International Journal of Electronic and Electrical Engineering.
ISSN 0974-2174, Volume 7, Number 1 (2014), pp. 43-50
© International Research Publication House
http://www.irphouse.com

Hybridization of Heuristic Genetic Algorithm: A Survey

Chaitanya Kamboj[1] Mandeep Singh[2] and Deepanshu Rana[3]

[1]3rdYear Student, EECE Department, ITM University, Guragon, Haryana
[2]EECE Department, ITM University, Guragon, Haryana
[1]3rdYear student, EECE Department, ITM University, Guragon, Haryana

Abstract

Genetic algorithm (GA) has been the topic of research in recent years. A genetic algorithm is capable to blend different approaches to form a hybrid to emphasize on the techniques to produce best solutions. In this paper we have discussed the basic design of global GA and reviewed different methods used to design hybrid GA's. Also, it covers the issues that need to be considered at the time of designing GA incorporating various other search techniques.

Keywords: Genetic algorithm (GA), Hybrid GA, Adaptive GA, Real coded GA (RCGA), Saw tooth GA.

1. Introduction

One of the first algorithms to be developed for solving complex problems having discontinuous and non convex design space was GA. GA are usually modified to fit the problem statement, which has led to gradual development of various variants of GA. GA is inspired by the biological principle of *"survival of fittest"*. Genetic algorithm is a stochastic search algorithm which provides a global maxima or minima in a confined search space. It is successful due to its simple design i.e. it does not require information about the derivative of a function or differential equation of the function. It directly operates on the function. But it also faces difficulty in optimizing certain complex problems and solving multi peak functions. To overcome these problems many researchers around the globe have developed different variants of GA. In this paper we are going to present the basic idea of genetic algorithm and discuss some of its variants.

2. Methodologies

The methodology to proceed to an optimal solution is based on heuristic search in a given population and the direction of search is decided by the different operators used. A detailed discussion of this is presented in this section.

2.1 Initialization

The initial population is generated through a random process in all the Evolutionary algorithms. It is done by creating a population vector of size AB given by [1] comprising of individuals over G generation. As shown in [2] each individual (G) X is a vector that has elements according to the decision variable (V). The user controls the population size AB.

$$P^G = \{X_1^G, X_2^G, X_3^G, \ldots \ldots X_{AB}^G\} \qquad [2.1]$$

$$X_J^G = \{X_{1J}^G, X_{2J}^G, X_{3J}^G, \ldots \ldots X_{YJ}^G\} \qquad [2.2]$$

The initial population is created by assigning some values to decision variables according to the problem. It is randomly chosen to enclose the entire search space.

2.2 Genetic operators

(a) Fitness function: The fitness function is used to calculate the role of each chromosome present by assigning appropriate probability to each chromosome. The chromosomes with better fitness have high fitness value whereas chromosomes with lower fitness have less fitness value , they are ranked from best to worst fitness value

(b) Selection operator: The better chromosomes with above average fitness value are selected for the crossover. Multiple copies of the chromosomes are injected into the selection pool. A probability factor is multiplied to each chromosome depending on its fitness value. There are basically two methods of selection:

(i) Roulette wheel selection
(ii) Tournament selection

(i) Roulette wheel selection: In this technique after the calculation of fitness value each chromosome in normalized (i.e. if $H(i)$ represents the fitness value of the chromosome then its probability ($P(i)$) to be injected into the selection pool is given by

$$P(i) = \frac{H(i)}{\sum_{j=1}^{n} H(j)} \qquad [2.3]$$

where 'n' is number of individuals in population. After normalization the chromosome with the best fitness value will have the maximum chance of being selected if roulette wheel is spun 'n 'times. Where n is number of chromosomes. In this technique only one chromosome is selected each time wheel is spun.

(ii) Tournament selection: If the above process is repeated with multiple numbers of selections in a single spin then there will be multiple numbers of chromosomes selected which we can call above average chromosome out which the best is then selected. In this type of selection multiple "tournaments "go on at single time hence it is an efficient way to create the selection pool.

(c) Crossover: It is the most important operator in the algorithm. In this stage two random chromosomes are selected for crossover to create two chromosomes with better fitness. The new chromosome is created by exchanging information between the two chromosomes. This can be explained as follows;

Parent chromosome:

$$X(1) = \{\mathbf{1100}1011\}$$

$$X(4) = \{1101\mathbf{1111}\}$$

After one point crossover

Offspring:

$$X(5) = \{\mathbf{11001111}\}$$

Not all the chromosomes are used for crossover, chromosomes with better fitness are preserved and the rest are assigned a crossover probability to be selected for crossover. There are three types of crossover: uniform crossover, one point crossover and two point crossover.

