Self-Adaptive Population Size Adjustment for Genetic Algorithms

Michael Affenzeller[1], Stefan Wagner[1], Stephan Winkler[2]

Upper Austrian University of Applied Sciences
[1] Department of Software Engineering
[2] Research Center Hagenberg
Softwarepark 11, 4232 Hagenberg, Austria
{michael,stefan,stephan}@heuristiclab.com

Abstract. Variable population sizing techniques are rarely considered in the theory of Genetic Algorithms. This paper discusses a new variant of adaptive population sizing for this class of Evolutionary Algorithms. The basic idea is to adapt the actual population size depending on the actual ease or difficulty of the algorithm in its ultimate goal to generate new child chromosomes that outperform their parents.

1 Introduction

The effects of population size of Evolutionary Algorithms (EAs) has been subject of intensive research by the EA community since its beginning. Already in the first proposed variants of Evolution Strategies (ES), the community paid a lot of attention to the appropriate choice for μ (population size) and λ (birth surplus) for the certain variants of (μ, λ)-ES and $(\mu + \lambda)$-ES, respectively (for an overview see [5], e.g.). In the Genetic Algorithm (GA) community, too, well-known researchers concentrated on the effects of population size with respect to achievable solution quality right from the beginning [8]. All these considerations assume the population size to be constant during the run of an EA. However, there are many hints in population genetics and also in nature itself that underpin the reasonability of a variable population size during the evolution of a certain species or during the run of an EA, respectively.

The Genetic Algorithm with Variable Population Size (GAVaPS) [3] eliminates the population size as an explicit parameter by introducing the properties "age" and "maximal lifetime" [3]. A further strategy to introduce adaptive population sizing schemes is given by the so called APGA (Adaptive Population size GA) [4] which is based upon a steady state GA and adopts the lifetime of an individual as it stands. Eiben et al. [6] introduce a growing population size in case of high fitness improving capacity or in case of longer lasting stagnation. Short stagnation periods cause the decrease of population size.

The adaptive population sizing scheme presented in this article follows a basically different approach by interpreting the actual population size as a consequence of the capacities of the actually applied reproduction operators (crossover

and mutation) to create offspring that outperform the solution quality of their own parents. As long as it is possible to generate new and in this sense successful individuals, the next generation grows. Therefore, apart from an upper limit that comes with the limited environmental resources, the respective next generation's population is filled up with new individuals as long as the allele pool of the present generation is not yet fully utilized.

This article is organized as follows: Section 2 describes premature convergence on the basis of a typical GA application and aims to the bottom of reasons for premature convergence. The goal of Section 3 is to use these findings in order to discuss generic algorithmic extensions and also to describe their concrete implementation. The ability of these further developed algorithmic concepts to preserve essential genetic information more efficiently is demonstrated on the basis of a concrete example that has already been used in Section 2. Finally, Section 4 sums up the main results of the paper and states some perspectives for ongoing future research.

2 Premature Convergence and Its Reasons

GAs are also frequently faced with a problem which, at least in its impact, is quite similar to the problem of stagnating in a local but not global optimum. This drawback, called premature convergence in the terminology of GAs, occurs if the population of a GA reaches such a suboptimal state that the genetic solution manipulation operators (crossover and mutation) are no longer able to produce offspring that are able to outperform their parents (e.g. [7], [1]). In general this happens mainly when the genetic information stored in the individuals of a population does not contain that genetic information which would be necessary to further improve the solution quality. Therefore, in contrast to the present contribution, the topic of premature convergence is considered to be closely related to the loss of genetic variation in the entire population in GA-research [10], [11]. Here we do not identify the reasons for premature convergence in the loss of genetic variation in general but more specifically in the loss of what we call essential genetic information, i.e. in the loss of alleles which are part of a global optimal solution.

This basic concept of a GA poses several questions and associated problems:

- Is crossover always able to fulfil the implicit assumption that two above average parents can produce even better children?
- Which of the available crossover operators is best suited for a certain problem in a certain representation?
- Which of the resulting children are "good" recombinations of their parents chromosomes?
- What makes a child a "good" recombination?
- Which parts of the chromosome of above average parents are really worth being preserved?

In order to get to the bottom of the question how a Genetic Algorithm could optimally utilize the available gene pool, let us first consider the internal processes of a very typical simple GA applied to a typical combinatorial optimization problem, namely a 130 city benchmark Traveling Salesman Problem (TSP). By reasons of compactness, the results are mainly shown on the basis of diagrams and give only a brief description of introduced operators, parameter settings, and test environments. Furthermore, the chosen benchmark instance (the ch130 benchmark TSP) is of rather small dimension in order to allow the observation of essential alleles during the run of the algorithm.

