GLIMPSE -Journal of Computer Science •Vol. 2(1), JANUARY-JUNE 2023, pp. 25-28

A REVIEW OF PARALLEL APPROACH USING GENETIC ALGORITHMS

Manish Kumar
Assistant Professor, Ajay Kumar Garg Engineering College,Ghaziabad, U.P., India
kumar.manish@akgec.ac.in

***Abstract*— Generally speaking, genetic algorithms are considered as simulations of evolution, of what kind ever. Genetic algorithms (Genetic algorithms) are powerful search techniques that are used successfully to solve problems in many different disciplines. Single and Parallel Genetic algorithms are the two approaches which are generally used to implement and promise gains in performance. In most of the cases, however, genetic algorithms are nothing else than probabilistic optimization methods which are based on the principles of evolution. This paper covers a genetic approach to finding near optimal solution for some graph theory problem. As such, there has been extensive research in this field. This survey attempts to collect, organize, and present in a unified way some of the most representative publications on genetic algorithms (parallel or single). To organize this literature, the paper presents a categorization of the techniques used to Genetic algorithms with the help of examples. Also, the paper describes some of the most significant problems in modeling and designing Genetic algorithms and presents some recent advancement. This paper is designed to cover a few important implicational aspects of genetic algorithm under a single umbrella.**

***Keywords*— Genetic algorithms, Hybridation, Master- slave genetic algorithms, Hierarchical genetic algorithms, Evolutionary, adjacency and matrix representation.**

I. INTRODUCTION

Genetic Algorithms (Genetic algorithms) are efficient search methods based on principles of natural selection and genetics. Knowledge-based information systems are designed to imitate the performance of biological systems. Such information systems use Evolutionary computing algorithms which are used for search and optimization applications and also includes fuzzy logic, which further provides an approximate reasoning basis for representing uncertain and imprecise knowledge. Genetic algorithms [3] are being applied successfully to find acceptable solutions to problems in business, engineering, and science [1]. Genetic algorithms are generally able to find good solutions in realistic amounts of time, but as they are applied to harder and bigger problems there is an increase in the time required to find satisfactory solutions. As significance, there have been numerous efforts to make Genetic algorithms faster, and one of the most promising choices is to use parallel implementations. As we know that, artificial neural networks imitate the brain or biological information processing mechanisms. So, Neural networks, fuzzy logic and evolutionary computing have shown capability on many problems but have not yet been able to solve the really complex problems. Over the last 3 decades several attempts have been made to develop optimization algorithms which simulate natural optimization processes. These attempts have resulted in the following optimization methods:

(1) Simulated Annealing, based on natural annealing processes.
(2) Artificial Neural Networks, based on processes in central nervous systems.
(3) Evolutionary Computation based on biological evolution processes [2].

The objective of this paper is to collect, classify and present some of the most relevant publications on parallel Genetic algorithms. There are several examples in the literature where parallel Genetic algorithms are applied to a particular problem, but the objective of this paper is not to itemize all the instances where parallel Genetic algorithms have been successful in finding fine solutions, but to highlight those publications that have contributed to the growth of the field of parallel Genetic algorithms in some way. The survey examines in more detail those publications that introduce something new or that attempt to explain why this or that works, but it also mentions a few of the application problems to show that parallel genetic algorithms are useful in the "real world" and are not just an academic curiosity.

II. GENETIC ALGORITHMS

Figure 1 shows diagrammatically where the field of genetic algorithms is placed in the hierarchy of knowledge based information systems or evolutionary computing.

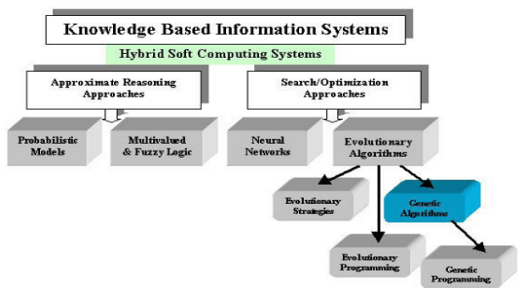

Figure 1. Steps of Genetic Algorithm

Evolutionary computing algorithms are probabilistic search algorithms which simulate natural evolution. They were proposed about 30 years ago as given in [4] and [5]. Their relevance to combinatorial optimization problems, have recently became an actual research topic. [2] Introduced genetic algorithms. Here, in these algorithms the search space of a problem is represented as a collection of individuals and these individuals are represented by character strings (or matrices), which are usually referred to as chromosomes. The purpose of using a genetic algorithm [6] is to find the individual from the search space with the finest "genetic material". The quality of an individual is measured with an evaluation function. The part of the search space to be examined is called the population. Roughly, a genetic algorithm works as follows:-

