Journal of ELECTRICAL ENGINEERING, VOL. 59, NO. 1, 2008, 45–48

COMMUNICATIONS

DISTRIBUTED GENETIC ALGORITHMS: A SCHEME FOR GENETIC DRIFT AVOIDANCE

Taisir Eldos *

Genetic Algorithms (GA's) are of particular significance in applications with highly irregular search spaces, especially when the computing power requirement for exhaustive search is prohibitively large. GA's are capable of finding near optimal solutions and highly amenable to parallelism and hence Distributed Genetic Algorithms (DGA's) present an efficient solution to the time requirement issue. However, both the serial and the parallel implementations are subject to the genetic drift phenomenon, which drives the search into local optima if the exploration and exploitation are not well balanced. This work proposes a mechanism to achieve this balance by mimicking the natural catastrophes of life on earth; demes are partially destroyed in a randomized pattern and encouraged to reconstruct themselves towards better diversification and hence genetic drift avoidance or reduction.

K e y w o r d s: distributed, genetic, drift, population, optimization, search

1 INTRODUCTION

In this paradigm, the process of finding a solution to an optimization problem starts with a set of initial solutions, individuals or chromosomes, collectively called population or deme. A set of operators with predefined characteristics; frequencies, thresholds and rules are then applied to the population to perform reproduction. This process iterates producing better and better generations towards the end. This paradigm has been used to deliver efficient solutions to many complex problems in science, engineering and business.

Genetic algorithms demonstrate probabilistic convergence, and do not necessarily converge to an optimal solution. The search for a solution is based on the concept of schema, which was introduced by John Holland [1] to characterize the informal notion of "building block". Holland argued that the search for an optimal string combines exploitation (preservation of schemata) and exploration (creation of new schemata) in close to optimal proportion. However, there is only little theoretical support for the convergence and great deal of critic, especially when it comes to the schema theorem [2, 3] that is used to explain the behavior of the genetic algorithms as methods for building solutions from basic building blocks; it can only predict the next generation, and hence cannot predict the long term behavior of the genetic algorithms.

While it is relatively easy to design a suitable mapping and fitness measure, it is still hard to fine tune the parameters to achieve good results. This yields to the challenge of progressive loss of diversity in the population due to the consecutive application of genetic operators; as the tops tend to attract individuals forming subpopulations. The convergence on a peak or even small number of individuals without differential advantage is caused by a phenomenon called genetic drift, which is a result of errors in sampling that occur in small populations. To reduce the chance of dominance of such populations, one would want to provide for stable subpopulations to form around each peak, or at least a good collection of peaks, which is difficult to implement due to the lack of information. It would be even better if the subpopulations are allocated to peaks in proportion to their magnitudes [4]. An alternative approach is to consider the rate of decrease in the population fitness [5]. Several methods have been devised to deal with this problem by maintaining diversity; an example for proportional selection is to tune the selective pressure adaptively by a nonlinear scaling of the fitness function [6], another is the Evolutionary Local Selection Algorithm [7].

The proposed scheme targets the diversification aspect of the search through a new operator; the catastrophic effect. Although the idea itself is applicable to the standard genetic algorithm, it is more intuitive to apply it to the distributed model. The catastrophic effect operator mimics the effects of nature on the evolution process; earthquakes, landslides, volcanic eruptions, hurricanes and other massive destruction phenomena that are capable of destroying thousands of individuals in several cities or countries. The proposed addition to the basic evolutionary model will be evaluated by comparing it with the standard model using two measures; likelihood of optimality and population best fitness and standard deviation, and ergodicity.

This paper is organized as follows: section 2 reviews the various avenues of enhancing the performance of genetic algorithms, section 3 introduces the new operator and its characteristics, section 4 details the implementation of embedding the new factor into a generic algorithm, and section 5 discusses the results of the simulation.

