On Stopping Criteria for Genetic Algorithms

Martín Safe[1], Jessica Carballido[1,2], Ignacio Ponzoni[1,2], and Nélida Brignole[1,2]

[1] Grupo de Investigación y Desarrollo en Computación Científica (GIDeCC)
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur, Av. Alem 1253, 8000, Bahía Blanca, Argentina
msafe@uns.edu.ar, {jac,ip@cs.uns.edu.ar}, dybrigno@criba.edu.ar
[2] Planta Piloto de Ingeniería Química - CONICET
Complejo CRIBABB, Camino La Carrindanga km.7
CC 717, Bahía Blanca, Argentina

Abstract. In this work we present a critical analysis of various aspects associated with the specification of termination conditions for simple genetic algorithms. The study, which is based on the use of Markov chains, identifies the main difficulties that arise when one wishes to set meaningful upper bounds for the number of iterations required to guarantee the convergence of such algorithms with a given confidence level. The latest trends in the design of stopping rules for evolutionary algorithms in general are also put forward and some proposals to overcome existing limitations in this respect are suggested.

Keywords: stopping rule, genetic algorithm, Markov chains, convergence analysis

1 Introduction

During the last few decades genetic algorithms (GAs) have been widely employed as effective search methods in numerous fields of application. They are typically used in problems with huge search spaces, where no efficient algorithms with low polynomial times are available, such as NP-complete problems [1].

Although in practice GAs have clearly proved to be efficacious and robust tools for the treatment of hard problems, the theoretical fundamentals behind their success have not been well-established yet [2]. There are very few studies on key aspects associated with how a GA works, such as parameter control and convergence analysis [3]. More specifically, the answers to the following questions concerning GA design remain open and constitute subjects of current interest. How can we define an adequate termination condition for an evolutionary process? [4–6]. Given a desired confidence level, how can we estimate an upper bound for the number of iterations required to ensure convergence? [7–9].

In this work we present a critical review of the state-of-the-art in the design of termination conditions and convergence analysis for canonical GAs. The main contributions in the field are discussed, as well as some existing limitations. On the basis of this analysis, future research lines are put forward. The article has been organized as follows. In section 2 the traditional criteria typically employed

A.L.C. Bazzan and S. Labidi (Eds.): SBIA 2004, LNAI 3171, pp. 405–413, 2004.

to express GA termination conditions are presented. Then, basic concepts on the use of Markov chain models for GA convergence analysis are summed up. Section 4 contains a discussion of the results obtained in the estimation of upper bounds for the number of iterations required for GA convergence. A description of the present trends as regards termination conditions for evolutionary algorithms in general is given next. Finally, some conclusive remarks and proposals for further work are stated in section 6.

2 Termination Conditions for the sGA

A simple Genetic Algorithm (sGA) exhibits the following features: finite population, bit representation, one-point crossover, bit-flip mutation and roulette wheel selection. The sGA and its elitist variation are the most widely employed kinds of GA. Consequently, this variety has been studied quite extensively. In particular, most of the scarce theoretical formalizations of GAs available in the literature are focused on sGAs. The following three kinds of termination conditions have been traditionally employed for sGAs [10, p. 67]:

- An upper limit on the number of generations is reached,
- An upper limit on the number of evaluations of the fitness function is reached, or
- The chance of achieving significant changes in the next generations is excessively low.

The choice of sensible settings for the first two alternatives requires some knowledge about the problem to allow the estimation of a reasonable maximum search length. In contrast, the third alternative, whose nature is adaptive, does not require such knowledge. In this case, there are two variants, namely genotypical and phenotypical termination criteria. The former end when the population reaches certain convergence levels with respect to the chromosomes in the population. In short, the number of genes that have converged to a certain value of the allele is checked. The convergence or divergence of a gene to a certain allele is established by the GA designer through the definition of a preset percentage, which is a threshold that should be reached. For example, when 90% of the population in a GA has a 1 in a given gene, it is said that that gene has converged to the allele 1. Then, when a certain percentage of the genes in the population (e.g. 80%) has converged, the GA ends. Unlike the genotypical approach, phenotypical termination criteria measure the progress achieved by the algorithm in the last n generations, where n is a value preset by the GA designer. When this measurement, which may be expressed in terms of the average fitness value for the population, yields a value beyond a certain limit ϵ, it is said that the algorithm has converged and the evolution is immediately interrupted.

The main difficulty that arises in the design of adaptive termination policies concerns the establishment of appropriate values for their associated parameters (such as ϵ in phenotypical rules), while for the criteria that set a fixed amount of iterations, the fundamental problem is how to determine a reasonable value for

that number, so that sGA convergence is guaranteed with a certain confidence level. In this case, the values not only depend on the dimension of the search space, but also on the rest of the parameters involved in the sGA, which include the crossover and mutation probabilities as well as the population size.

