New Variants of Genetic Algorithms Applied to Problems of Combinatorial Optimization

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

To be presented in symposium **B**

Abstract. Problems of Combinatorial Optimization distinguish themselves by their well-structured problem description as well as by their huge number of possible action alternatives. Especially in the area of production and operational logistics these problems frequently occur. Their advantage lies in their subjective understanding of action alternatives and their objective functions. The use of classical optimization methods for problems of combinatorial optimization often fails because of the exponentially growing computational effort. Therefore, even if they are not able to ensure a global solution, heuristic methods like Genetic Algorithms (GAs) or Evolution Strategies (ESs) are massively utilized in practice because of their significant lower computational effort. Both, GAs and ESs have a number of drawbacks that reduce their applicability to that kind of problems.

During the last decades plenty of work has been investigated in order to introduce new coding standards and operators especially for Genetic Algorithms. All these approaches have one thing in common: They are rather problem specific and often they do not challenge the basic principle of Genetic Algorithms. In the present paper we take a different approach and look upon the concepts of a Standard Genetic Algorithm (SGA) as an artificial self organizing process in order to overcome some of the fundamental problems Genetic Algorithms are concerned with in almost all areas of application.

With the purpose of providing concepts which make the algorithm more open for scalability on the one hand, and which fight premature convergence on the other hand, this paper presents an extension of the SGA that does not introduce any problem specific knowledge: On the basis of an Evolution-Strategy-like selective pressure handling a concept of dynamically dealing with multiple crossover operators in parallel is introduced. 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 novel 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 SGA. Furthermore, the corresponding Genetic Algorithm is unrestrictedly included in all of the newly proposed hybrid variants under especial parameter settings.

In the present paper the new algorithm is discussed for the Traveling Salesman Problem (TSP) as a well documented instance of a multimodal combinatorial optimization problem. Even if we did not presuppose any information about the quality of the involved operators we were able to achieve results superior to the results obtained with a corresponding Genetic Algorithm for all considered benchmark problems and operators.

1 Introduction

Work on what nowadays is called Evolutionary Computation started in the sixties in the United States and Germany. Substantially there are two basic approaches in computer science that copy evolutionary mechanisms: Evolution Strategies(ES) [Rec73], [Sch94] and Genetic Algorithms(GA)[Hol92]. Despite many things in common the two attempts developed almost independently from each other in the USA (GA) and Germany (ES).
The major differences between Evolution Strategies and Genetic Algorithms lie in the form of the genotype, the calculation of the fitness and the operators (mutation, recombination, selection). In contrast to Genetic Algorithms where the main-role of the mutation operator is simply to avoid stagnation, mutation is the primary operator of Evolution Strategies. As a further difference between the two major representatives of Evolutionary Computation, selection in case of Evolution Strategies is absolutely deterministic which is not the case in the context of Genetic Algorithms or in nature. Therefore, in case of Evolution Strategies, arbitrary small differences in fitness can decide about the survival of an individual. The mapping between a bit-string and the real numbers often is a problem when applying Genetic Algorithms whereas Evolution Strategies resign in copying nature that exactly.
Applied to problems of Combinatorial Optimization, Evolution Strategies tend to find local optima quite efficiently. But in the case of multimodal test functions, global optima can only be detected by Evolution Strategies if one of the start values is located in the narrower range of a global optimum. Nevertheless, the concept how Evolution Strategies handle the selective pressure has turned out to be very useful in the context of the new hybrid Genetic Algorithm presented in this paper.
Furthermore, we have borrowed the cooling mechanism from Simulated Annealing (SA), introduced by Kirkpatrick [KGV83], in order to obtain a variable selective pressure for the enhanced GA-model as described in section 2.
The second newly introduced concept allows the dynamic usage of multiple crossover operators in parallel in order to somehow imitate the parallel evolution of a variety of species that are struggling for limited resources. This strategy seems very adopted for problems which consider more than one crossover operator - especially if the properties of the considered operators may change as evolution proceeds.
As an important property of all the newly introduced hybrids it should be pointed out that under special parameter settings the corresponding GA/GAs is/are unrestrictedly included in the new hybrids.