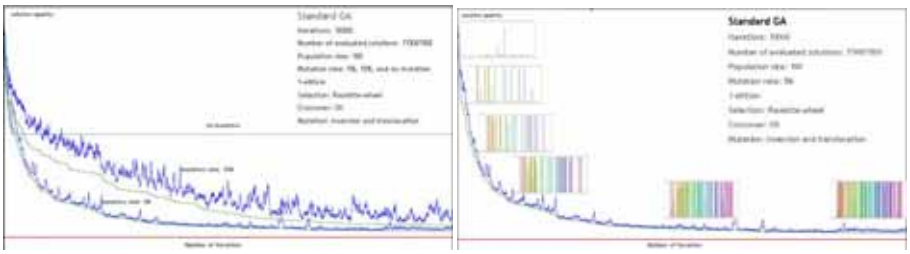

Fig. 1. The effect of mutation for certain mutation rates (left diagram) and the distribution of essential genetic information for a mutation rate of 5% (right diagram) both in case of a standard GA for the ch130 benchmark TSP.

The results displayed in Fig. 1 (left diagram) show the effect of mutation for reintroducing already lost genetic information. The horizontal line of the diagram shows the number of iterations and the vertical line stands for the solution quality. The bottom line indicates the global optimal solution which is known for this benchmark test case. The three curves of the diagram show the performance of a Genetic algorithm with no mutation, with a typical value of 5% mutation as well as a rather high mutation rate of 10%. For each of the three curves the lower line stands for the best solution of the actual population and the upper line shows the average fitness value of the population members. The results with no mutation are extremely weak and the quality curve stagnates very soon and far away from the global optimum; the best and average solution quality are the same and no further evolutionary process is possible - premature convergence has occurred.

The right diagram of Fig. 1 shows the distribution of essential alleles over the iterations for a standard GA with a mutation rate of 5%. The interesting thing is that some minor ratio of essential alleles (visualized by inserting bar charts which have to be interpreted as snapshots after a certain number of iterations approximately corresponding to the position in the figure) is rapidly fixed in the population and the majority of essential alleles which are still missing have disappeared in the entire population. During the further run of the algorithm it is only mutation which can reintroduce this essential genetic information. As is shown in Fig. 1, without mutation premature convergence occurres at this early

state of evolutionary search. But with an appropriate mutation rate (5% in this example) more and more essential alleles are discovered ending up with quite a good solution.

These observations show that the role of mutation may be much more important than usually considered in GA literature where mutation is often described as a background operator. As our small example has shown, mutation can also hide severe weaknesses of certain crossover operators w.r.t. their inability to preserve the essential genetic information during the run of a GA.

In the following section a new generic GA variant is introduced that uses adaptive population sizing in combination with Offspring Selection [2] in order to bring out the best of the gene pool which is available at a certain generation of a GA.

3 Efficient Preservation of Essential Alleles by Adaptive Population Sizing

Assuming generational replacement as the underlying replacement strategy, the most essential question at generation i is, which parts of genetic information from generation i should be maintained in generation $i + 1$ and how this could be done most effectively applying the available information (chromosomes and according fitness values) and the available genetic operators selection, crossover and mutation.

The further developed algorithmic concepts based upon GA-solution manipulation operators aim to achieve this goal by trying to bring out as much progress from the actual generation as possible and loosing as little genetic diversity as possible at the same time.

The implementation of this idea is done with on the fly population size adjustment in that sense that potential offspring generated by the basic genetic operators are accepted as members or the next generation if and only if they are able to outperform the fitness of their own parents and if they are new in that sense that their chromosome consists of a concrete allele alignment that is not represented yet in an individual of the next generation. As long as new and w.r.t. the definition above "successful" individuals can be created from the gene pool of the actual generation, the population size is allowed to grow. A potential offspring which is not able to fulfill these requirements is simply not considered for the gene pool of the next generation

The right part of Fig. 2 represents the gene pool of the alleles at a certain generation, say i whereas the left part aims to illustrate how this genetic can be used in order to generate a next population i of a certain size which may be smaller or larger than the actual population i depending on how successful the genetic manipulation operators crossover and mutation are in their above stated stated claim to produce new and successful chromosomes.

For a generic, stable and robust realization of the ideas stated above, some practical aspects have to be considered and implemented additionally:

Fig. 2. The left part of the figure represents the gene pool at generation i and the right part indicates the possible sice of generation i+1 which must not go below a minimum size and also not exceed an upper limit which are user defined parameters.