The minimum number of iterations required in a GA can be found by means of a convergence analysis. This study may be carried out from different approaches, such as the scheme theory [11, Chap. 2] or Markov chains [12–14]. The usefulness of the schema theorem has been widely criticised [15]. As it gives a lower bound for the expectation for one generation, it is very difficult to extrapolate its conclusions to multiple generations accurately. In this article we have concentrated on Markov chains because, as pointed out by Bäck et al. [2], this approach has already provided remarkable insight into convergence properties and dynamic behaviour.

3 Markov Chains and Convergence Analysis of the sGA

A *Markov chain* may be viewed as a stochastic process that traverses a sequence of states $\sigma_0, \sigma_1, \sigma_2, \ldots$ through time. The passage from state σ_i to state σ_{i+1} is called a transition. A distinguishing feature that characterizes Markov chains is the fact that, given the present state, future states are independent from past states, though they may depend on time. For a formal definition see, for example, [16, pp. 106–107].

Nix and Vose [12] showed how the sGA can be modelled exactly as a finite Markov chain, i.e. a Markov chain with a finite set of states. In their model, each state represents a population and each transition corresponds to the application of the three genetic operators. They found exact formulas for the transition probabilities to go from one population to another in one GA iteration as functions of the mutation and crossover rates. By forming a matrix with these transition probabilities and computing its powers, one can predict precisely the behaviour of the sGA in terms of probability, for fixed genetic rates and fitness function. This approach was taken up by De Jong et al. [17]. Unfortunately, the number of rows and columns of the corresponding matrices is equal to the number of all possible populations, which, according to [12], amounts to

$$\binom{2^\ell + n - 1}{n} . \tag{1}$$

This quantity becomes extremely large as the population size n or the strings length ℓ grows. Also notice that these matrices are not sparse because their entries are all non-zero probabilities. Therefore, this method can only be applied for small values of n and ℓ.

Nevertheless, Nix and Vose's formulation can lead to an analysis of the sGA convergence behaviour. For instance, they confirm the intuitive fact that, unless mutation rate is zero, each population is reachable from any other in one transition, i.e. the transition probability is non-zero for any pair of populations.

According to the theory about finite Markov chains, this simple fact has immediate consequences in the sGA behaviour as the number of iterations grows indefinitely. More specifically, whatever the initial population σ, the probability to reach any other population σ' after t iterations does not approach 0 as t tends to infinity. It tends to a positive limit probability instead. This limit depends on σ', but is independent from σ. Then, although the selection process tends to favour populations that contain high-fitness individuals by making them more probable, the constant-rate mutation introduces enough diversity to ensure that all populations are visited again and again. Thus, the sGA fails to converge to a population subset, no matter how much time has elapsed.

Moreover, Rudolph [14] showed that the same holds for more general crossover and selection operators, if a constant-rate mutation is kept. Nevertheless, reducing mutation rates progressively does not seem to be enough. Davis and Príncipe [13] presented a variation of the sGA that uses the mutation rate as a control parameter analogous to temperature in simulated annealing. They show how the mutation rate can be reduced during execution in order to ensure that the limiting distribution focuses only on populations consisting of replicas of the same individual, which is however, not necessarily a globally optimal one.

In contrast, the elitist version of the sGA, which always remembers the best individual found so far, does converge in a probabilistic sense. In this respect, Rudolph [14] shows that the probability of having found the best individual sometime during the process approaches 1 when the number of iterations tends to infinity, and he points out that this property does not mean that genetic algorithms have special capabilities to find the best individual. In fact, since any population has nonzero probability of being visited and there is a finite number of populations, then each of them will eventually be visited with probability 1 as the number of iterations grows indefinitely. Then, this observation lacks significance in practice because, for example, the direct enumeration of all the individuals guarantees the discovery of the global optimum in a finite time.

