A New Approach to Evolutionary Computation: Segregative Genetic Algorithms (SEGA)

Michael Affenzeller

Institute of Systems Science
Systems Theory and Information Technology
Johannes Kepler University
Altenbergerstrasse 69
A-4040 Linz - Austria
ma@cast.uni-linz.ac.at

Abstract. This paper looks upon the standard genetic algorithm as an artificial self-organizing process. With the purpose to provide concepts that make the algorithm more open for scalability on the one hand, and that fight premature convergence on the other hand, this paper presents two extensions of the standard genetic algorithm without introducing any problem specific knowledge, as done in many problem specific heuristics on the basis of genetic algorithms. In contrast to contributions in the field of genetic algorithms that introduce new coding standards and operators for certain problems, the introduced approach should be considered as a heuristic appliable to multiple problems of combinatorial optimization, using exactly the same coding standards and operators for crossover and mutation, as done when treating a certain problem with a standard genetic algorithm. The additional aspects introduced within the scope of segregativ egenetic algorithms (SEGA) are inspired from optimization as well as from the views of bionics. In the present paper the new algorithm is discussed for the travelling salesman problem (TSP) as a well documented instance of a multimodal combinatorial optimization problem. In contrast to all other evolutionary heuristics that do not use any additional problem specific knowledge, we obtain solutions close to the best known solution for all considered benchmark problems (symmetric as well as asymmetric benchmarks) which represents a new attainment when applying evolutionary computation to the TSP.

1 Introduction

A genetic algorithm (GA) may be described as a mechanism that imitates the genetic evolution of a species. The underlying principles of GAs were first presented by Holland [5]. An overview about GAs and their implementation in various fields was given by Goldberg [4] or Michalewicz [7].
Evolutionary Strategies, the second major representative of evolutionary computation, were introduced by Rechenberg [8]. Applied to problems of combinatorial optimization, evolutionary strategies tend to find local optima quite efficiently. But in the case of multimodal test functions, global optima can only be detected

J. Mira and A. Prieto (Eds.): IWANN 2001, LNCS 2084, pp. 594-601, 2001.

by evolutionary strategies if one of the start values is located in the narrower range of a global optimum. Nevertheless, the concept how evolutionary strategies handle the selective pressure has turned out to be very useful in the context of the segregative genetic algorithm (SEGA) presented in this paper.
Furthermore, we have borrowed the cooling mechanism from simulated annealing (SA), introduced by Kirkpatrick [6] in order to obtain a variable selective pressure. This will mainly be needed when segregating and reunifying the population as described in subsection 2.2.
The aim of dividing the whole population into a certain number of subpopulations (segregation) that grow together in case of stagnating fitness within those subpopulations is to combat premature convergence which is the source of GA-difficulties. This segregation and reunification approach is a new idea to overcome premature convergence. The principle idea is to divide the whole population into a certain number of subpopulations at the beginning of the evolutionary process. These subpopulations evolve independently from each other until the fitness increase stagnates because of too similar individuals within the subpopulations. Then a reunification from n to (n-1) subpopulations is done. By this approach of width-search, building blocks in different regions of the search space are evolved at the beginning and during the evolutionary process which would disappear early in case of standard genetic algorithms and whose genetic information could not be provided at a later date of evolution when the search for global optima is of paramount importance. In this context the above mentioned variable selective pressure is especially important at the time of joining some residents of another village to a certain village in order to steer the genetic diversity.
Experimental results on some symmetric and asymmetric benchmark problems of the TSP indicate the supremacy of the introduced concept for locating global minima compared to a standard genetic algorithm. Furthermore, the evaluation shows, that the results of the segregative genetic algorithm (SEGA) are comparable for symmetric benchmark problems and even superior for asymmetric benchmark problems when being compared to the results of the cooperative simulated annealing technique (COSA) [13] which has to be considered as a problem specific heuristic for routing problems. This represents a major difference to SEGA that uses exactly the same operators as a corresponding GA and can, therefore, be applied to a huge number of problems - namely all problems GAs can be applied to.
Moreover it should be pointed out that a corresponding GA is unrestricted included in the SEGA when the number of subpopulations (villages) and the cooling temperature are both set to 1 at the beginning of the evolutionary process.

The rest of the paper is organized as follows: In section 2, we first introduce the concept of a variable selective pressure and the concept of segregation and reunification in detail to end up with the SEGA. In section 3, we discuss the performance of SEGA compared to GA and COSA based on standard benchmarks of the TSP. Finally, section 4 summarizes the main results of this contribution.