2 The Variable Selective Pressure Model

> "It is a matter of fact that in Europe Evolution Strategies and in the USA Genetic Algorithms have survived more than a decade of non-acceptance or neglect. It is also

true, however, that so far both strata of ideas have evolved in geographic isolation and thus not led to recombined offspring. Now it is time for a new generation of algorithms which make use of the rich gene pool of ideas on both sides of Atlantic, and make use too of the favorable environment showing up in the form of massively parallel processor systems[...]. There are not many paradigms so far to help us make good use of the new situation. We have become too familiar with monocausal thinking and linear programming, and there is now a lack of good ideas for using massively parallel computing facilities."[1]

2.1 Introduction

The handling of selective pressure in the context of Genetic Algorithms mainly depends on the choice of a certain replacement scheme. The 'generational replacement scheme', for example, replaces the entire population by the next one, whereas the 'elitism replacement scheme', which keeps the best individuals of the last generation and only replaces the rest, usually performs faster. On the other hand, elitism might cause too homogeneous populations, i.e. little population diversity, and therefore might cause unwanted premature convergence. Anyway, there exists no manageable model for controllable selective pressure handling within the theory of Genetic Algorithms. Therefore, we introduce some kind of intermediate step (a 'virtual population') into selection which provides a handling of selective pressure very similar to that of Evolution Strategies [Aff01]. As we will exemplarily point out, the most common replacement mechanisms can easily be implemented in this intermediate selection step. Furthermore, this Evolution Strategy like variable selective pressure will help us to steer the degree of population diversity on the one hand and, on the other hand, it will act as a basic model for a new hybrid metaheuristics based upon Genetic Algorithms as being proposed in section 3.
Actually, all modifications that are and will be taken into account use exactly the same operators for crossover and mutation as a corresponding Genetic Algorithm. As no further problem specific information is used, the new hybrids can be applied to a huge number of problems - namely all problems Genetic Algorithms can be applied to. Moreover it should be pointed out that under especial parameter settings, the corresponding Genetic Algorithm is unrestrictedly included in the presented hybrid variants of a Genetic Algorithm.

2.2 Formal Integration of the Variable Selective Pressure Model

Similar to any other commonly used Genetic Algorithm we use a population of fixed size that will evolve to a new population of same size by selection, crossover, and mutation.

[1] Taken from the introduction to the proceedings of the First Workshop on Parallel Problem Solving from Nature [SM90].

What we additionally have done is to introduce an intermediate step in terms of a so-called virtual population of variable size where the size of the virtual population usually has to be greater than the population size. This virtual population is created by selection, crossover, and mutation in the common sense of Genetic Algorithms. But like in the context of Evolution Strategies, only a certain percentage of this intermediate population will survive which represents a direct analogy to selective pressure handling in the context of Evolution Strategies.
This handling of selective pressure in our context is mainly motivated by (μ, λ)-Evolution Strategies where μ parents produce λ descendants from which the best μ survive. Within the framework of Evolution Strategies, 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 [SHF94]). Applied to Genetic Algorithms, this means that from $|POP|$ (population size) number of parents $|POP| \cdot T$ ((size of virtual population) $> |POP|$, i.e. $T > 1$)[2] descendants are generated by crossover and mutation from which the best $|POP|$ survive as illustrated in Fig. 1.