4 Stopping Criteria for the sGA with Elitism

Aytug and Koehler [7, 8] formulated a stopping criterion for the elitist sGA from the fact that all populations are visited with probability 1. Given a threshold α, they aimed at finding an upper bound for the number of iterations t required to guarantee that the global optimum has been visited with probability at least α in one of these iterations. Using Nix and Vose's model [12], they showed [7] that it is enough to have

$$t \geq \left\lceil \frac{\ln(1-\alpha)}{\ln(1-\min\{\mu^{\ell n}, (1-\mu)^{\ell n}\})} \right\rceil \tag{2}$$

to ensure that all the populations, and consequently all the individuals, have been visited with probability greater or equal to α. In equation 2, $\mu \in (0,1)$ is the mutation rate, ℓ is the length of the chains that represent the individuals, and n is the population size. Later, Aytug and Koehler [8] determined an upper

bound for the number of iterations required to guarantee, with probability at least α, that all possible individuals have been inspected, instead of imposing the condition on all the populations. In this way, they managed to improve the bound in (2) significantly, proving that a number of iterations t that satisfies

$$t \geq \left\lceil \frac{\ln(1-\alpha)}{n \ln(1 - \min\{\mu^\ell, (1-\mu)^\ell\})} \right\rceil \tag{3}$$

is enough to achieve this objective. Greenhalgh and Marshall [9] obtained similar results independently on the basis of simpler arguments. In the rest of this section, we will show that, in spite of being theoretically correct, these criteria are of little practical interest.

Let us consider a random algorithm (RA) that generates in each iteration a population of n individuals, not necessarily different from each other, chosen at random and independently. Just like Aytug and Koehler [7, 8] did for the elitist sGA, we shall determine the lowest number of iterations required to guarantee with probability at least α that the RA has generated all the possible individuals in the course of the procedure. Let us consider the populations $\sigma_1, \sigma_2, \sigma_3 \ldots$ generated by the RA and an element ω from the space of individuals (for example, a global optimum). Our objective is to find the lowest value for t so that $\Pr(\omega \in \sigma_1 \cup \cdots \cup \sigma_t) \geq \alpha$. Since the populations are generated independently from each other, then

$$\begin{aligned} &\Pr(\omega \in \sigma_1 \cup \cdots \cup \sigma_t) \\ &\qquad = 1 - \Pr(\omega \notin \sigma_1) \times \cdots \times \Pr(\omega \notin \sigma_t) = 1 - \left[1 - \left(\frac{1}{2}\right)^\ell\right]^{nt} . \end{aligned} \tag{4}$$

The expression in brackets is lower than 1, so the whole expression approaches 1 as t tends to infinity. Since ω is an arbitrary individual, (4) shows that the RA will visit all individuals with probability 1 if it is allowed to iterate indefinitely. Moreover, by applying logarithms to (4), we get

$$\Pr(\omega \in \sigma_1 \cup \cdots \cup \sigma_t) \geq \alpha \quad \text{if and only if} \quad t \geq \left\lceil \frac{\ln(1-\alpha)}{n \ln(1 - (1/2)^\ell)} \right\rceil . \tag{5}$$

This is an upper bound for the number of iterations required to ensure with probability at least α that the RA has examined all the individuals, and consequently discovered the global optimum.

Since (3) reaches its minimum for $\mu = 1/2$, then the bound for RAs given in (5) is always at least as good as the bound for GAs presented in (3). Then, the latter does *not* provide a stopping criterion in practice because it always suggests waiting for the execution of at least as many iterations as the amount that an RA without heuristics of any kind would require. Moreover, when μ tends to 0, which constitutes the situation of practical interest, the amount of iterations required by (3) grows to infinity. Figure 1 depicts the behaviour of (3) and its relation to (5).

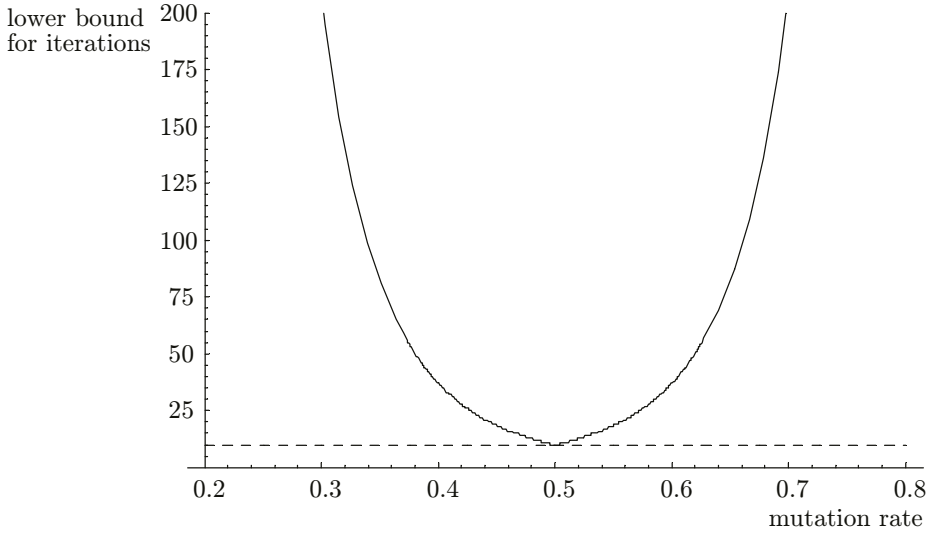

Fig. 1. This graph illustrates how the lower bound for GA iterations (3) (*continuous line*) grows quickly to infinity as mutation rate μ moves away from $1/2$. In this case $\ell = 6$, $n = 20$ and $\alpha = 0.95$. The lower bound for RA iterations (5) for the same ℓ, n and α is also indicated (*dashed line*) and coincides with the minimum attained by (3)