2 Introducing the New Concepts

In principle, the new SEGA introduces two enhancements to the basic concept of genetic algorithms. The first is to bring in a variable selective pressure, as described in subsection 2.1, in order to control the diversity of the evolving population. The second concept introduces a separation of the population to increase the broadness of the search process and joins the subpopulation after their evolution in order to end up with a population including all genetic information sufficient for locating the region of a global optimum (subsection 2.2). The incorporation of that measures with a usual genetic algorithm is schematically illustrated in subsection 2.3.

2.1 Variable Selective Pressure

The handling of selective pressure in our context is mainly motivated by evolutionary strategies where μ parents produce λ descendants from which the best μ survive. Within the framework of evolutionary strategies, the selective pressure is defined as $s = \frac{\mu}{\lambda}$, where a small value of s indicates a high selective pressure and vice versa (for details see for instance [12]). Applied to the new genetic algorithm this means that from $|POP|$ (population size) number of parents $|POP| * T$ ((size of virtual population) $> |POP|$, i.e. T>1) descendants are generated by crossover and mutation from which the best $|POP|$ survive. Obviously we define the selective pressure as $s = \frac{|POP|}{|POP|*T} = \frac{1}{T}$ where a small value of s, i.e. a great value of T stands for a high selective pressure. When processing SEGA, the temperature slowly decreases from a certain value down to 1 in the evolution process of each subpopulation.

2.2 Segregation and Reunification

It appears that the GA prematurely converges to very different regions of the solution space when repeatedly running a GA. Moreover it is known from GA-theory that depending on the problem-type and the problem-dimension there is a certain population size, where exceeding this population size doesn't effect any more improvements in the quality of the solution.
Motivated by that observations, we have developed an extended approach to genetic algorithms where the total population is split into a certain number of subpopulations or villages, all evolving independently from each other (segregation) until a certain stage of stagnation in the fitness of those subpopulations is reached. Then, in order to bring some new genetic information into each village, the number of villages is reduced by one which causes new overlapping-points of the villages. Fig. 1 shows a schematic diagram of the described process. This process is repeated until all villages are growing together ending up in one town (reunification). The variable selective pressure (subsection 2.1) is of particular importance, if the number of subpopulations is reduced by one because this event brings new diversification into the population. In this case a higher selective pressure is reasonable, i.e. if reunifying members of neighbouring villages,

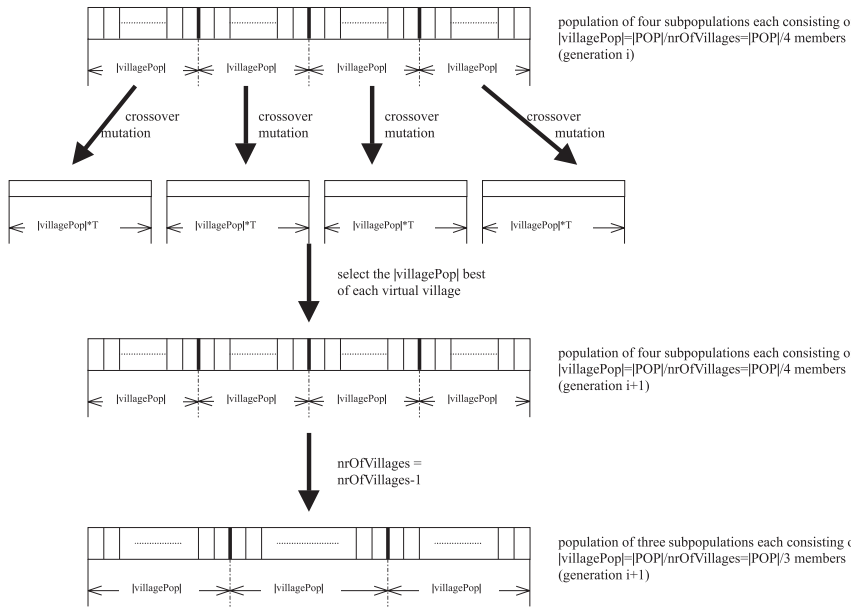

Fig. 1. Evolution of a new population for the instance that four subpopulations are merged to three.

the temperature is reset to a higher lever in order to cool down to 1 as the new system of subpopulations evolves. As the number of villages decreases during evolution, it is recommended to reset the selective pressure to a higher level, because the genetic diversity of the emerging greater subpopulations is growing. For fighting premature convergence, the main advantage of the described strategy is, that building-blocks in different regions of the search space are evolved independently from each other at the beginning of the evolutionary process. The aim is that the best building-blocks survive during the recombination phase, yielding in a final population (if the number of villages is 1) containing all essential building-blocks for the detection of a global optimum. In case of ordinary GAs building blocks that disappear early and which may be important at a later stage of the evolutionary process, when the search for global optima is of paramount importance, can hardly ever be reproduced (premature convergence).