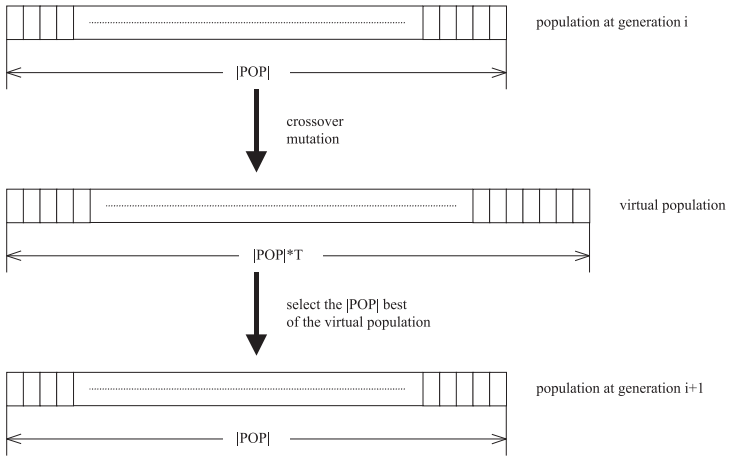

Fig. 1. Evolution of a new population with selective pressure $s = \frac{1}{T}$ for a virtual population built up in the sense of a (μ, λ)-Evolution Strategy.

Obviously we define the selective pressure as $s = \frac{|POP|}{|POP| \cdot T} = \frac{1}{T}$, where a small value of s, i.e. a great value of T stands for a high selective pressure and vice versa. Equipped with this enhanced GA-model it is quite easy to adopt further extensions based upon a controllable selective pressure, i.e. it becomes possible either to reset the temperature up/down to a certain level or simply to cool down the temperature in the sense of Simulated An-

[2] In principle, also temperatures T less than 1 are imaginable in case of too homogeneous populations in order to replenish the rest of the population $((1-T)|POP|)$ with 'new' individuals for fighting premature convergence.

nealing during the evolutionary process in order to steer the convergence of the algorithm. Bionically interpreting this (μ, λ)-Evolution Strategy like selective pressure handling, for Genetic Algorithms this means, that some kind of 'infant mortality' has been introduced in that sense that a certain ratio of the population ($|POP| \cdot T - |POP| = |POP| \cdot (T-1)$) will never become procreative, i.e. this weaker part of a population will not get the possibility of reproduction. Decreasing this 'infant mortality', i.e. reducing the selective pressure during the evolutionary process also makes sense in a bionic interpretation as also in nature stronger and higher developed populations suffer less from infant mortality. From the point of view of optimization, decreasing the temperature during the optimization process means that a greater part of the search space is explored at the beginning of evolution - whereas at a later stage of evolution, when the average fitness is already quite high, a higher selective pressure is quite critical in that sense that it can easily cause premature convergence. Anyway, operating with a temperature converging to zero, this (μ, λ)-Evolution Strategy like selective pressure model for Genetic Algorithms acts like the corresponding Genetic Algorithm with generational replacement. Moreover, implementing

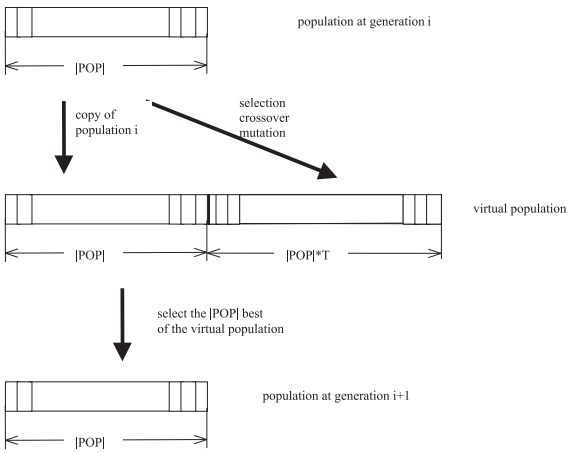

Fig. 2. Evolution of a new population for a virtual population built up in the sense of a $(\mu+\lambda)$-Evolution Strategy.