* Department of Computer Engineering, Faculty of Computer and Information Technology, Jordan University of Science and Technology, Irbid 22110-3030 Jordan; eldos@just.edu.jo

ISSN 1335-3632

2 RELATED WORK

Traditionally, performance enhancement had few avenues; from domain specific adjustments to parameters optimization. Considerable performance improvement has been reported through the algorithm behavioral enhancements, although the optimal setting of the parameters depends on the application and the search stage [8]. Eric and Goldberg [9] provided guidelines to rationally select the parallel genetic algorithm. Adamidis [10] allowed the populations to behave differently to enrich the evolution process, and introduced a varying fitness function that increases penalty as the search progress towards the end, to assist in avoiding infeasible solutions and locating the area of the global optimum. Back [11] used self-adaptive cross-over and mutation rate and a variable population size model. Cantu-Paz [12] reported that careful selection of the migration scheme improves the converge speed. Matsumura [13] discussed the effects of the chromosome migration in distributed genetic algorithms; Eldos [14] proposed a selective migration scheme with conditions applied at both the source and the destination demes. Domain specific improvements were reported by some researchers; Chen [15] reported improved performance by using domain dependent settings like random respectful recombination and greedy cross-over.

3 CATASTROPHIC EFFECTS

DGAs are either symmetrical, where nodes have their own populations and perform similar tasks, and communicate periodically to exchange information and parts of their populations, or asymmetrical, where a master node assumes the administrative responsibility and the other nodes (slaves) perform computations only, mainly fitness evaluation. Normally, all nodes can exchange information. However, the cellular model restricts exchange among nodes within cells of certain structures, causing the quality of solutions achieved is topology dependent.

The proposed scheme introduces a randomized effect that partially destroy some demes and an adaptive recombination mechanism that reconstruct the demes gradually. We simulate the proposed enhancement using a symmetric DGA with cellular communication using a randomized effect characterized by five parameters:

1. Frequency, how often the effect will be imposed
2. Center, the deme that has the direct hit by the effect, selected randomly
3. Diameter, the extension of the effect
4. Level or strength, percentage of the deme size to be destroyed
5. Pattern, how victims are selected (partly at random and partly on fitness bases)

Normally, every deme has a chance to be the center of the effect and the diameter is chosen at random. Although the fitness is a measure of the ability to survive even natural disasters, all individuals are prone to destruction save the elitist (best 2 or 4). We assume that the central effect is superior to the peripheral, *ie* if a node happens to be under central and peripheral effect at the same time, then the central effect will prevail.

4 IMPLEMENTATION

We consider the binary knapsack problem as a case study, although it can be solved more effectively using other methods (dynamic programming), because its uniform structure makes it easy to study the schemata distribution as an ergodicity measure. The simulation is carried out on a hypercube with 32 nodes in a symmetric mode, and first degree neighbor information exchange. Each node runs the following procedure:

1. Create Initial Deme
2. Perform Selection (a percent of the population is moved to the next generation)
3. Apply Genetic Operations (cross-over and mutation)
4. Recombine (place offspring into the next generation)
5. Perform Migration (exchange individuals, and place entrants into the next generation)
6. Perform Probabilistic Destruction (based on a preset frequency)

a. If this deme is a center
   - i. Define characteristic and broadcast to neighbors
   - ii. Carry out the destruction according to the characteristics defined
   - iii. Revive the deme over few generations

b. If this deme is a member but not a center
   - i. Receive characteristics from the center
   - ii. Carry out the destruction according to the characteristics received
   - iii. Revive the deme over few generations

7. Rank (evaluate the fitness for new individuals in the next generation)
8. STOP if stopping criteria is satisfied, otherwise GOTO step 2

An instance of the knapsack problem with weights in the range 1 to 20 and values in the range 1 to 100 all drawn at random for 36 objects and knapsack capacity of 200. The optimal solution was unique with 23 objects, weight 198 and a value of 880. A manually modified instance is created from this one to have two optimal solutions to test the impact of the proposed scheme on instances with multiple optimal solutions.