BEGIN *GA*
Make initial population.
WHILE *stop condition not satisfy* ***DO***
BEGIN
Select individuals from the population.
Produce offspring from the selected individuals.
Mutate the individuals.
Extend the population adding the offspring to it.
Reduce the extend population.
END
Output the best individual found.
END *GA*

As from the above genetic algorithm, the transition from one generation to the next consists of four basic components, which are as follows-

Selection: Mechanism for selecting individuals (strings) for reproduction according to their fitness (objective function value).

Crossover: Method of merging the genetic information of two individuals; if the coding is chosen properly, two good parents produces good children.

Mutation: In real evolution, the genetic material can by changed randomly by erroneous reproduction or other deformations of genes, e.g. by gamma radiation.

Sampling: Procedure which computes a new generation from the previous one and its off springs.
When compared with traditional optimization methods, like Newton or gradient descent methods, the following differences can be point out:

A. Genetic algorithms deals coded versions of the problem parameters instead of the parameters themselves.
b. Almost all conventional methods search from a single point, while genetic algorithms always operate on whole population of points (strings). So, genetic algorithms are much robust.
c. Normal genetic algorithms do not use any auxiliary information about the objective function value but can be applied to any kind of continuous or discrete optimization problem. The only thing is to specify a meaningful decoding function.
d. Genetic algorithms use probabilistic transition operators while conventional methods for continuous optimization apply deterministic transition operators i.e. some random components.

The basic mechanism in Genetic algorithms is Darwinian evolution: "bad traits are eliminated from the population because they do not survive the process of selection and the good traits survive since they are mixed by recombination (mating) to form better individuals. The notion of 'good' traits is the concept of building blocks (BBs), which are string templates [7]. Table 1 gives a list of different expressions, which are common in genetics, along with their equivalent in the framework of Genetic algorithms:

Natural Evolution	Genetic Algorithm
genotype	coded string
phenotype	uncoded point
chromosome	string
gene	string position
allele	value at a certain position
fitness	objective function value

Table 1. GA Expressions

III. PARALLEL GENETIC ALGORITHMS

The fundamental idea behind most of the parallel programs is to partition a task into chunks and to solve the chunks concurrently using multiple processors. So, divide-and-conquer approach can be applied to genetic algorithms in several ways, and the text contains many examples of successful parallel implementations. Some parallelization concepts use a single population, while some divide the population into several subpopulations. Some methods make use of massively parallel computer architectures and multicomputers. The classification of parallel genetic algorithms used here is similar to others like in [8], [9] and [10].but it is extended to include one more category. There are following three main types of parallel Genetic Algorithms:

(1) global single-population master slave genetic algorithms
(2) single-population fine-grained, and
(3) multiple-population coarse grained genetic algorithms.

In a master-slave GA there is a single population, but the evaluation of fitness is distributed among several processors as shown in Figure 2. Since in this type, selection and crossover consider the entire population so, it is also known

as global parallel Genetic algorithms. Fine-grained parallel Genetic algorithms are best suited for massively parallel computer systems and consist of structured population. Here, selection and mating are limited to a small neighborhood and neighborhoods were overlapped permitting some interaction among all the individuals as shown in Figure 3 for a schematic of this class of Genetic algorithms. The ideal case is to have only one individual for every processing element available. Multiple-population (or multiple-deme) Genetic algorithms are more complicated, because they consist several subpopulations which occasionally exchange individuals as shown in schemata of Figure 4. This exchange of individuals is called migration.

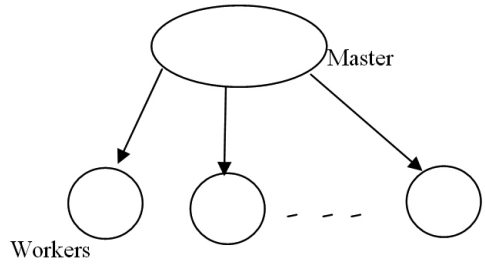

Figure 2. A schematic of a master-slave parallel GA.

