An overview of the parameter-less genetic algorithm

Fernando G. Lobo
ADEEC, FCT
Universidade do Algarve
Campus de Gambelas
8000-062 Faro, Portugal
flobo@ualg.pt

David E. Goldberg
Dept. General Engineering
University of Illinois at Urbana-Champaign
104 South Mathews Avenue
Urbana, IL 61801
deg@uiuc.edu

Abstract

This paper presents an overview of the parameter-less genetic algorithm and shows its application to a network expansion problem. The technique simplifies genetic algorithm operation by incorporating knowledge of parameter selection and population sizing theory in the genetic algorithm itself.

1 Introduction

The need for solving complex problems occur in a variety of domains and genetic algorithms (GAs) can be useful tools for that purpose. Unfortunately, traditional GAs require the specification of a number of parameters for which users usually don't know how to specify. This paper presents a technique that eliminates this drawback by doing a rational and automated parameter selection on behalf of the user.

In the original work (Harik & Lobo, 1999), the validity of the parameter-less GA was illustrated on artificial problems. In this paper it is illustrated on a quasi-real-world problem, a scenario that users are more likely to encounter in practice.

The paper starts by reviewing the parameter-less genetic algorithm. Section 3 describes the application problem, and section 4 shows the application of the parameter-less genetic algorithm.

2 Overview of the parameter-less GA

This section briefly reviews the parameter-less GA. With the parameter-less GA, the user doesn't need to specify the selection rate, the crossover probability and the population size parameters.

2.1 Selection Rate and Crossover Probability

The selection rate s and crossover probability p_c are preset to fixed values ($s = 4$, $p_c = 0.5$) in order to obey the schema theorem and ensure building block growth.

At first sight, one could argue that setting $s = 4$ and $p_c = 0.5$ for all problems constitutes a similar kind of mistake that other practitioners have done in the past when adopting the so-called "standard parameter settings". However, there is an important difference in this case. Previous theoretical studies (Goldberg, Deb, & Thierens, 1993) have shown that there must be a balance between selection pressure and schema disruption in order to ensure building block growth. This argument comes from a simplified view of the schema theorem. Let $\phi(H, t)$ be the effect of the selection operator on schema H at generation t, and $\epsilon(H, t)$ be the disruption factor on schema H due to the crossover operator. Then the overall net growth ratio on schema H at generation t is given by:

$$\phi(H, t)[1 - \epsilon(H, t)]$$

The above expression is nothing but a simplification of the schema theorem. Under the conservative hypothesis that a schema is destroyed during the crossover operator, and substituting s and p_c in the formula above, we obtain that the growth ratio of a schema is given by the expression:

$$s(1 - p_c)$$

Setting $s = 4$ and $p_c = 0.5$ gives a net growth factor of 2, and ensures that the necessary building blocks will grow. Other values of s and p_c could also be used as long as the net growth factor is somewhat greater than 1, and some care is taken not to fall in the extreme cases of a very high or very low selection pressure (Goldberg, Deb, & Thierens, 1993). In other words, setting s and p_c to fixed values is a rational decision which is backed up by previous theoretical work.

2.2 Population sizing

The population size is a critical parameter in a GA. Too small and the GA converges to poor solutions. Too large and the GA spends unnecessary computational resources. There are theoretical models (Harik, Cantú-Paz, Goldberg, & Miller, 1997) that can be used to size populations but they are not easy to apply in practice because they rely on parameters that are usually unknown and are also hard to estimate for real world problems.

Although not trivial to put in practice, the theoretical models on population sizing are important and crucial for understanding the role of the population in a GA. Among other things, an important lesson of those models is that setting the population size to a fixed value regardless of the problem's size and difficulty, is certainly a mistake.

In practice, the bottom line is that the user has to do some experimentation and guess the population size. But guessing right is pure luck and most likely the user will guess wrong by doing one of the following two mistakes: (1) a population size that is too small, or (2) a population size that is too large.

The parameter-less GA uses a technique that was developed on purpose to eliminate the need of guessing the population size, and therefore, avoid the two type of errors. The basic idea is to continuously increase the population size in an attempt to reach the right sizing. When to stop this growing process of the population is left to the user. He/she will decide when to stop as soon as he/she realizes that is not worth to wait more for an improvement in solution quality.

The way the parameter-less GA simulates a kind of continuous population size growth is by establishing a race among multiple populations of various sizes (powers of 2). The different populations are at different stages of evolution, the smaller ones being ahead of the larger ones in terms of generations. For example, a snapshot of the parameter-less GA at a particular point in time could reveal the existence of 3 populations whose sizes could be 256, 512, and 1024. The population of size 256 could be running its 30^{th} generation, while the population of size 512 could be on generation 6, and the population of size 1024 could still be on generation 1.