Due to the way Aytug and Koehler posed their problem, they were theoretically impeded to go beyond the bound for RAs given in (5). In fact, since they make no hypotheses on the fitness function, they implicitly include the possibility of dealing with a fitness function that assigns a randomly-chosen value to each individual. When this is the case, only exploration is required and no exploitation should be carried out. Therefore, the RA exhibits better performance than the sGA.

5 Present Trends in Stopping Rules for Evolutionary Algorithms

Whatever the problem, it is nowadays considered inappropriate to employ sets of fixed values for the parameters in an evolutionary algorithm [3]. Therefore, it would be unadvisable to choose a termination condition based on a preestablished number of iterations. Some adaptive alternatives have been explored in the last decade.

Among them we can cite Meyer and Feng [4], who suggested using fuzzy logic to establish the termination condition, and Howell et al. [5], who designed a new variant of the evolutionary algorithms called Genetic Learning Automata (GLA). This algorithm uses a peculiar representation of the chromosomes, where each gene is a probability. On this basis, a novel genotypical stopping rule is defined. The execution stops when the alleles reach values close to 0 or 1.

In turn, Carballido et al. [6] present a representative example of a stopping criterion designed ad hoc for a specific application, namely the traveling salesman problem (TSP). In that work, a genotypical termination criterion defined for both ordinal and path representations is proposed.

Finally, it is important to remark the possibility of increasing efficiency by using parallel genetic algorithms (pGAs) in particular. As stated in Hart et al. [18], the performance measurements employed in parallel algorithms, such as the speed-up, are usually defined in terms of the cost required to reach a solution with a pre-established precision. For this reason, when you wish to calculate metrics of parallel performance, it is incorrect to stop a pGA either after a fixed number of iterations or when the average fitness exhibits little variation. This constitutes a motivation for the definition of stopping rules based on the attainment of thresholds. The central idea is to stop the execution of the pGA when a solution that reaches this threshold is found. For instance, Sena et al. [19] present a parallel distributed GA (pdGA) based on the master-worker paradigm. The authors illustrate how this algorithm works by applying it to the TSP, using a lower bound estimated for the minimum-cost tour as threshold for the termination condition.

Nevertheless, threshold definition requires a good estimation of the optimum of the problem under study, which is unavailable in many cases. Unfortunately, the most recent reviews on pGAs ([20, 21]) fail to provide effective strategies to overcome these limitations.

6 Conclusions

Research work in this field shows that the sGA does not necessarily lead to better and better populations. Although its elitist version converges probabilistically to the global optimum, this is due to the fact that the sGA tends to explore the whole space, rather than to the existence of any special capability in its exploitation mechanism. This is not, indeed, in contradiction to the interpretation of sGAs as evolutionary mechanisms because the introduction of a fixed fitness function implies an assumption that may not be in exact correspondence with natural environments, whose character is inherently dynamic. As pointed out by De Jong [22], Holland's initial motivation for introducing the concept of GAs was to devise an implementation for robust adaptive systems, without focusing on the optimization of functions. Furthermore, De Jong makes a clear distinction between sGA and GA-based function optimizers. The successful results achieved through the use of the latter for the solution of hard problems often blurs this distinction.

Until recently, this trend has led researchers to look for a general measure of elitist sGA efficiency from a theoretical viewpoint, applicable when finding the solution of *any* optimization problem on binary strings of length ℓ [8, 9]. The smoothness of the fitness function is extremely important when choosing the most convenient kind of heuristic strategy to adopt when facing a given problem. The higher the smoothness, the higher the exploitation level and conversely, as

the function is less smooth, more exploration is required. This fact is so clear that it should not be overlooked. Since no hypotheses on the fitness function have been made, and also considering that no measure of its smoothness has been included in its formula, this approach is overestimating exploration to the detriment of exploitation, this being just the opposite of what one really wishes in practice when implementing a heuristic search.