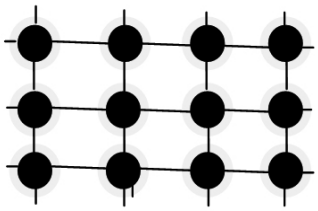

Figure 3. A schematic of a fine-grained parallel GA.

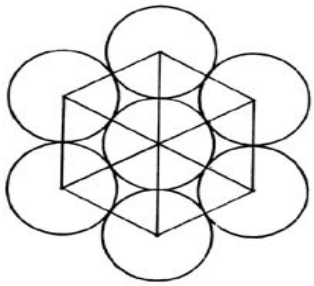

Figure 4. A schematic of a multiple-population parallel GA.

IV. HIERARCHICAL PARALLEL ALGORITHMS

A few researchers have tried to combine more than one method to parallelize Genetic algorithms, which results in hierarchical parallel genetic algorithms as shown in figure 5. Some of these new hybrid algorithms add a new degree of complication to the already complex scene of parallel genetic algorithm, but some hybrid algorithms keep the same complexity as one of their components. So, when two methods of parallelizing Genetic algorithms are combined they form a hierarchy. At the upper level most of the hybrid parallel Genetic algorithms are multiple-population algorithms. Some hybrids have a fine-grained GA at the lower level. ASPARAGOS was updated recently [11], and its ladder structure was replaced by a ring, because the ring has a longer diameter and allows a better differentiation of the individuals. Interestingly, a very similar concept was invented by Goldberg [12] in the context of an object-oriented implementation of a "community model" parallel GA. Moreover, hierarchical implementations can reduce the execution time more than any of their components alone.

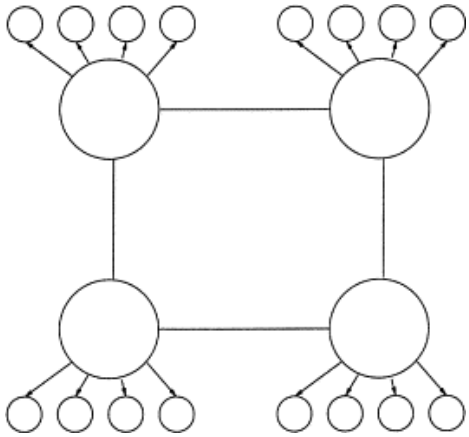

Figure 5. A schematic of a hierarchical parallel GA.

V. RECENT ADVANCEMENTS

This section summarizes some recent advancement in the theoretical study of parallel genetic algorithms. Firstly, I represent a result on master-slave Genetic algorithms. An important observation on master-slave Genetic algorithms is that as more processors are used, the time to evaluate the fitness of the population decreases. But at the same time, the cost of sending the individuals to the slaves increases. A recent study [13] concluded that the optimal solution is $S=(nT_f/T_c)^{1/2}$where n is the population size, T_f is the time it takes to do a single function evaluation, and T_c is the communications time. Recently, Cantu-Paz and Goldberg in [14] extended a simple genetic algorithm population sizing model to account for two bounding cases of coarse-grained genetic algorithms. Here, the two bounding cases were defined a set of isolated demes and a set of fully connected demes and it is seen that in the case of the connected demes, the migration rate is set to the highest possible value. The population size is also the major factor to determine the time that the genetic algorithm needs to find the solution. Further Cantu-Paz and Goldberg in [15] integrated the deme sizing models with a model for the communications time and predicted the expected parallel speed-ups for the two bounding cases. There are three main

conclusions from this theoretical analysis. First, the expected speed-up when the demes are isolated is not very significant. Second, the speed-up is much better when the demes communicate. And finally, there is an optimal number of demes that maximizes the speed-up. Parallel genetic algorithms are very complex and, of course, there are many problems that are still unresolved. A few examples includes how : (1) to determine the migration rate that makes distributed demes act like a single population, (2) to determine an adequate communications topology that permits the integration of good solutions, but problem is that it does not result in excessive communication costs, (3) to find if there is an optimal number of demes that maximizes reliability.