As time goes by, the smaller populations are eliminated and larger populations are created. The creation and deletion of populations is controlled by inspecting the average fitness of the populations and taking rational decisions based on those readings. For example, if the population of size 512 has an average fitness greater than that of the population of size 256, then there is no point in continuing running the small population anymore because it is very unlikely that the smaller population will produce a fitter individual than the larger population. Recall that the larger population is at a much earlier stage of evolution but already contains better individuals than those contained in the smaller one, a clear indication that the smaller population is not large enough.

The interested reader should refer elsewhere (Harik & Lobo, 1999) (Lobo, 2000) for a through description and implementation details of the parameter-less GA.

2.3 A Note About Mutation

The parameter-less technique ignores the mutation operator. We recognize that mutation can be important for many problems but haven't considered yet how to automate it within the rest of the parameter-less GA framework. Mutation makes small variations on solutions and thus is not likely to benefit from very large populations. On the contrary, crossover requires large population sizes in order to mix the bits and pieces of the different solutions.

The next section describes an electrical utility network expansion problem, which will be solved subsequently by the parameter-less GA.

3 A network expansion problem

Consider the example shown in Figure 1. The figure shows a region that doesn't have electrical facilities, an hypothetical instance of the network expansion problem.

There are four types of entities depicted in the figure: cables, substations, possible transformer locations, and houses. These entities are represented in the figure by lines, squares, triangles, and dots respectively. In the example there are five substations (squares) connected by cables in a network. The objective of the problem is to expand the network so that the houses (dots) can get electricity. Moreover, the electrical utility company would like to do so with minimum cost.

The substations are the only entities that take part of the electrical infrastructure. The transformers (triangles in the figure) don't exist yet but there are a number of possible locations where they can be built. These locations are given in advance by the electrical utility and may take into account a variety of restrictions.

The total cost of expanding the network is the sum of the costs of all the cables and transformers that need to be built. Each transformer that is built has a fixed cost associated and the cost of each cable is proportional to its length.

In summary, one has to decide which transformers should be built. Once that decision is taken, expanding the network can be done with a straightforward computation. Figure 1 through 4 illustrate the fitness function by showing the construction of the network on a hypothetical 10-transformer network problem. There are 10 possible locations (the 10 triangles in Figure 1) to build transformers. For each location, a binary decision must be made by the power company: build or not build a transformer in that location. In Figure 2, 4 locations are selected for building transformers and the other 6 are left out. The example corresponds to solution 0101000110, which is one out of the 2^{10} possible solutions. Then, a graph is constructed in the following way. For each selected transformer node, an edge is added from that node to all the existing substation nodes and to all the other selected transformers. Following that, a minimum spanning tree of the graph is computed (see Figure 3). Once the tree is constructed, each house can be connected to the network by adding an edge from the house to the closest node of the tree. The final result is shown on

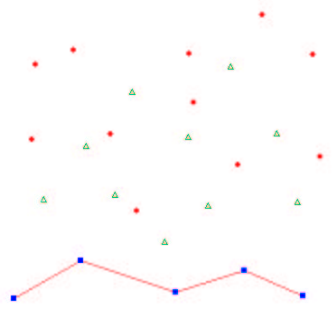

Figure 1 A hypothetical 10-bit (10-transformer) network expansion problem instance.

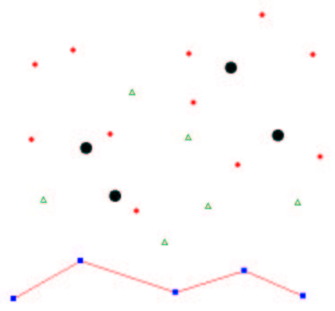

Figure 2 4 transformer nodes are selected to be built. They are represented on the figure by the large circles.

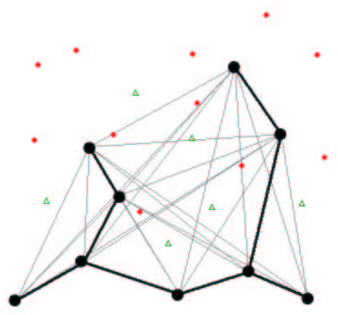

Figure 3 A minimum spanning tree of the graph is computed.

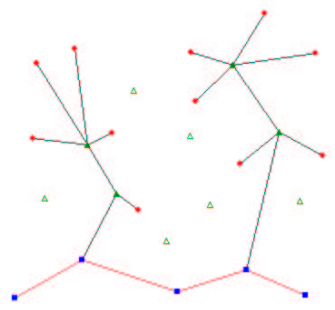

Figure 4 Each house connects to the closest node of the minimum spanning tree.

Figure 4.

It should be stressed that in this paper we are treating the objective function as a black-box. In other words, the purpose of the paper is not to show that the GA is superior or even competitive with specialized algorithms specifically tuned for this particular problem. Instead, the purpose of the paper is to illustrate how GA technology may be applied in an environment where not much is known other than the objective function values of individual solutions.

