Journal of Heuristics, 10: 243–267, 2004

SASEGASA: A New Generic Parallel Evolutionary Algorithm for Achieving Highest Quality Results

MICHAEL AFFENZELLER*
STEFAN WAGNER
Institute of Systems Science, Systems Theory and Information Technology, Johannes Kepler University, Altenbergerstrasse 69, A-4040 Linz, Austria
email: ma@cast.uni-linz.ac.at
email: sw@cast.uni-linz.ac.at

Submitted in September 2003 and accepted by Enrique Alba in March 2004 after 1 revision

Abstract

This paper presents a new generic Evolutionary Algorithm (EA) for retarding the unwanted effects of premature convergence. This is accomplished by a combination of interacting generic methods. These generalizations of a Genetic Algorithm (GA) are inspired by population genetics and take advantage of the interactions between genetic drift and migration. In this regard a new selection scheme is introduced, which is designed to directedly control genetic drift within the population by advantageous self-adaptive selection pressure steering. Additionally this new selection model enables a quite intuitive heuristics to detect premature convergence. Based upon this newly postulated basic principle the new selection mechanism is combined with the already proposed Segregative Genetic Algorithm (SEGA), an advanced Genetic Algorithm (GA) that introduces parallelism mainly to improve global solution quality. As a whole, a new generic evolutionary algorithm (SASEGASA) is introduced. The performance of the algorithm is evaluated on a set of characteristic benchmark problems. Computational results show that the new method is capable of producing highest quality solutions without any problem-specific additions.

Key Words: Evolutionary Algorithm (EA), Genetic Algorithm (GA), selection, selection pressure, premature convergence, genetic drift, migration

1. Introduction

Evolutionary Algorithms (EAs) may be described as a class of bionic techniques that imitate the evolution of a species. In figure 1 we locate EAs in relation to other search techniques with special attention to Genetic Algorithms (GAs) that represent the basis for our new Evolutionary Algorithm.

The fundamental principles of GAs were first presented by Holland (1975). Since that time GAs have been successfully applied to a wide range of problems including multimodal function optimization, machine learning, and the evolution of complex structures such as neural networks. An overview of GAs and their implementation in various fields is given by Goldberg (1989) or Michalewicz (1996).

*Author to whom all correspondence should be addressed.

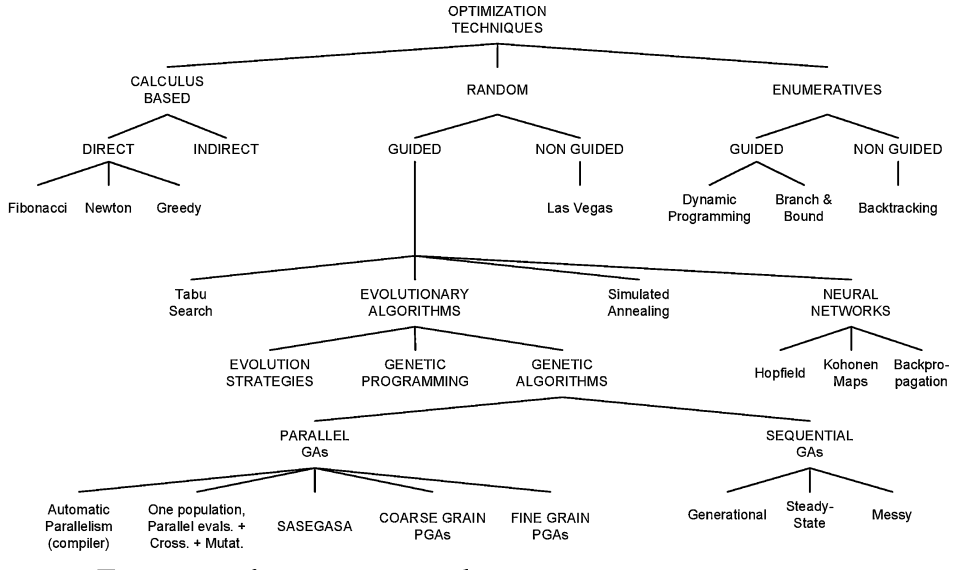

Figure 1. Taxonomy of optimization techniques.

