

ADAPTIVE GENETIC ALGORITHMS APPLIED TO DYNAMIC MULTI-OBJECTIVE PROBLEMS

ZAFER BINGUL, ALI SAFAK SEKMEN and SALEH ZEIN-SABATTO

Department of Electrical and Computer Engineering
Tennessee State University
Nashville, Tennessee

ABSTRACT

This paper describes an application of adaptive genetic algorithms in which multi-objectives such as minimizing the territory losses and maximizing enemy aircraft losses are achieved by finding the optimum war allocation scenario simulated by the THUNDER software. A genetic algorithm with two fuzzy logic based mechanisms is used. In the first mechanism, the mutation and crossover rates are changed adaptively to provide a fast and smooth convergence to the optimum possible solutions. In the second mechanism, the multi-objective fitness function coefficients are changed dynamically in each run. Comparisons of the results of the best fitness values, the adaptive genetic algorithms with dynamic fitness function improve the convergence rates considerably and maintain smoother convergence to the best possible solutions than that of the conventional genetic algorithms.

INTRODUCTION

It is clear that there are many optimization problems in which optimizing several measures that may conflict with each other is unavoidable. The most important aspect in solving this class of problems is to determine suitable design objectives to evaluate the “goodness” of a certain solution. Generally, multi-objective optimization problems are either represent a minimization or a maximization of all the objectives. However, it is sometimes necessary to minimize some objectives and maximize the others.

Evolutionary algorithms (EA) are useful tools to explore big search spaces in reasonable time (Schaffer, 1985). They could provide means for addressing complex engineering problems such as rapid analysis of battlefield tactics and supply-chain management. These complex problems are usually involved with chaotic disturbances, randomness, and complex nonlinear dynamics. It is not possible to solve these complex problems with use of traditional algorithms. Evolutionary algorithms are capable of processing sets of solutions in parallel and they exploit the similarities of the solutions by recombination method. Genetic algorithms (GAs) are subclass of the evolutionary algorithms and are based on the principles of natural selection (Zitzler and Thiele, 1999). The major benefits of the GAs is that they provide a robust search in complex spaces and are usually less expensive, in terms of computation, compared to most other optimization solutions (Schaffer, 1985). They are also resistant to getting trapped in local optima. This leads to a wide range of applications in the large-scale optimization problems of various fields. Fuzzy logic techniques depend on fuzzy models rather than mathematical models. The major advantages of these techniques are that a fuzzy objective function is not differentiable by its nature and the fuzzy set theory offers a very attractive way to extract information from data having an amount of ambiguity or uncertainty. Some complex systems may be considered as fuzzy systems whose implementations can be formulated as a search problem in high-dimensional space. This

search space is digitized by a rule set, membership functions, and the corresponding system's behavior. According to some given performance criteria, a hyper-surface in the search space can be created. This hyperspace may be infinitely large, non-differentiable, noisy, multi-modal, and deceptive. Fuzzy logic techniques are used for optimization problems and are defined as the processes of finding the optimal location of the hyperspace.

Definitions

A multi-objective optimization problem with inequality constraints can be represented mathematically as a vector function f , which maps a set of m parameters (decision variables) to a set of n objectives. This can be defined as follows.

$$Y = f(x) = (f_1(x), f_2(x), f_3(x), \dots, f_n(x)) \quad (1)$$

subject to the constraint: $x = (x_1, x_2, \dots, x_m) \in X$ and $y = (y_1, y_2, \dots, y_n) \in Y$

where x is the set of decision vectors and X is the parameter space, while y is the objective vector, and Y is the objective space. The set of solutions of a multi-objective optimization problem consists of all decision parameters for which corresponding objective vectors cannot be improved in any dimension without degrading another. Such solutions are termed as Pareto-optimal solutions (Zitzler and Thiele, 1999). A Pareto optimal solution is not unique, but it is a member of a set of such points, which are considered equally good in terms of the vector objective.

Heuristic search in most of game programs uses the following static fitness function to lead the direction of the search (Goldberg, 1989):

$$F = \sum_{i=1}^n w_i f_i \quad (2)$$

where f_i is the features and w_i is the corresponding weights which indicate the importance of the features. Determining these features and weights are very important for effective search in multi-objective optimization problems. It is difficult to choose the weights in dynamic multicriteria environment and the weights are often assigned through trial-and-error process that may be very time consuming.