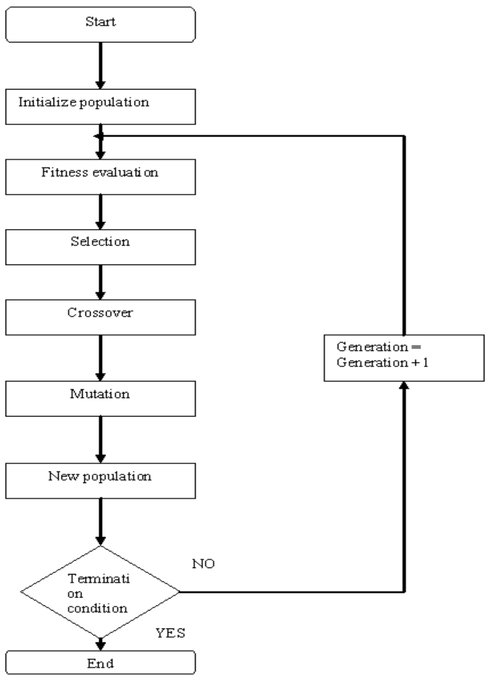

Fig. 2.1: Flowchart of GA.

(d) Mutation: The mutation operation is implemented on the new generation of chromosomes created after crossover. A new chromosome is selected through mutation probability, then the chromosome is mutated to cover the entire region in the proximity to that chromosome so that the optimal solution is not locally converged, hence injecting diversity into the search space. In one point mutation the data at a particular point is mutated (in binary changed from 0 to 1 or vice versa). Mutation changes the chromosomes locally only.

(e) Elite operator: This operator makes sure that the chromosome with the best fitness value is selected in the next generation bypassing crossover and mutation. It makes sure that the fitness of best chromosome in the next generation will not decrease, hence improve the quality of solution.

3. Variants of GA

Due to variations in the problem statement many variation in the parameters of GA are possible which has led to emergence of many variants of GA. For e.g. hybrid GA, Real Coded GA, Binary Coded GA, Saw tooth GA, Adaptive GA, Differential evolution (inspired by GA) etc.

(a) Hybrid GA: The main errors that a traditional GA suffers from are slow convergence rate and pre mature convergence. To improve upon these defects a new GA known as hybrid GA was developed. It is a combination of simple GA and local search algorithm. The purpose is to distribute the optimization task into two parts, the GA first performs the search and then the refinement is done by the local search algorithm. Both the algorithm run in parallel, say after 'n' iteration of local search the local optimal solution is injected into the current generation. A local search method locates the local minima which complements the GA to capture global minima. A variety of techniques have been crossbred to encapsulate the best of both the techniques in real world applications [4, 5]. Population size is also a imperative element in a genetic algorithm. A model was introduced [6] in which size of the population banks on standard deviation of population and the signal difference between best and second best chromosome. In this model if a local search method is implemented in such a way that it reduces the standard deviation and increase the signal difference the concluding hybrid could be very efficient even in small population size. Espinoza [7] demonstrated the effect of a local search method in reduction of population size. Generally, the mutation and the crossover operators produce infeasible solutions for a highly constrained problem.

To avoid generation of infeasible solutions many techniques have been proposed like partial matched crossover (PMX) [8] for use in order- based problems. To solve the highly constrained timetabling problem[9] a heuristic crossover operator was introduced with direct representation of the timetable so that fundamental constraints are never violated for solving traveling salesman problem modified crossover (MOX)[10], order crossover(OX) [11] .

In problem model-building genetic algorithm (PMBGA), a probabilistic model is used to replace the crossover and mutation operator to learn the structure of a problem on the fly. It is done to improve on growth and mixing of building blocks. New potential solutions are obtained by sampling the model. Bi-variate distribution algorithm (BMDA), compact genetic algorithm (COA) is some examples of PMEGA. The adjustment of the control parameters is also a necessary task to improve the performance of the algorithm.

(b) Adaptive GA: if the parameters are varied in accordance to the required condition throughout the optimization process, it gives rise to another form of GA known as adaptive GA. In this method the crossover and mutation operator are multiplied by a probability factor known as crossover probability and mutation probability respectively and these probability factors are varied in accordance to the fitness value of the chromosomes. The mutation and crossover steps are the most important steps that ensure diversity and improve convergence of the algorithm. An adaptive GA is inspired by the human reproduction was proposed in [2] and the constraints like consanguinity, reproduction age, same sex reproduction etc have been carefully addressed. The methodology used is that the each individual is assigned a binary code; the code has two parts which represent the individual sex and individual exhibition model. The fitness value of the individual's is calculated. The selection process make sure that there are nearly equal number of male and female selection pool and after that they judged on the criteria of their age . The individuals with higher fitness values mate with each other, this speed up the algorithm toward global convergence.

(c) Real coded GA: In conventional GA the parameter are encoded in binary digits. But in recent years there has been an emergence of real coded optimization parameters which has led to better solutions than its binary counterpart. RCGA is most efficient if the search space is continuous and very precise solution is required. In RCGA the length of the chromosomes is kept equal to the length of the optimized solution due to which this algorithm can handle large search space without giving up on precision. The crossover operator is considered the fundamental search operator in RCGA therefore a large number of crossover operators have been developed for e.g. Blend crossover, heuristic crossover, arithmetic crossover, unimodal normal distribution crossover, simplex crossover, Laplace crossover etc. The RCGA has 5 operators which work on the initial population of chromosomes to generate an optimal solution, and are named as scaling operator, selection operator, crossover operator, mutation operator and elite operator. The initial population function is scaled to form a fitness function, this is done to prevent pre mature convergence in early stages and increase the rate of convergence in later stage.