The algorithm is set to have the following parameters: deme size of 20, selection rate is 25 % (with a probability proportional to the fitness, 5 individuals are moved to the next generation), cross-over rate of 50 % (5 pairs are mated in each generation, all at random), mutation rate of 5 % (carried out on a copy of an individual picked at random in each generation). In each generation, each node sends copies of 4 individuals to its 4 neighbors, and receive from each, thus 4 new individuals are added to each deme. Catastrophic effect occurs at frequencies of

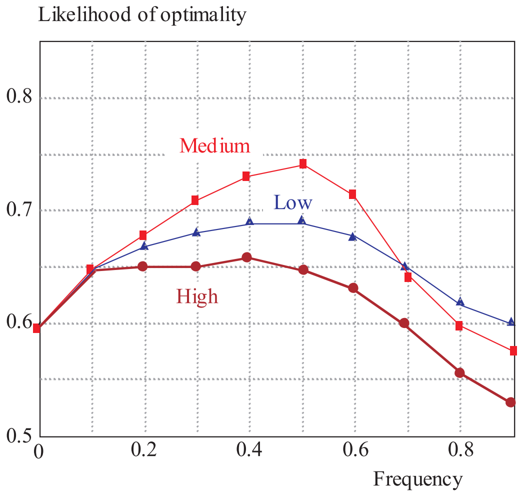

Fig. 1. Likelihood of Optimality versus Frequency

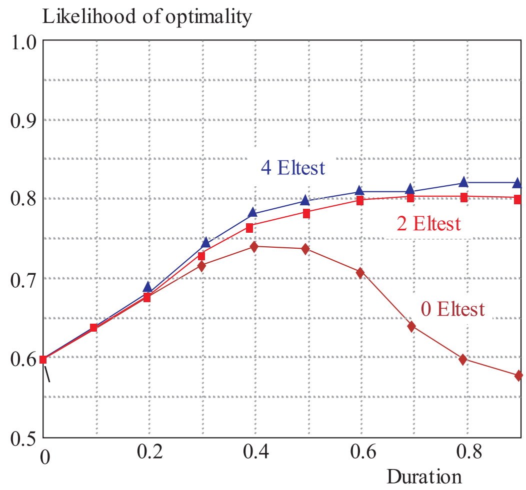

Fig. 2. Likelihood of Optimality versus duration of effect

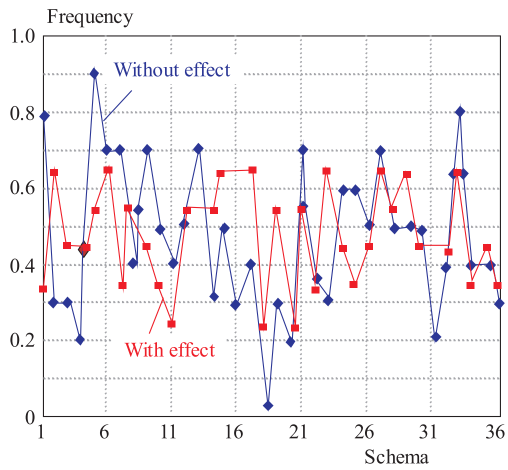

Fig. 3. Fitness Deviation versus Time

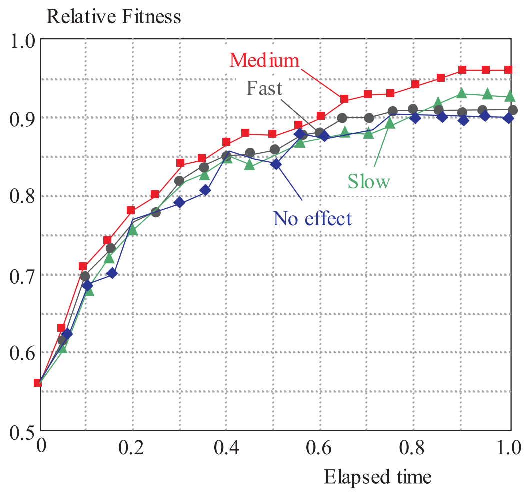

Fig. 4. Schema Spectrum

0.1 %, 0.2 %, up to 0.9 %, and degrees of destruction that are: light, medium and high with 10 %, 20 % and 30 % destruction at the centers and half of that in the peripheries. These settings keep the deme size fixed under normal condition, but when the size drops due to a catastrophe, it gets back to normal over few generations, typically 10, by inserting few randomly picked individuals and adjusting the migration rate to allow few more entrants. All runs stop processing after 100,000 generations.