VI. SUMMARY AND CONCLUSIONS

This paper reviewed some of the most representative publications on parallel genetic algorithms. The review started by classifying the work on this field of genetic algorithms into four categories: (1) global master-slave parallelization, (2) fine-grained algorithms, multiple-population, and hierarchical parallel genetic algorithms. Some of the most essential contributions in each of these above categories were analyzed to undertake to identify the issues that affect the design and the implementation of each class of parallel genetic algorithms on existing parallel computer systems. The research on parallel genetic algorithms is dominated by multiple-population algorithms, and in outcome, this survey focused on them. The survey on multiple-population genetic algorithms discovered that there are several fundamental questions that were still unanswered after many years they were first identified. This class of parallel genetic algorithms is very complex, and the behavior of such classes is affected by many parameters and it seems that, the only way to achieve a greater understanding of parallel genetic algorithms is to study individual facts independently, and I have seen that most of the publications in parallel genetic algorithms concentrate on only one aspect i.e. migration rates, communication topology, or population size either neglecting or assuming simplifying assumptions on the others. I also reviewed some publications on master-slave and fine-grained parallel genetic algorithms and realized that the combination of different parallelization strategies results in faster algorithms.

REFERENCES

[1] GOLDBERG D. E., « Genetic and evolutionary algorithms come of age ». *Communicationsof the ACM*, vol. 37, n 3, p. 113–119, 1994.

[2] Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor: University ofMichigan Press.

[3] Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means ofNatural Selection*. MIT Press.

[4] Bremermann, H. J., Rogson, M. & Salaff, S. (1965). Search by Evolution. In Maxfield, M.,CallahanA.&Fogel, L. J. eds.) *Biophysics and Cyberntic Systems*, 157–167.Washington:Spartan Books.

[5] Rechenberg, I. (1973). *Optimierung Technischer Systeme Nach Prinzipien der Biologischen Information*. Stuttgart: Frommann Verlag.

[6] GORGES-SCHLEUTER M., ASPARAGOS : A population genetics approach to genetic algorithms. In VOIGT H.-M., MÜHLENBEIN H., SCHWEFEL H.-P., Eds., *Evolution and Optimization '89*, p. 86–94. Akademie-Verlag (Berlin), 1989.

[7] GORDON V. S., WHITLEY D., « Serial and Parallel Genetic Algorithms as Function Optimizers ». In FORREST S., Ed., *Proceedings of the Fifth International Conference on Genetic Algorithms*, p. 177–183, Morgan Kaufmann (San Mateo, CA), 1993.

[8] ADAMIDIS P., « Review of parallel genetic algorithms bibliography ». Tech. rep. version 1, Aristotle University of Thessaloniki, Thessaloniki, Greece, 1994.

[9] GORDON V. S., WHITLEY D., « Serial and Parallel Genetic Algorithms as Function Optimizers ». In FORREST S., Ed., *Proceedings of the Fifth International Conference on Genetic Algorithms*, p. 177–183, Morgan Kaufmann (San Mateo, CA), 1993.

[10] LIN S.-C., PUNCH W., GOODMAN E., « Coarse-Grain Parallel Genetic Algorithms: Categorization and New Approach ». In *Sixth IEEE Symposium on Parallel and Distributed Processing*, IEEE Computer Society Press (Los Alamitos, CA), October 1994.

[11] GORGES-SCHLEUTERM., « Asparagos96 and the Traveling Salesman Problem ».In BÄCK T., Ed., *Proceedings of the Fourth International Conference on Evolutionary Computation*, p. 171–174, IEEE Press (Piscataway, NJ), 1997.

[12] GORGES-SCHLEUTER M., ASPARAGOS : A population genetics approach to genetic algorithms. In VOIGT H.-M., MÜHLENBEIN H., SCHWEFEL H.-P., Eds., *Evolution and Optimization '89*, p. 86–94. Akademie-Verlag (Berlin), 1989.

[13] CANTÚ-PAZ E., « Designing efficient master-slave parallel genetic algorithms ». IlliGAL Report No. 97004, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1997.

[14] CANTÚ-PAZ E., GOLDBERG D. E., « Modeling Idealized Bounding Cases of Parallel Genetic Algorithms ». In KOZA J., DEB K., DORIGO M., FOGEL D., GARZON M., IBA H., RIOLO R., Eds., *Genetic Programming 1997 : Proceedings of the Second Annual Conference*, Morgan Kaufmann (San Francisco, CA), 1997.

[15] CANTÚ-PAZ E., GOLDBERG D. E., « Predicting speedups of idealized bounding cases of parallel genetic algorithms ». In BÄCK T., Ed., *Proceedings of the Seventh International Conference on Genetic Algorithms*, p. 113–121, Morgan Kaufmann (San Francisco), 1997.

ABOUT THE AUTHOR

Mr Manish Kumar has 19 years of experience currently working in AKGEC. He has deep interest in ML, DL and AI He has more than 15 National & International Publication.