the analogue to the $(\mu+\lambda)$-Evolution Strategy denotes the other extreme of immortal individuals. However, also the implementation of this strategy is quite easy with our model as indicated in Fig. 2. Other replacement mechanisms, like elitism or the goldcage-model for example, are also very easy to handle by just adding the best individuals respectively the best individual of the last generation to the virtual generation.
In the following section we are going to discuss new aspects and models built up upon the described variable selective pressure model. Nevertheless, there is nothing against using a Genetic Algorithm equipped with the variable selective pressure model without further

extensions. Experiments performed on the variable selective pressure model already indicate the supremacy of this approach. In those experiments the temperature is slowly cooled down to zero. If one is aware of an appropriate measure of population diversity, the temperature can be set dynamically in order to steer the the genetic diversity which seems to represent a further promising approach.

3 The Dynamic Usage of Multiple Crossover Operators in Parallel

Genetic Algorithms as well as its most common variants consider the evolution of a single species, i.e. crossover can be done between all members of the population. This supports the aspect of depth-search but not the aspect of width-search. Considering natural evolution, where a multiplicity of species evolve in parallel, as a role model, we could introduce a number of crossover operators and apply each one to a certain subpopulation. In order to keep that model realistically it is necessary to choose the size of those subpopulations dynamically, i.e. depending on the actual success of a certain species its living space is expanded or restricted. Speaking in the words of Genetic Algorithms, this means that the size of subpopulations (defined by the used crossover and mutation operators) with lower success in the sense of the quality function is restricted in support of those subpopulations that push the process of evolution. But as no Genetic Algorithm known to the author is able to model jumps in the evolutionary process and no exchange of information between the subpopulations takes place, the proposed strategy would fail in generating results superior to the results obtained when running the Genetic Algorithms with the certain operators one after another. Thus, the achieved profits would 'only' concern the performance of the algorithm.
Therefore, it seems reasonable to allow also recombination of individuals that have emerged from different crossover operators, i.e. the total population is taken into account for each crossover operator and the living space (habitat) of each virtual subpopulation is defined by its success during the last iterations as illustrated in Fig. 3. Strictly interpreted, the term 'Multi-Creature Genetic Algorithms' might be too strong for the construction shown in Fig. 3 and we might better term this extension as a Genetic Algorithm that dynamically uses multiple crossover operators in parallel.
Exemplarily considering the properties of the OX and the ERX operators for crossover it is reported (e.g. in citewhitley89 or [Mic96]) that the OX-operator outperforms the ERX-operator in terms of speed by a factor of about 20 whereas the ERX-operator surpasses OX in terms of global convergence. Dynamically using multiple crossover operators in parallel utilizes the 'fast' OX-operator for a long evolution period until the performance in terms of solution quality of ERX significantly outperforms OX at a later stage of evolution.

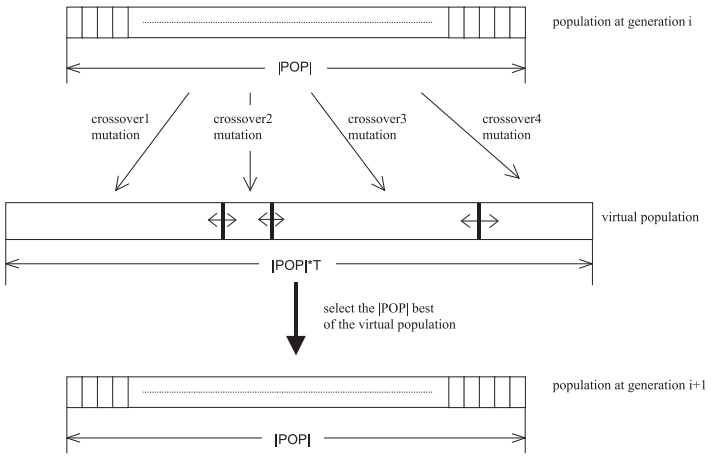

Fig. 3. Evolution of a new population for the instance that four crossover operators are used in parallel.