Evolution Strategies, the second major representative of Evolutionary Algorithms, were introduced by Rechenberg (1973) and Schwefel (1994). Evolution Strategies tend to find local optima quite efficiently. Though, in the case of multimodal solution spaces, Evolution Strategies tend to detect a global optimum hardly, if none of the starting values is located in the absorbing region of such a global optimum.

Provided that the used problem representation and the operators are adequate, the advantage of applying Genetic Algorithms to hard problems of combinatorial optimization lies in their ability to search the solution space in a broader way than heuristic methods based upon neighborhood search. Nevertheless, also Genetic Algorithms are 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 Genetic Algorithms, occurs when the population of a GA reaches such a suboptimal state that the genetic operators can no longer produce offspring that outperform their parents (e.g. Fogel, 1994). There is a close relationship between premature convergence and genetic drift which will be further considered within the scope of the present work.

During the last decades plenty of work has been investigated to introduce new coding standards and operators in order to overcome this essential handicap of Genetic Algorithms. As these coding standards and the belonging operators often are quite problem specific, we try to take a different approach. By looking upon the concepts of Genetic Algorithms as an artficial self organizing process in a biologically and sociologically inspired generic way, we aim to improve the global convergence behaviour independently of the actually employed implementation.

A very essential question about the general performance of a GA is, whether or not good parents are able to produce children of comparable or even better fitness (the building block

hypothesis implicitly relies on this). In natural evolution, this is almost always true. For Genetic Algorithms this property is not so easy to guarantee. The disillusioning fact is that the user has to take care of an appropriate coding in order to make this fundamental property hold. In order to overcome this strong requirement we have developed an advanced selection mechanism (Affenzeller and Wagner, 2003b), which is based on the idea to consider not only the fitness of the parents, in order to produce a child for the ongoing evolutionary process. Additionally, the fitness value of the evenly produced offspring is compared with the fitness values of its own parents. The offspring is accepted as a candidate for the further evolutionary process if and only if the reproduction operator was able to produce an offspring that could outperform the fitness of its own parents. This strategy guarantees that evolution is presumed mainly with crossover results that were able to mix the properties of their parents in an advantageous way.

As in the case of conventional GAs, offspring are generated by parent selection, crossover, and mutation. In a second (offspring) selection step, the number of offspring having to be generated depends on a predefined ratio-parameter giving the quotient of next generation members that have to outperform their own(!!) parents. As long as this ratio is not fulfilled further children are created and only the successful offspring will definitely become members of the next generation. When the postulated ratio is reached, the rest of the next generation members are randomly chosen from the children that did not reach the success criterion. Within our new selection model, selection pressure is defined as the ratio of generated candidates to the population size. An upper limit for selection pressure gives a quite intuitive termination heuristics: If it is no more possible to find a sufficient number of offspring that outperform their parents, new genetic information has to be brought in.

Based upon this enhanced EA-model two further generic extensions are being discussed:

(1) By interpreting 'building blocks' as the essential genetic information, which is responsible for locating a global optimal solution, in a more general sense, the concept of segregation and reunification of subpopulations aims to assure an independent development of 'building blocks' in very different regions of the search space in order to improve global convergence. The algorithm divides the population into a certain number of subpopulations. These evolve independently, until local premature convergence is detected which is indicated by a selection pressure value that has reached a given upper limit for a certain subpopulation. In this case the calculations for this subpopulation are stopped until the next reunification phase is reached. Such a reunification phase is initiated, if all subpopulations have converged prematurely. By this approach of width-search, building blocks which would disappear early in the case of standard GAs are evolved in different regions of the search space at the beginning and during the evolutionary process. In contrast to other parallel and distributed GA approaches like island models (Cantu-Paz, 2001), in our algorithm the single subpopulations grow together in case of local stagnation in order to bring in new genetic information into the further evolving and growing subpopulations. This strategy should lead to one final population that has collected all essential building blocks of the subpopulations.
(2) The second newly introduced concept suggests the usage of multiple crossover operators in parallel to somehow imitate the parallel evolution of a variety of species. This strategy

seems very capable for problems which consider more than one crossover operator—especially if the properties of the available operators may change as evolution proceeds. Furthermore, it is observable that population diversity is supported, if more crossover operators are involved.