Multi-Objective Genetic Algorithms

Vector Evaluated Genetic Algorithm (VEGA) was first developed by Schaffer (1985). It is different from simple genetic algorithms since it uses a modified selection operator so that in every generation a number of subpopulations are generated by performing proportional selection according to each objective function in turn. The main drawback of this technique is that it is incapable of producing Pareto-optimal solutions in the presence of non-convex search spaces. In lexicographic ordering technique (Fourman, 1985), the designer ranks based on importance of the objectives. The optimum solution is then obtained by minimizing the objective functions, according to their importance, starting from the most important one. The main weakness of this method is that it tends to favor certain objectives and this may lead to undesirable results when there exist many objectives and randomness involved in the process. Hajela's and Lin's genetic algorithm (HLGA) (Hajela and Lin, 1992) uses the weighted-sum method for fitness assignments. The weighting coefficients in each objective are included in the chromosome. The diversity of the weight combinations is promoted by phenotypic fitness sharing and GA evolves solutions and weight combinations simultaneously. There are certain advantages

of this technique: it is computationally efficient, and it generates strongly non-dominated solutions. However, determining appropriate weights in this method is not easy for black-box optimization that is considered as the presence of little or no knowledge about the existing domain. The non-dominated Sorting Genetic Algorithm (NSGA) (Sirinivas and Deb, 1993) is based on several layers of classifications of the individuals. Before selection is performed, the population is ranked on the basis of non-domination: all non-dominated individuals are classified into one category (with a dummy fitness value, which is proportional to the population size, to provide an equal reproductive potential for these individuals). To maintain the diversity of the population, these classified individuals are shared with their dummy fitness values. Then this group of classified individuals is ignored and another layer of non-dominated individuals is considered. The process continues until all individuals in the population are classified. Zitzler et al. (1999) developed Strength Pareto EA (SPEA). In this algorithm, several desirable properties of earlier multi-objective EA such as continuous updated population, preserving population diversity and evaluating an individual's fitness is came together.

PROBLEM STATEMENT AND DESCRIPTION

THUNDER software is a very large campaign simulation model and it was built based on Monte-Carlo simulation. This software is a stochastic, two-sided, analytical simulation of military operations developed by System Simulation Solutions Inc. (S3I) for the Air Force Studies and Analyses Agency (AFSAA). This simulation was designed in order to examine issues involving the utility and effectiveness of air and ground power in a theater-level joint warfare context. The Thunder software can define approximately 25 air missions grouped under air-to-ground missions, air-to-air missions, air defense suppression missions, reconnaissance, anti-tactical ballistic missile, and air refueling. This software automatically plans military moves and actions in a rule-based manner. It also estimates the outcomes of these actions and then it dynamically adjusts the on-going military operations.

The main purpose of this research is to determine how to effectively allocate force power using adaptive genetic algorithms with dynamic fitness functions. The war allocations are made based on the capabilities of threat forces, conditions of the war, and the capabilities of friendly forces, all simulated by THUNDER software. A war allocation input file is generally given by the user. It does not always produce consistent results since software experts do not understand unknown dynamics and randomness in the simulation. This causes lack of understanding the relationship between inputs and outputs of THUNDER. In this study, only four missions and 15-day war were used as input. The missions were Offensive Counter Air (OCA), Strategic Target Interdiction (STI), long range air INTerdiction (INT), and lethal direct air Defense Suppression of the Enemy Air Defense (DSEAD). OCA, STI and INT are air-to-ground missions. OCA is against airbases and INT is against units moving on the network and in garrison, logistics facilities, transportation network transshipment points, checkpoints, supply convoys, and air defense complexes. STI is against strategic targets. DSEAD is a suppression of enemy air defense missions and it is against air defense sites. The THUNDER software can be viewed more like a two-player game in which blue represents the friendly side and red is the enemy side. Our objectives for these scenarios would be to:

1. Minimize the territory that blue side losses
2. Minimize the blue side aircraft lost
3. Maximize the number of red side strategic targets killed
4. Maximize the number of red side armor killed

This is a typical multi-objective optimization problem because it is desired to optimize all of the four objectives simultaneously.