Anyway, this dynamic (self-organizing) strategy seems particularly suitable for situations where a couple of crossover operators whose properties are not exactly known are taken into account.

4 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 the new concepts on a selection of symmetric as well as asymmetric TSP benchmark problem instances taken from the TSPLIB [Rei91] using updated results[3] for the best, or at least the best known, solutions. In doing so, we have performed a comparison of the new hybrid 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. 4 shows the experimental results for the problem eil76 (76 city problem) as an example of a symmetric TSP benchmark. This example demonstrates the predominance of the new GA compared to the standard-GA. Moreover it even shows the competitiveness of our new hybrid when compared to the problem specific COSA heuristic. The preeminence 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 [WK97]. Both, GA and the new GA use a mutation probability of 0.05 combined with the golden-cage population model (e.g. [Mic96]), i.e. the entire population is replaced with the exception

[3] Updates for the best (known) solutions can for example be found on ftp://ftp.zib.de/pub/Packages/mp-testdata/tsp/tsplib/index.html

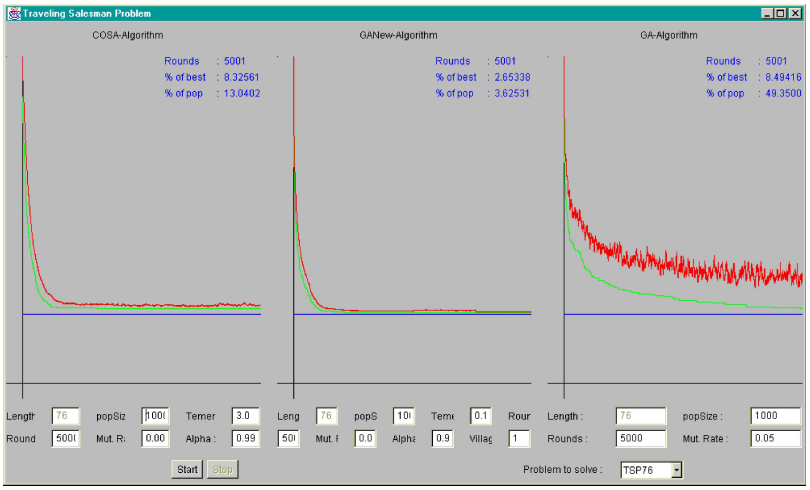

Fig. 4. Comparison of COSA, GA_{ERX}, and the new GA on the basis of the eil76 benchmark problem: For each algorithm, the average fitness and the fitness of the best member of the population is diagrammed relatively to the best known solution represented by the horizontal line.

that the best member of the old population survives until the new population generates a better one (wild-card strategy).

Table 1 shows the experimental results of COSA, GA with OX respectively ERX for crossover, and with the new GA that dynamically uses a combination of OX and ERX concerning various types of problems in the TSPLIB. For each problem, the algorithms were run three times. 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 $relativeDifference = (\frac{Fitness}{Optimal} - 1) * 100\%$.

Table 1. Experimental results of COSA, GA (using OX or ERX for crossover) and the new GA with a dynamic combination of OX- and ERX crossover.

Problem	Number of Iterations	Average difference(%)			
		COSA	GA_{OX}	GA_{ERX}	$NewGA_{var.}$
eil76 (symmetric)	5000	6.36	17.56	7.62	2.54
ch130 (symmetric)	5000	14.76	84.54	32.44	6.42
kroA150 (symmetric)	5000	20.91	102.40	71.97	11.07
kroA200 (symmetric)	10000	48.45	95.69	117.11	18.47
br17 (asymmetric)	200	0.00	0.00	0.00	0.00
ftv55 (asymmetric)	5000	44.22	41.34	23.52	3.14
kro124p (asymmetric)	10000	26.78	30.61	15.49	3.26
ftv170 (asymmetric)	15000	187.34	87.12	126.22	17.59