5 DISCUSSION

The results show a decent improvement in the likelihood of optimality (percentage of runs that delivers optimal solution) at almost no extra cost. Figure 1 reports the impact of three different levels versus frequency. A 25 % increase associated with medium strength (20 % at the center and 10 % in the peripheries) at 0.5 % frequency (1 per 200 generations). Since no individual is immune to destruction, high strength effects at high frequency may even decrease the likelihood of optimality.

The effect application is intended to maintain diversity even at the end, but it was empirically found that it is of no use to continue more than half the way. Figure 2 shows that up to 30 % increase in the likelihood of optimality can be achieved if the elitist are immune to destruction.

Figure 3 records the standard deviation of the population fitness (average of all demes) over time; it is almost similar with the standard model (no effect) at the beginning and reasonably higher towards the end.

Figure 4 shows the schema spectrum during a typical run with 0.5 % frequency and medium strength. In an exhaustive search, the probability of appearance for each object is 50 %, but in a guided search, smaller and larger numbers are recorded corresponding to lightly unexplored and heavily explored subspaces. The min, max and standard deviation of the probabilities of appearance for all objects are 0.22, 0.70 and 0.13 for the proposed scheme, while they are 0.04, 0.92 and 0.21 without this effect. While this behavior is quite natural and healthy, having very extremely small coverage is a sign of poor exploration and a chance to miss the optimal solution. The proposed scheme reduces this difference, and avoids the extreme case of no exploration recorded sometimes by the

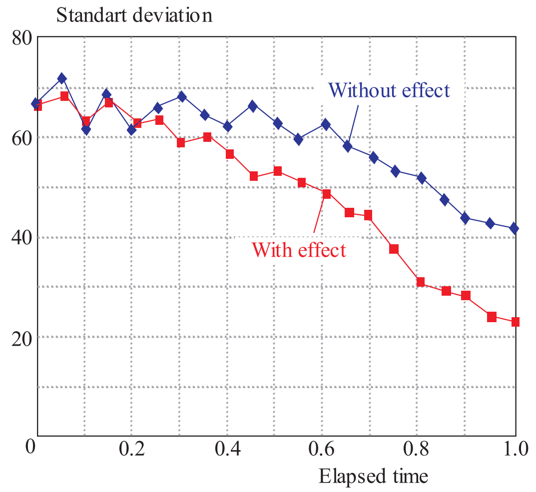

Fig. 5. Relative Fitness versus Time

standard model, where some schema never shows up in a candidate solution. Clearly, the effect makes the space sampling more ergodic, and diminishes the probability of improper convergence.

Figure 5 shows the relative fitness (fitness divided by the optimal) of the best solution over time (as percent of the total run time), and again the moderate effect shows a continuous superiority, which means a better solution is in hand no matter where in time the search process stops. A slight 5 % percent increase in the quality can be achieved.

6 CONCLUSION

The proposed work has shown that genetic algorithms could deliver better solutions when they behave more naturally; catastrophic effect in the form of randomized massive deletion of individuals has a positive impact on the solution quality. The exploration/exploitation balance reduced the genetic drift and hence increased the chance of finding optimal solutions; the likelihood of optimality increased by 21 % to 26 % in instances with unique optimal solution and by 22 % to 30 % in instances with multiple optimal solutions. In addition to that, a slight increase in the quality is recorded in most of the runs. Future work may address the problem quantitatively through some mathematical analysis.

References

[1] HOLLAND, J.: Adaptations in Natural and Artificial System, The University of Michigan Press, Ann Arbor, 1975.

[2] GREFENSTETTE, J.—BAKER, J.: How Genetic Algorithms Work: A Critical Look at Implicit Parallelism, Proceedings of the 3rd International Conference on Genetic Algorithms, San Mateo, CA, Morgan KaufmannPublishers, 1989, pp. 20–27.