DYNAMIC MULTI-OBJECTIVE OPTIMIZATION APPROACH

Transforming the above multi-objective optimization problem to a form suitable for direct implementation of GA was achieved using two fuzzy logic systems. The procedure followed to solve the allocation problem using GA and THUNDER Software is illustrated in Figure 1. Outputs from THUNDER Software (territory lost, aircraft lost, the number of strategic targets killed and the number of red armor killed) were assigned a minimum score and a maximum score. They were translated to a minimum score of 1 and a maximum score of 2. Scores between the minimum and maximum were interpolated based on the worst case and the best case (determined by expert knowledge).

In general, it is important to improve only the high priority objectives, such as hard constraints, until the corresponding design goals are met. After that improvements may be sought in the lower priority objectives. However, in our case, all objectives must be achieved dynamically in each war scenario. It is known that different battlefield tactics ought to be employed in different war stages. These tactics can only be described linguistically. Thus, rule based fitness function was used to calculate the fitness value based on the above objectives. In this method, individual score of each objective is squared and weighted. The assigned fitness value was defined as the sum of these terms:

$$F(t) = \sum_{i=1}^4 \mu_i(t) w_i f_i^2 \quad (2)$$

where μ_i and w_i are membership function and weight characterizing feature f_i .

Genetic algorithms work very efficiently when appropriate value for parameters such as mutation rate, crossover rate and population size are chosen. In most of conventional GAs, the probabilities of crossover and mutation are usually held constant for the entire run. The problem with selecting probability values for the parameters is that they vary in a wide range and do not yield the best GA performance. Hence, it is apparent that these parameters must be varied based on fitness score of each individual in the population during run. This leads to smooth convergence of the fitness function and less number of generations.

In this work, two of these three parameters (mutation and crossover rates) were adjusted as GA runs. The population size (100) was kept constant. In order to change these parameter adaptively, a three-inputs two-outputs fuzzy logic system (FLS) was used. The inputs to the FLS are the best fitness value, the average and the variance of the fitness values of the population. Three ranges (LEFT-small, TR-medium, RIGHT- large) and triangular membership functions were used to fuzzify the inputs. A set of rules that describes how to adjust the mutation and crossover rates was constructed.

To extract meaningful information from the war results, a second fuzzy logic system was employed as a general framework to determine the fitness function weights. Using Equation 2, the fuzzy GA tries to improve the worst scores by assigning to it a high weight value, assign an intermediate weight value to the less bad scores, and assign a low weight value to the high scores. By doing so, the fuzzy GA is forced to push up the lower scores and to push down the higher scores by adjusting the weights. This makes priority adjustment in the objectives.

RESULTS AND DISCUSSION

In this research, the GA operators and the fitness function are adaptively changed on-line. These adaptations allow to concurrently evolve non-dominated solutions and guide the search in several directions on the multi-dimensional search space. Figure 2 illustrates variations of the mutation and crossover rates over 50 generations. It is seen from the figure that both rates are changed between 0.018 and 0.047, and 0.65 and 0.75 respectively. These changes allow to explore new solution points and to optimize run time of the GAs. By using automatic settings of these rates, the search did not trap at a local area especially during the start up time.

During simulation time, interactions between four objectives occurred in interrelated manner and correct setting of importance for each objective was made dynamically. The fuzzy based GA develops an optimum average fitness value of about 1.5 for each objective during the run. Figure 3 displays a comparison of the performances of the static and dynamic GA systems. The stochastic nature of a GA optimization requires multiple runs to ensure reproducibility. Thus, the GA was run four times and each time similar results were obtained. It is observed that the dynamic GA system has higher fitness values and faster convergence compared to the static system.

CONCLUSIONS

In this research we focused on the problem of how to effectively allocate force power using adaptive genetic algorithms. In order to achieve optimum solution of four objectives, two fuzzy logic systems were employed. By judging the overall fitness of the solutions, the rule based GA produce superior performance for all of the objectives. This paper concludes that the adaptive genetic algorithms with dynamic fitness function improve the convergence rates considerably and maintain smoother convergence to the best possible solutions than that of the conventional genetic algorithms.

ACKNOWLEDGEMENT

This research was partially supported by the Boeing Company and ONR. We would like to acknowledge the two agencies for their technical and financial supports.