2.3 The New Algorithm

With all strategies described above, finally the new genetic algorithm is stated as follows:

The segregative genetic algorithm

```
procedure SEGA
```

```
initialize population size|POP|
initialize number of iterations nrOfIterations ∈ ℕ
initialize number of villages nrOfVillages ∈ ℕ
initialize temperature T* ∈ ℝ+
initialize adaptive cooling factor α ∈ [0, 1]
generate initial population POP0 = (I1, ...., I|POP|)
calculate dates of reunification
for i:=1 to nrOfIterations do
  if (i = dateOfReunification) then
     nrOfVillages:=nrOfVillages -1
     reset temperature
  end if
  POPi:=calcNextGeneration(POPi−1, T, nrOfVillages, |POP|)
  T:=T* * α
next i
```

Function "calcNextGeneration" implements the evolution of the next generation of subpopulations.

```
function calcNextGeneration: (POPi−1, T, nrOfVillages, |POP|)
  villagePopulation=|POP| / nrOfVillages
  for i:=(0 to (nrOfVillages-1)) do
    for j:=(i*villagePopulation) to ((i+1)*villagePopulation) do
      calculate fitnessj of each member of the village population
      (like in standard GA).
    next j
    |virtualPopulation|=|villagePop|*T
    for k:=0 to |villagePopulation| do
      generate individuals of virtual population Ik ∈ S
      from the members of the village.
      I(i*|villagePopulation|)...I((i+1)*|villagePopulation|) ∈ POPi−1
      due to their fitnesses by crossover and mutation
    next k
    select the best |villagePopulation| from the virtual population
    in order to achieve the village population
    I(i*|villagePopulation|)...I((i+1)*|villagePopulation|) ∈ POPi
    of the next generation.
  next i
```

SEGA, as described above, uses a fixed number of iterations for termination. Depending on this total number of iterations and the initial number of subpopulations (villages), the dates of reunification are calculated at the beginning of the evolutionary process. Further improvements, particularly in the sense of running time, are possible, if, in order to determine the dates of reunification,

a dynamic criterion for the detection of stagnating genetic diversity within the subpopulations is used. A further aspect worth mentioning is the choice of the temperature and the cooling-factor when merging certain subpopulations. Experimental research has indicated that the best results can be achieved resetting the temperature within a range of 0.2 to 2.0 and choosing a cooling-factor α such that the selective pressure converges to one as the subpopulation's genetic diversity is stagnating. Convenient choice of the GA specific parameters can be found in [1], [2], or [3].

3 Experimental Results

In our experiment, all computations are performed on a Pentium III PC with 256 megabytes of main memory. The programs are written in the Java programming language.
We have tested SEGA on a selection of symmetric as well as asymmetric TSP benchmark problem instances taken from the TSPLIB [9] using updated results for the best, or at least the best known, solutions taken from [10]. In doing so, we have performed a comparison of SEGA with a GA using exactly the same operators for crossover and mutation and the same parameter settings and with the COSA-algorithm as an established and successful ambassador of a heuristic especially developed for routing problems.
Fig. 2 shows the experimental results for the problem ch130 (130 city problem) as an example of a symmetric TSP benchmark. This example demonstrates the

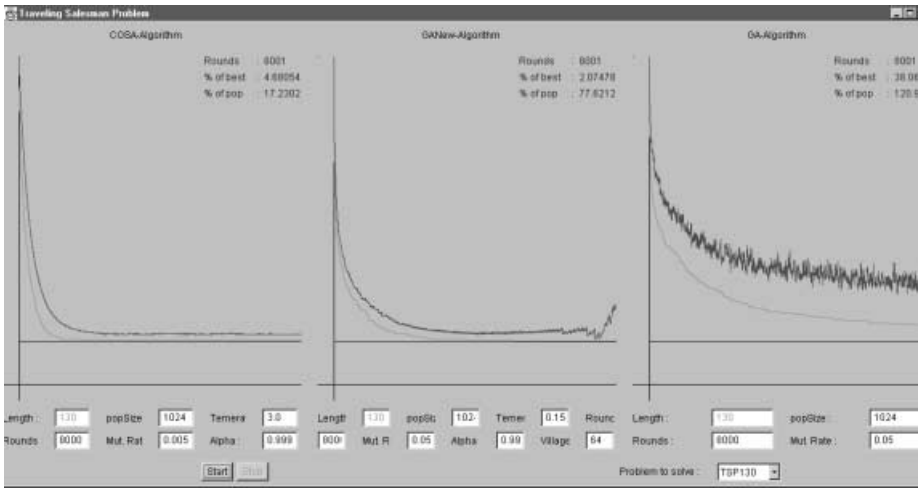

Fig. 2. Comparison of COSA, SEGA, and GA on the basis of the ch130 benchmark problem: For each algorithm, the average fitness and the fitness of the best member of the population is diagrammed relatively to the best known solution represented by the horizontal line.

