluate one individual directly is relatively high, which is not the case in the test cases considered here. Recall, however, that our main purpose here is to evaluate the quality of the solutions. In this case, all HGA performed as well as GA1 in all test cases, but with far less direct evaluations.

The experiments reported here, however, still do not give us an answer on how to select the most appropriate schemes to choose representative individuals and indirect evaluation procedures. It seems that combinations of methods and procedures are a promising scheme. From Fig. 7 we note that only HGA2 performs well in case 3, but failed to be the best in the remaining cases. This suggests a need to develop methods that performs consistently well. It is worth to note that, overall, the HGA suggested in this paper has efficiently reduced the number of direct evaluations without sacrificing solution quality.

4 Conclusion

The HGA addressed in this paper reduces considerably the number of direct evaluations of individuals using population clustering. Only one individual for each cluster is evaluated and the remaining individuals have their fitness estimated indirectly. Two methods to choose representative individuals and to perform indirect evaluation were reported. Experimental results show that HGA has similar performance, but with fewer direct evaluations than conventional GA. This is a key for problems with high costs to evaluate individuals.

Many important issues still remain to be investigated. There is a need to consider other methods to choose representative individuals and other indirect evaluation procedures. It is critical to investigate the evolution of clusters themselves and the performance of HGA in discrete search spaces.

The use of fuzzy rules to control parameter values, such as crossover and mutation rate, population size, number of clusters, could be also investigated using rule-based genetic fuzzy systems.

Acknowledgments

The first author acknowledges CAPES, Brazilian Ministry of Education, for its support. The second author thanks CNPq, the Brazilian National Research Council, grant #304299/2003-0, and FAPESP, the Research Foundation of the State of São Paulo, grant #03/10019-9. The authors are also grateful to the anonymous referees for their many helpful and enlightening comments.

References

- [1] J. C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*, Plenum Press, 1987.
- [2] D. E. Goldberg, *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley Publishing Co.Inc., 1989.
- [3] Y. Hanaki, T. Hashiyama, S. Okuma, "Accelerated evolutionary computation using fitness estimation", *IEEE Trans. SMC*, vol. 1, pp. 643-648, 1999.
- [4] K. Kado, P. Ross, D. Corne, "A study of genetic algorithm hybrids for facility layout problems", *Proc. of ICGA*, pp. 498-505, 1995.
- [5] H. Kim, S. Cho, "An efficient genetic algorithm with less fitness evaluation by clustering", *IEEE Publication 0-7803-6657-3*, 2001.
- [6] F. Mota Filho, R. Gonçalves, F. Gomide, "Genetic algorithms, fuzzy clustering and discrete event systems: An application in scheduling", *Proc. of 1st Workshop on Genetic Fuzzy Systems*, Granada, Spain, pp. 83-88, 2005.
- [7] M. Salami, T. Hendtlass, "A fitness estimation strategy for genetic algorithms", *Lecture Notes in Computer Science*, vol. 2358, pp. 502-513, 2002.
- [8] T. Unemi, "A design of multi-field user interface for simulated breeding", *Proc. of the 3rd AFSS*, pp. 489-494, 1998.