In view of the fact that the sGA is not an optimizer of functions, efforts should be directed to the devise of adequate modifications to tackle each specific problem in order to design an optimizer that is really efficient for a determinate family of functions. Besides, it is important to remark that current trends are towards the employment of adaptive termination conditions, either genotypical or phenotypical, instead of using a fixed number of iterations, because for most applications in the real world the mere estimation of the size of the search space constitutes in itself an extremely complex problem.

Acknowledgments

The authors would like to express their acknowledgment to the "Agencia Nacional de Promoción Científica y Tecnológica" from Argentina, for their economic support given through Grant N°11-12778. It was awarded to the research project entitled "Procesamiento paralelo distribuido aplicado a ingeniería de procesos" (ANPCYT Res N°117/2003) as part of the "Programa de Modernización Tecnológica, Contrato de Préstamo BID 1201/OC-AR".

References

1. Brassard, G., Bratley, P.: Algorithmics: Theory and Practice. Prentice-Hall, Inc., New Jersey (1988)
2. Bäck, T., Hammel, U., Schwefel, H.P.: Evolutionary computation: Comments on the history and current state. IEEE Transactions on Evolutionary Computation **1** (1997) 3–17
3. Eiben, Á.E., Hinterding, R., Michalewicz, Z.: Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation **3** (1999) 124–141
4. Meyer, L., Feng, X.: A fuzzy stop criterion for genetic algorithms using performance estimation. In: Proceedings of the Third IEEE Conference on Fuzzy Systems. (1994) 1990–1995
5. Howell, M., Gordon, T., Brandao, F.: Genetic learning automata for function optimization. IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics **32** (2002) 804–815
6. Carballido, J.A., Ponzoni, I., Brignole, N.B.: Evolutionary techniques for the travelling salesman problem. In Rosales, M.B., Cortínez, V.H., Bambill, D.V., eds.: Mecánica Computacional. Volume XXII. Asociación Argentina de Mecánica Computacional (2003) 1286–1294
7. Aytug, H., Koehler, G.J.: Stopping criterion for finite length genetic algorithms. INFORMS Journal on Computing **8** (1996) 183–191
8. Aytug, H., Koehler, G.J.: New stopping criterion for genetic algorithms. European Journal of Operational Research **126** (2000) 662–674

9. Greenhalgh, D., Marshall, S.: Convergence criteria for genetic algorithms. SIAM Journal on Computing **20** (2000) 269–282
10. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, New York (1996)
11. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Longman, Inc., Reading, Massachusetts (1989)
12. Nix, A.E., Vose, M.D.: Modeling genetic algorithms with Markov chains. Annals of Mathematics and Artificial Intelligence **5** (1992) 79–88
13. Davis, T.E., Príncipe, J.C.: A Markov chain framework for the simple genetic algorithm. Evolutionary Computation **1** (1993) 269–288
14. Rudolph, G.: Convergence analysis of canonical genetic algorithms. IEEE Transactions on Neural Networks **5** (1994) 96–101
15. Poli, R.: Exact schema theorem and effective fitness for GP with one-point crossover. In Whitley, L.D., Goldberg, D.E., Cantú-Paz, E., Spector, L., Parmee, I.C., Beyer, H.G., eds.: Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann (2000) 469–476
16. Çınlar, E.: Introduction to Stochastic Processes. Prentice-Hall, Inc., Englewood Cliffs, New Jersey (1975)
17. De Jong, K.A., Spears, W.M., Gordon, D.F.: Using Markov chains to analyze GAFOs. In Whitley, L.D., Vose, M.D., eds.: Proceedings of the Third Workshop on Foundations of Genetic Algorithms, Morgan Kaufmann (1995) 115–137
18. Hart, W.E., Baden, S., Belew, R.K., Kohn, S.: Analysis of the numerical effects of parallelism on a parallel genetic algorithm. In: Proceedings of the 10th International Parallel Processing Symposium, IEEE Computer Society (1996) 606–612
19. Sena, G.A., Megherbi, D., Isern, G.: Implementation of a parallel genetic algorithm on a cluster of workstations: travelling salesman problem, a case study. Future Generation Computer Systems **17** (2001) 477–488
20. Alba, E., Tomassini, M.: Parallelism and evolutionary algorithms. IEEE Transactions on Evolutionary Computation **6** (2002) 443–462
21. Veldhuizen, D.A.V., Zydallis, J.B., Lamont, G.B.: Considerations in engineering parallel multiobjetive evolutionary algorithms. IEEE Transactions on Evolutionary Computation **7** (2003) 144–173
22. De Jong, K.A.: Genetic algorithms are NOT function optimizers. In Whitley, L.D., ed.: Proceedings of the Second Workshop on Foundations of Genetic Algorithms, Morgan Kaufmann (1993) 5–17