5 Conclusion and Future Perspectives

In this paper an enhanced Genetic Algorithm and two upgrades have been presented and exemplarily tested on some TSP benchmarks. The proposed GA-based techniques couple aspects from Evolution Strategies (selective pressure), Simulated Annealing (temperature, cooling) as well as a special strategy fur using more than one crossover operator in parallel with crossover, mutation, and selection in a general way, so that established crossover and mutation operators for certain problems may be used analogously to the corresponding Genetic Algorithm. The investigations in this paper have mainly focused on the avoidance of premature convergence and on the introduction of methods which make the algorithm more open for scalability in the sense of convergence versus running time.
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 the experimental results section, there absolutely exists no objection of doing so in order to improve the convergence.
It is believed by the author that a main area of application for the new hybrids is given by applications that are already treated by Genetic Algorithms:
Independent from the actually employed variant of a Genetic Algorithm it should be possible in almost all cases to implement the additional concepts of a variable selective pressure and that of parallel operators (presumed that more than one operator is known for a certain problem) quite easily.
Ongoing research indicates further possible extensions to the methods described in the present thesis:
The efficiency of a variable selective pressure certainly depends on the genetic diversity of the entire population and therefore, it could be a very fruitful approach to define the actual selective pressure depending on the actual genetic diversity of the population. Thus, assuming one is aware of an appropriate measure for population diversity, it would be possible to steer the temperature, i.e. the selective pressure, depending on the actual value of genetic diversity within the population. This would primarily relieve the handling of the algorithm because the quite tricky locating of suited annealing parameters (temperature T, cooling factor α) would not be necessary any more.
Even though some attempts have been undertaken in order to measure the degree of population diversity ([MKB94], [SFP93], [YA94]) we have not utilized these methods as they are all problem specific or unprecise in some way.

References

[Aff01] M. Affenzeller. Transferring the concept of selective pressure from evolutionary strategies to genetic algorithms. *Proceedings of the 14th International Conference on Systems Science*, 2:346–353, 2001.

[Hol92] J.H. Holland. *Adaption in Natural and Artificial Systems*. MIT Press, 1992.

[KGV83] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[Mic96] Z. Michaliwicz. *Genetic Algorithms + Data Structurs = Evolution Programs*. Springer-Verlag Berlin Heidelberg New York, 3. edition, 1996.

[MKB94] D.C. Mattfeld, H. Kopfer, and C. Bierwirth. Control of parallel population dynamics by social-like behavior of GA-Individuals. *Lecture Notes in Computer Science*, 866:16–25, 1994.

[Rec73] I. Rechenberg. *Evolutionsstrategie*. Friedrich Frommann Verlag, 1973.

[Rei91] G. Reinelt. TSPLIB - A traveling salesman problem library. *ORSA Journal on Computing 3*, 3:376–384, 1991.

[Sch94] H.P. Schwefel. *Numerische Optimierung von Computer-Modellen mittels Evolutionsstrategie*. Birkhäuser Verlag. Basel, 1994.

[SFP93] R.E. Smith, S. Forrest, and A.S. Perelson. Population diversity in an immune system model: Implications for genetic search. *Foundations of Genetic Algorithms*, 2:153–166, 1993.

[SHF94] E. Schoneburg, F. Heinzmann, and S. Feddersen. *Genetische Algorithmen und Evolutionsstrategien*. Addison-Wesley, 1994.

[SM90] H.-P. Schwefel and R. Männer(editors). *Proceedings of the First International Conference on Parallel Problem Solving from Nature (PPSN)*. Springer-Verlag Berlin Heidelberg New York, 1990.

[WK97] O. Wendt and W. König. Cooperative simulated annealing: How much cooperation is enough. Working Paper 97-19, Institute of Information Systems, GoetheUniversity Frankfurt, 1997.

[YA94] Y. Yoshida and N. Adachi. A diploid genetic algorithm for preserving population diversity - pseudo-meiosis GA. *Lecture Notes in Computer Science*, 866:36–45, 1994.