(d) Binary coded GA: The binary coded GA is heuristic search algorithm which overhaul's the initial population of chromosomes using the genetic operators into a new generation with better fitness values. The chromosomes are assigned certain fitness value; the selection process eliminates all the individuals with low fitness value. Then the crossover is performed followed with mutation which gives rise to a more fit

generation. The genetic operators perform their function as in a conventional GA. The best chromosome of the new generation is replaced by the best chromosome of the previous generation by the elite operator if the fitness of the best chromosome of the new generation is less than that of the worst chromosome of the previous generation. The bit representation is used to assign binary values to all the chromosomes. (encoded in the form of 0 or 1) it has a lot of advantage as it is easy to encode any kind of information, the natural selection and crossover operator can be applied without any complexity but they have certain disadvantages when the search space is continuous in nature. In such case they are not able to cover the entire search space and hence the solution cannot be considered as a global optimal solution.

(e) Saw tooth GA: This is another technique to improve the robustness and efficiency of GA. In simple GA a constant population of chromosomes is considered at the time of initialization which guides the algorithm to an optimal solution after performing the series of operations. If a large population is initialized the algorithm slows down considerably and if the population is smaller in size then their risk of pre mature convergence. So, to address these problems saw tooth GA was proposed .The population is reinitialized periodically with constant amplitude and period. (Saw tooth). The selection operator is affected by the variation in population size. The probability of selection of an individual chromosome remains constant throughout. The population size decreases linearly during the current period and randomly generated chromosomes are introduced at the beginning of the next period.

4. Conclusion

In this paper we have tried to highlight the importance of hybridized genetic algorithm for solving different problem with large constraints by reviewing currently used methods. These techniques show that hybridizing is an efficient way to solve hard problems. Hybridization can be achieved by combining GA with local search methods making the use of domain specific knowledge [4] to enhance the speed of convergence of GA and they can be made competitive with others when the search space is too large to explore. In adaptive GA the crossover and mutation operator are multiplied with probability factor and the probability factors depend on the clustering analysis of the optimization state of the chromosomes. RCGA has evolved in recent time and is most efficient if the search space is continuous and very precise solution is required. Various hybridizing techniques have been used for selection of a local search method, the selection of individuals and other design aspects.

References

[1] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby, *"Hybrid genetic algorithms: A review"*, Engineering Letters, vol. 3, no. 2, pp. 124-137, 2006.

[2] Tai- Shan *"An Improved Genetic Algorithm and its Blending Application with Artificial Neural Network"*, 2nd International Workshop on Intelligent Systems and Applications, Wuhan, China, 22-23 May 2010, pp. 1 - 4.

[3] Pengfei Guo, Xuezhi Wang and Yingshi Han, *“The enhanced genetic algorithms for the optimization design”*, Biomedical Engineering and 647 Informatics (BMEI), 2010 3rd International Conference on, vol.7, no., pp.2990-2994, 15-18 Oct. 2010.

[4] P. Preux and E.-G. Talbi, *“Towards hybrid evolutionary algorithms," International Transactions in Operational Research”*, vol. 6, pp. 557-570, 1999.

[5] T. Yamada and C. Reeves, *“Solving the Csum permutation flow shop scheduling problem by genetic local search”*, in International Conference on Evolutionary Computation. Anchorage, USA, 1998, pp.230-234.

[6] G. Harik, E. Cantu-Paz, D. E. Goldberg, and B. l. Miller, *“The gambler's ruin problem, genetic algorithms, and the sizing of populations”*, Evolutionary Computation, vol. 7, pp. 231 - 253, 1999.

[7] F. B. Espinoza, B. Minsker, and D. Goldberg, *“Performance evaluation and population size reduction for self adaptive hybrid genetic algorithm(SAHGA)”*, in the Genetic and Evolutionary Computation Conference, vol. 2723, Lecture Notes in Computer Science San Francisco, USA: Springer, 2003, pp. 922-933.

[8] H. Asoh and H. Mühlenbein, *“On the mean convergence time of evolutionary algorithms without selection and mutation”*, in Parallel Problem Solving from Nature, PPSN III, Y. Davidor, H.-P. Schwefel, and R. Manner, Eds. Berlin, Germany: Springer-Verlag, 1994, pp. 88–97.

[9] D. E. Goldberg and R. Lingle, *“Alleles, loci, and the traveling salesman problem”*, in the International Conference on Genetic Algorithms and their Applications. Hillsdale, USA: Lawrence Erlbaum, pp. 154-159, 1985.

[10] J. Wroblewski *“Theoretical foundations of order-based genetic algorithms”*, Fundamental Informaticae, Volume 28, Number 3-4, pp. 423–430, 1996.

[11] M. Oliver, D. J. Smith, and J. R. C. Holland, *“A study of permutation crossover operators on the traveling salesman problem”*, in the Second International Conference on Genetic Algorithms on Genetic algorithms and their application. Hillsdale, USA, 1987, pp. 224–230.