- The algorithm should offer different settings also for conventional parent selection. So the selection mechanisms for the two respective parents do not necessarily have to be the same. In many examples a combination of proportional (roulette-wheel) selection and random selection has already shown a lot of achievement potential (for example in combination with Genetic Programming [12]). It is also possible and reasonable with the algorithmic concepts described here to disable parent selection totally as scalable selection pressure comes along with the selection mechanisms after reproduction. This can be achieved by setting both parent selection operators to *Random*.
- The fact that reproduction results are only considered in case they are successful recombinations and eventually mutations of their parents chromosomes to use more than one crossover operator and more than one mutation operator at the same time. The reason for this possibility is given by the fact that only successful offspring chromosomes are considered for the ongoing evolutionary process which allows the application of crossover and mutation operators which do not produce good results mostly as long as they are still able to generate good offspring at least sometimes. On the one hand the insertion of such operators just increases the average selection pressure and therefore also the average running time, but on the other hand these operators can help a lot to broaden evolutionary search and therefore retard premature convergence. In case more than one crossover and mutation operator is allowed, the choice occurs by pure chance which has proven to produce better results than a preference of more successful operators.
- As indicated in Fig. 2, a lower as well as an upper limit of population size are still necessary in order to achieve efficient algorithmic performance. In case of a missing upper limit the population size would snowball especially in the first rounds which is inefficient. A lower limit of at least 2 individuals is also necessary as this indicates that it is no more possible to produce a sufficient amount of chromosomes that are able to outperform their own parents and therefore acts as a good detector for convergence.

- Depending on the problem at hand there may be several possibilities to fill up the next population with hopefully new individuals. If the problem representation allows an efficient check for structural identity it is recommendable to do this and accept new chromosomes as members for the next generation if there is no structurally identical individual yet. If a check for structural identity is not possible or too time-consuming there is still the possibility to assume two individuals as identical if they have the same fitness values as an approximative identity check.
- In order to terminate the run of a certain generation in case it is not possible to fill up the maximally allowed population size with new successful individuals, an upper limit of effort in terms of generated individuals is necessary. This maximum effort per generation is the maximum number of newly generated chromosomes per generation (no matter if it has been accepted or not).
- A further question is: When is a child chromosome better than its own parents? Is a child better, if it is better than the better of the two parents or is the child already better, if its better than the worse of the two parents? In order to answer this question, we have borrowed an aspect from Simulated Annealing: The Threshold fitness value that has to be outperform lies between the worse and the better parent and the user is able to adjust a lower starting value and a higher end value (*Comparison Factor Bounds*) where a *Comparison Factor Bound* value of 0.0 means that we consider the fitness of the worse parent and a *Comparison Factor Bound* of 1.0 means that we consider the better of the two parents. During the run of the algorithm, the *Comparison Factor Bound* is linearly scaled between the lower and the upper bound resulting in a broader search at the beginning and ending up with a more and more directed search at the end which picks up a basic idea of Simulated Annealing.

In order to inspect the consequences of the newly introduced algorithmic measures let us consider its effects when applying it to the 130-city TSP *ch130* and observe the allele frequencies over time as done for a standard GA in Section 2.

Fig. 3 shows the quality curve and the distribution of essential alleles for the new GA. When applying this GA with the new adaptive population sizing to the same benchmark test case as already treated in section 2, one can see that the global optimal solution is detected in only about 100 iterations. Nevertheless, the computational effort is comparable to the standard GA as much more individuals have to be evaluated at each iteration step due to the higher effort. Considering the distribution of essential alleles we see a totally different situation than in the case of the standard GA. Almost no essential alleles are lost and the ratio of essential alleles continuously increases in order to end up with a final population that contains almost all pieces of essential genetic information and therefore includes a very good solution. This shows that the essential alleles are preserved much more effectively and indicates that the influence of mutation is much smaller.

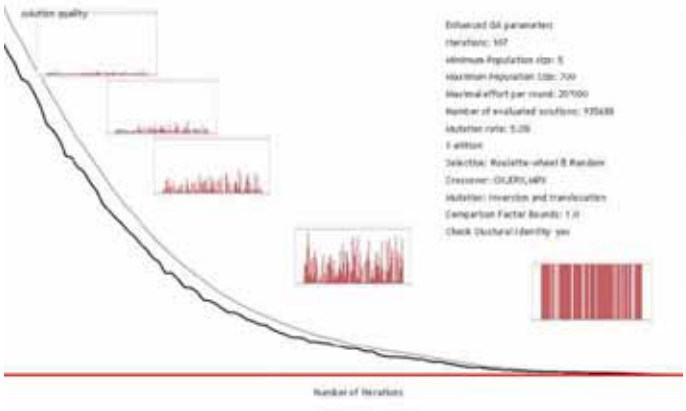

Fig. 3. The distribution of essential genetic information when using the enhanced GA with variable population sizing considering the ch130 benchmark TSP.