REFERENCES

- Schaffer, J. D., 1985, "Multiple objective optimization with vector evaluated genetic algorithms," Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms, pp. 93-100.
- Fourman, M. P., 1985, "Compaction of symbolic layout using genetic algorithms," *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pp. 141-153
- Hajela, P. and Lin, C. Y., 1992, "Genetic search strategies in multicriterion optimal design", *Structural Optimization*, Vol. 4, pp. 99-107.
- Srinivas, N and Deb, K, 1993, "Multiobjective optimization using nondominated sorting in genetic algorithms," Technical report, Department of Mechanical Engineering, Indian Institute of Technology, Kanpur, India.
- Zitzler, E and Thiele, L., 1999, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Trans. on Evolutionary Computation*, Vol. 3, pp. 257-271.
- Bingul, Z., Sekmen, A. S., Palaniappan S., and Zein-Sabatto, S., 1999, "An application of multi-dimensional Optimization Problems Using Genetic Algorithms," Proceedings of the IASTED International Conference Intelligent systems and control, Santa Barbara, CA, October 28-30.
- Goldberg, D. E., 1989, "Genetic algorithms in Search, optimization and machine learning," Addison-Wesley Publishing Company.
- Reardon, B J., 1998, "Fuzzy logic versus Niched Pareto multiobjective genetic algorithm optimization", *Modeling Simul. Mater. Sci. Eng.*, Vol. 6, pp.717-735.

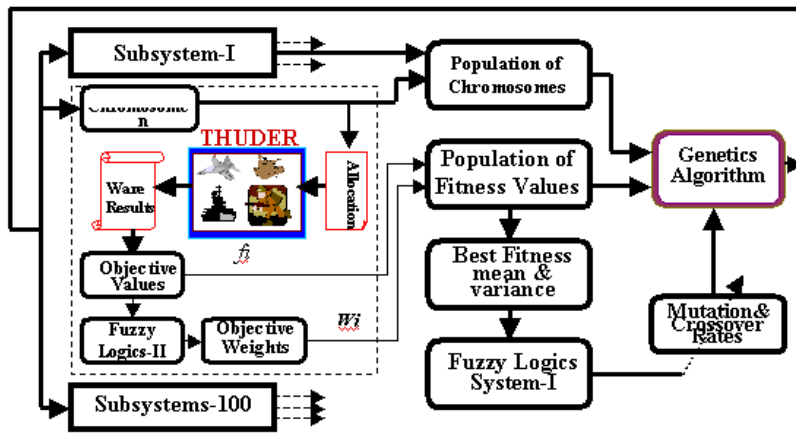


Figure 1. Schematic representation of optimization procedure.

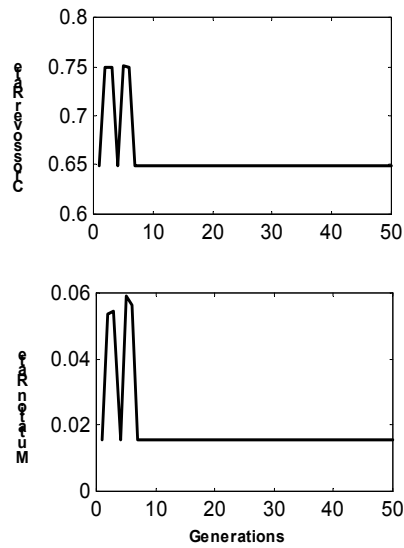


Figure 2. Adaptation of Mutation and Crossover Rates

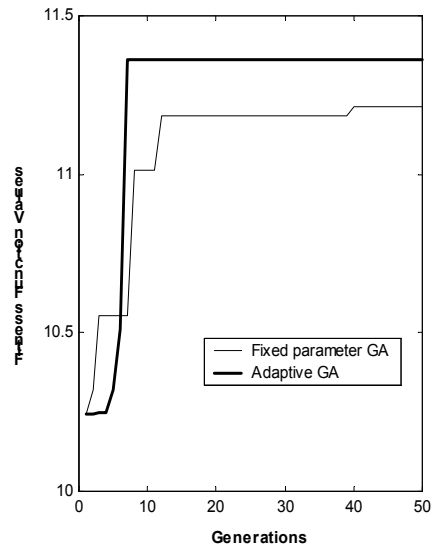


Figure 3. Performance Comparison of Static and Dynamic GAs