We have described a method to expand the existing network once the transformers to be built are specified. The decision variables of the problem are binary variables, one for each possible transformer location, indicating whether that transformer should be built or not. Thus, if the electrical utility company specifies ℓ possible locations to build transformers, the total number of possible network configurations is 2^ℓ and the application of a genetic algorithm is straightforward.

4 Parameter-less GA Application

This section shows the application of the parameter-less GA to the network expansion problem described during the previous section. In particular, the GA is applied to a 60-bit problem instance. It should be stressed that the parameter-less technique relieves the user from having to specify the population size, selection rate, and crossover probability, but it does not relieve the user from specifying the type of GA operators to be used. In fact, the parameter-less technique can be used with any kind of GA.

In a first experiment, we use the parameter-less technique coupled with a simple GA using uniform crossover and with mutation turned off. Table 1 shows a trace of the execution of a single run of the parameter-less GA. The algorithm starts with a population of size 16 and continuously increases (doubles) its size. For each population size, Table 1 shows the best solution quality found. For example, when using a population of size 128, the best solution had a cost of 1990.25.
A number of observations are worth mentioning. First and foremost, the experiment is very simple (of course, there are no parameters!). Larger and larger populations are continuously spawned. By doing that, the parameter-less technique injects more and more power into the GA as time goes by.

The parameter-less GA was stopped when the population of size 65536 had already converged but the population of size 131072 was still in its early generations (see Harik and Lobo (1999) and Lobo (2000) for details of the execution of the algorithm). In this specific exam-

Table 1 The parameter-less GA on a 60-bit problem instance.

population size	solution quality with parameter-less SGA
16	2131.79
32	2109.33
64	2004.93
128	1990.25
256	1971.50
512	1984.76
1024	1986.49
2048	1967.21
...	...
65536	1967.21

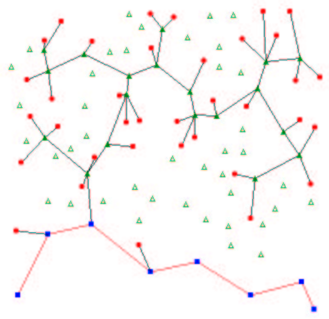

Figure 5 The best solution found by the parameter-less GA on a 60-bit problem instance.

ple, the user (ourselves) stopped the parameter-less GA at that point because we didn't want to wait longer for a better solution quality. That of course depends on a problem by problem basis and from user to user. For instance, another user might have been happy with the solution quality of 1990.25 and might have stopped the parameter-less GA by the time that solution was found (population size 128 in the example).

The best solution was found at population size 2048 (see Fig. 5). Larger populations also found that solution but couldn't get any better. If the user knew that beforehand he/she would have pressed the stop button at population size 2048 and would have saved computational time. Unfortunately, the user doesn't know that beforehand. In fact, the user can't predict what would happen if he/she had let the algorithm run for a longer time; it's unknown whether the algorithm can reach a better solution quality or not.

The only thing that we can say is that if the user is not satisfied with the solution quality that he has gotten so far then the best option is to put more power into the GA and that can be done by increasing the population size; the parameter-less GA does that automatically.

5 Summary and Conclusions

This paper reviewed the parameter-less genetic algorithm and showed a practical application of it to a utility network expansion problem. The problem has characteristics that contrast with those of pure artificial problems, and constitutes a more representative scenario of what users might encounter in practice.

With the parameter-less GA the user does not have to do trial and error experiments to find adequate parameter settings for the GA. It is our strong belief that GAs should be designed with the user in mind. That is the only way to have more and more people using them. This paper makes an important effort in that direction and we hope it can be useful for practitioners seeking to apply state GA technology to solve real world problems.

Acknowledgements

This work was sponsored by FCT/MCT under grants POSI/SRI/42065/2001 and POCTI/MGS/37970/2001.

The work was also sponsored in part by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grants F49620-00-0163, the National Science Foundation under grant DMI-9908252.

The views and conclusions contained herein are our own and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the National Science Foundation, the U. S. Army, or the U.S. Government.

References

Goldberg, D. E., Deb, K., & Thierens, D. (1993). Toward a better understanding of mixing in genetic algorithms. *Journal of the Society of Instrument and Control Engineers, 32*(1), 10–16.

Harik, G. R., Cantú-Paz, E., Goldberg, D. E., & Miller, B. L. (1997). The gambler's ruin problem, genetic algorithms, and the sizing of populations. In Bäck, T. (Ed.), *Proceedings of 1997 IEEE International Conference on Evolutionary Computation* (pp. 7–12). New York: IEEE Press.

Harik, G. R., & Lobo, F. G. (1999). A parameter-less genetic algorithm. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., & Smith, R. E. (Eds.), *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference* (pp. 258–267). San Francisco, CA: Morgan Kaufmann.

Lobo, F. G. (2000). *The parameter-less genetic algorithm: Rational and automated parameter selection for simplified genetic algorithm operation*. Doctoral dissertation, Universidade Nova de Lisboa, Lisboa.