[3] MUHLENBEIN, H.: How Genetic Algorithms Really Work: Mutation and Hill Climbing, In: Parallel Problem Solving from Nature (R. Manner and B. Manderick, eds.), North-Holland, Amsterdam, 1992, pp. 15–26.

[4] GOLDBERG, D.: Genetic Algorithm in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[5] MORAL, P.—KALLEL, L.—ROWE, J.: Modeling Genetic Algorithms with Interacting Particle Systems, In: Theoretical Aspects of Evolutionary Computing, Natural Computing Series (L. Kallel, B. Naudts and A. Rogers, eds.), Springer Verlag, 2001.

[6] MENCZER, F.—PARISI, D.: Network, Computation in Neural Systems **3** (4) (1992), 423–442.

[7] MENCZER, F.—DEGERATU, M.—STREET, W.: Evolutionary Computation, MIT Press, Cambridge, MA.

[8] SCHLIERKAMP-VOOSEN, D.—MUHLENBEIN, H.: Adaptation of Population Sizes by Competing Subpopulation, Proceeding of the IEEE International Conference on Evolutionary Computation, May 20-22, Nayoya University, Japan, 1996, pp. 330–335.

[9] CANTÚ-PAZ, E.—GOLDBERG, D. E.: Efficient Parallel Genetic Algorithms: Theory and Practice, Computer Methods in Applied Mechanics and Engineering **186** (2000), 221–238.

[10] ADAMIDIS, P.—KAZARLIS, S.—PETRIDIS, V.: Advanced Methods for Evolutionary Optimization, LSS'98, 8th IFAC/IFORS/IMACS/IFIP Symposium on Large Scale Systems: Theory and Applications, July 15-17, (Invited Session on Evolutionary Algorithms), University of Patras, Greece.

[11] BÄCK, T.—EIBEN, A. E.—van der VAART, N. A. L.: An Empirical Study on GAs without Parameters, In: Parallel Problem Solving from Nature PPSN V (Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J.J., and Schwefel, H.-P., eds.), Lecture Notes in Computer Science, Springer, 2000.

[12] CANTÚ-PAZ, E.: Migration Policies, Selection Pressure, and Parallel Evolutionary Algorithms, Journal of Heuristics **7**(4) (2001), 311–334.

[13] MATSUMURA, T.—NAKAMURA, M.—MIYAZATO, D.—OKECH, J.—ONAGA, K.: Effects of Chromosome Migration on Parallel and Distributed Genetic Algorithms, Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks, Taipei, Taiwan, Dec. 18-20, 1997, pp. 357–361.

[14] ELDOS, T.: A New Migration Model for the Distributed Genetic Algorithms, Proceedings of the International Conference on Scientific Computing (CSC'06), Las Vegas, NV, June 26-29, 2006, pp. 128–134.

[15] CHEN, S.—SMITH, S.: Putting the Genetics Back into Genetic Algorithms: Reconsidering the Role of Crossover in Hybrid Operators, In: Foundations of Genetic Algorithms 5, Morgan Kaufmann, 1999.

Received 3 July 2007

Taisir Eldos is an assistant professor at the Department of Computer Engineering at Jordan University of Science and Technology. Born in Karak-Jordan 1958, received BS Electronic Engineering at Menofia University, Menofia-Egypt in 1981, MS and PhD Computer Engineering at University of Alabama in Huntsville, Alabama, USA 1992 and 1996, respectively. He joined the Arab Army Royal Signal Division in Amman-Jordan during 1981-1983 as instructor, SEDCO in Amman-Jordan as design engineer and production manager during 1983-1989, and Microsystems International Birmingham Alabama, USA during 1989-1990 as director of engineering. Research interest includes associative processing, neural computing, evolutionary algorithms and data mining. Affiliation include: Jordan Engineers Association (JEA), Institute of Electrical and Electronic Engineers (IEEE) and Computational Intelligence Society (CIS).