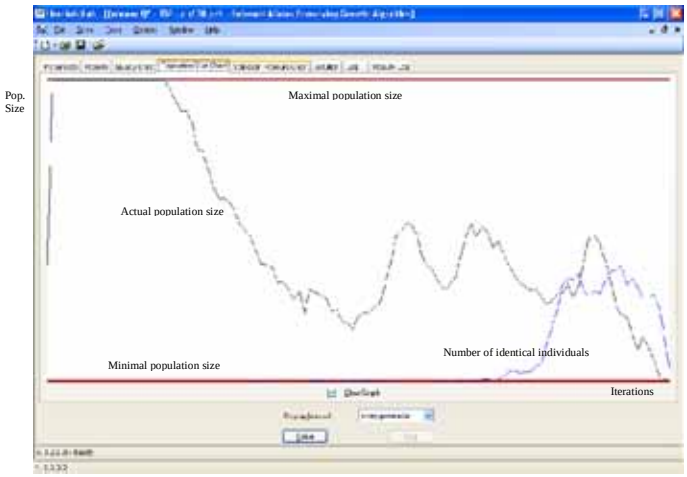

Fig. 4. Typical development of actual population size between the two borders (lower and upper limit of population size) displaying also the identical chromosomes that occur especially in the last iterations.

4 Conclusion and Future Perspectives

The main aim of the present contribution is to give an idea of the potential of the presented adaptive population sizing mechanism. More sophisticated test series have to be performed and documented on the basis of well-known benchmark problems of different areas representing diverse fitness landscapes in order to demonstrate the capabilities and robustness of the algorithm.

The general strategy described in this paper guarantees that evolution is preserved mainly with crossover results that were able to mix the properties of their parents in an advantageous way in a sense that **survival of the fittest alleles is rather supported than survival of the fittest chromosomes** which is a very essential aspect for the preservation of essential genetic information stored in many individuals (which may not be the fittest in the sense of individual fitness). By this strategy, the essential genetic information stored in the population is preserved effectively during the run of the algorithm.

Some aspects of adaptive population sizing as presented in this paper are quite similar to the concept of Offspring Selection [2] and therefore it might be a fruitful idea to think about a parallel implementation of RAP-GA in a similar sense as SASEGASA [1] which is the parallel implementation based upon Offspring Selection.

References

1. Affenzeller, M., Wagner, S.: SASEGASA: A New Generic Parallel Evolutionary Algorithm for Achieving Highest Quality Results. Journal of Heuristics, Special Issue on New Advances on Parallel Meta-Heuristics for Complex Problems, 10(3) (2004) 239–263
2. Affenzeller, M., Wagner, S.: Offspring Selection: A New Self-Adaptive Selection Scheme for Genetic Algorithms. Adaptive and Natural Computing Algorithms (2005) 218–221
3. Arabas, J., Michalewicz, Z., Mulawka, J.: GAVaPS – A Genetic Algorithm with Varying Population Size. Proceedings of the First IEEE Conference on Evolutionary Computation (1994) 73–78
4. Baeck, T., Eiben, A.E., van der Vaart, N.A.L.: An Empirical Study on GAs "Without Parameters". Proceedings of the 6th Conference on Parallel Problem Solving from Nature, Springer LNCS 1917 (2000) 315–324
5. Beyer, H.G.: The Theory of Evolution Strategies. Springer-Verlag Berlin Heidelberg NewYork (2001)
6. Eiben, A.E., Marchiori, E., Valk, V.A.: Evolutionary Algorithms with On-the-Fly Population Size Adjustment. Proceedings of the 8th Conference on Parallel Problem Solving from Nature, Springer LNCS 3242 (2004) 41–50
7. Fogel, D.: An Introduction to Simulated Evolutionary Optimization. IEEE Trans. on Neural Networks, 5(1) (1994) 3–14
8. Goldberg, D.E.: Sizing Populations for Serial and Parallel Genetic Algorithms. Proceedings of the 3rd International Conference on Genetic Algorithms (1989) 70–79
9. Reinelt, G.: TSPLIB - A Traveling Salesman Problem Library. ORSA Journal on Computing, Vol. 3 (1991) 376–384
10. Smith, R., Forrest, S., Perelson, A.: Population Diversity in an Immune System Model: Implications for Genetic Search. Foundations of Genetic Algorithms, Vol. 2 (1993) 153–166
11. Yoshida, Y., Adachi, N.: A Diploid Genetic Algorithm for Preserving Population Diversity - Pseudo-Meiosis GA. Lecture Notes in Computer Science, 866 (1994) 36–45
12. Winkler, S., Affenzeller, M., Wagner, S.: Advanced Genetic Programming Based Machine Learning. Journal of Mathematical Modelling and Algorithms (2007)