predominance of the new SEGA compared to the standard-GA. Moreover it even shows the competitiveness of SEGA when compared to the problem specific COSA heuristic. The preeminence of SEGA becomes even more evident, if asymmetric benchmark problems are considered. For the tests the parameters of COSA are set as suggested by the author in [13]. Both, GA and SEGA use a mutation probability of 0.05 and a combination of OX-crossover [7] and ERX-crossover [11] combined with the golden-cage population model (e.g. [13]), i.e. the entire population is replaced with the exception that the best member of the old population survives until the new population generates a better one (wild-card strategy). Within SEGA, the described strategies are applied to each subpopulation. The results of a test presented in the present paper start with 64 villages (subpopulations), each consisting of 8 individuals, i.e. the total population size is set to 1024 for SEGA (as well as for COSA and GA).
Table 1 shows the the experimental results of SEGA, COSA, and GA concerning various types of problems in the TSPLIB. For each problem, the algorithms were run tentimes. The efficiency for each algorithm is quantified in terms of the relative difference of the best's individual fitness after a given number or iterations to the best or best-known solution. In this experiment, the relative difference is defined as $\text{relativeDifference} = (\frac{Fitness}{Optimal} - 1) * 100\%$.

Table 1. Experimental results of COSA, SEGA and GA.

Problem	Number of Iterations	Average difference(%)		
		COSA	SEGA	GA
eil76 (symmetric)	2000	3.22	1.55	11.21
ch130 (symmetric)	8000	4.76	1.84	35.44
kroA150 (symmetric)	8000	7.90	2.21	40.97
kroA200 (symmetric)	10000	8.54	5.21	45.11
br17 (asymmetric)	100	0.00	0.00	0.00
ftv55 (asymmetric)	2000	44.22	0.76	33.92
kro124p (asymmetric)	10000	26.78	2.61	37.49
ftv170 (asymmetric)	15000	202.33	4.13	131.61

4 Conclusion

The two main changes having been made within the segregative genetic algorithm (SEGA), the introduction of a variable selective pressure and the segregation of the population, have a direct influence on two of the major evolution criteria: speed and convergence. First of all, a far better convergence is noticeable for all tested problems, which is the primary objective of improving the algorithm. Concerning the speed of SEGA, it has to be pointed out that the superior performance concerning convergence requires a higher running time,

mainly because of the the greater population size $|POP|$ required. This should allow to transfer already developed GA-concepts to increasingly powerful computer systems in order to achieve better results. Using simultaneous computers seems especially suited to increase the performance of SEGA. Anyway, the corresponding GA is fully included within SEGA if starting with one subpopulation (village) and a temperature constantly set to one, achieving a performance only marginally worse than the performance of the equivalent GA. In other words, SEGAs can be interpreted as a superstructure to GA or as a technique upwards compatible to GAs. Therefore, an implementation of SEGA for a certain problem should be quite easy to do, presumed that the corresponding GA (coding, operators) is known. Even though, because of better comparability, no additional hybrid techniques like commonly used hill-climbing or certain other pre- or post-optimization techniques have been considered in the examples presented in this paper, there absolutely exists no objection of doing so in order to improve the convergence of SEGA.

References

1. Chambers, L. (ed.): Practical Handbook of Genetic Algorithms Volume I. CRC Press. (1995)
2. Chambers, L. (ed.): Practical Handbook of Genetic Algorithms Volume II. CRC Press. (1995)
3. Chambers, L. (ed.): Practical Handbook of Genetic Algorithms Volume III. CRC Press. (1998)
4. Goldberg, D. E.: Genetic Alogorithms in Search, Optimization and Machine Learning. Addison Wesley Longman (1989)
5. Holland, J. H.: Adaption in Natural and Artificial Systems. 1st MIT Press ed. (1992)
6. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science 220 (1983) 671–680
7. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996)
8. Rechenberg, I.: Evolutionsstrategie. Friedrich Frommann Verlag (1973)
9. Reinelt, G.: TSPLIB - A Traveling Salesman Problem Library. ORSA Journal on Computing 3 (1991) 376-384
10. Reinelt, G.: TSPLIB - A Traveling Salesman Problem Library. ftp://ftp.zib.de/pub/Packages/mp-testdata/tsp/tsplib/index.html (1997)
11. Schaffer J.: Proceedings of the Third International Conference od Genetic Algorithms. San Mateo (1989) 133–140
12. Schneburg, E., Heinzmann, F., Feddersen, S.: Genetische Algorithmen und Evolutionsstrategien. Addison-Wesley (1994)
13. Wendt, O.: Tourenplanung durch Einsatz naturanaloger Verfahren. Deutscher Universitätsverlag (1995)