As an important property of all the newly introduced hybrids it has to be pointed out that the corresponding GA is unrestrictedly included in the new variants of Evolutionary Algorithms under special parameter settings.

The experimental part analyzes the new algorithms for the Travelling Salesman Problem (TSP) as a very well documented instance of a multimodal combinatorial optimization problem. Additionally, the new concepts are tested for a collection of difficult high dimensional real-valued test functions in order to avoid the impression that the new concepts operate especially well for routing problems like the TSP. In contrast to all other evolutionary heuristics known to the authors that do not use any additional problem specific information, we obtain the best known solution for all considered benchmarks.

The rest of the paper is organized as follows: In Section 2, we introduce the self-adaptive selection mechanism for automated selection pressure steering more formally. Based upon this we reformulate the already proposed SEGA algorithm in Section 3 ending up with the new SASEGASA. Section 4 discusses the qualitative performance of SASEGASA under different aspects. Finally, Section 5 summarizes the main results of this contribution.

2. Offspring selection: A new model for self-adaptive selection pressure steering

"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]

The handling of selection pressure in the context of GAs mainly depends on the choice of the selection operator and the replacement scheme (Michalewicz, 1996). 'Generational replacement', for example, replaces the entire population by the next one, whereas 'elitism replacement' keeps the best individuals of the last generation and only replaces the rest and therefore usually performs faster. On the other hand elitism is likely to cause too homogeneous populations, i.e. low population diversity, and therefore might cause unwanted premature convergence. Furthermore, there exists no manageable model for controllable selection pressure handling within the theory of Genetic Algorithms (Schoenburg, Heinzmann, and Feddersen, 1994).

Therefore, we have introduced some kind of intermediate step (a 'virtual population') into the selection procedure which provides a handling of selection pressure very similar

to that of Evolution Strategies and also very close to the population genetics' viewpoint of selection (Affenzeller, 2001b, 2002). As pointed out in the above mentioned publications, the most common replacement mechanisms can easily be implemented in this intermediate selection step. Furthermore, this Evolution Strategy like variable selection pressure is well-suited to steer the degree of population diversity. However, within this model it is necessary to adjust a parameter for the actual selection pressure and in order to steer the search process advantageously a lot of parameter tuning is essential.

Motivated by these considerations, we have set up an advanced selection model which, in principle, acts in the following way:

The First selection step chooses the parents for crossover either randomly or in any well-known way of Genetic Algorithms like roulettewheel, linear-rank, or some kind of tournament selection strategy. After having performed crossover and mutation with the selected parents (sexual selection), we introduce a further selection mechanism that considers the success of the apparently applied reproduction. In order to assure that the proceeding of genetic search occurs mainly with successful offspring, this is done in a way so that the used crossover and mutation operators are able to create a sufficient number of children that surpass their parents' fitness. Therefore, a new parameter, called success ratio (*SuccRatio* $\in [0, 1]$), is introduced. The success ratio gives the quotient of the next population members that have to be generated by successful mating in relation to the total population size. Our offspring selection rule says that a child is successful if its fitness is better than the fitness of its parents, whereby the meaning of 'better' has to be explained in more detail: is a child better than its parents, if it surpasses the fitness of the weaker, the better, or is it in fact some kind of mean value of both?

For this problem we claim that an offspring only has to surpass the fitness value of the worse parent in order to be considered as 'successful' at the beginning, while as evolution proceeds the child has to be better than a fitness value continuously increasing between the fitness of the weaker and the better parent. As in the case of Simulated Annealing, this strategy gives a broader search at the beginning, whereas at the end of the search process this operator acts in a more and more directed way. Having filled up the claimed ratio (*SuccRatio*) of the next generation with successful individuals according to the above meaning, the rest of the next generation $((1 - SuccRatio) \cdot |POP|)$ is simply filled up with individuals randomly chosen from the pool of individuals that were also created by crossover and mutation—but did not reach the success criterion. The actual selection pressure *ActSelPress* at the end of a single generation is defined by the quotient of individuals that had to be considered until the success ratio was reached, and the number of individuals in the population in the following way:

$$ActSelPress = \frac{|virtualPOP| + SuccRatio \cdot |POP|}{POP}$$

Figure 2 shows the operating sequence of the above described concepts.

An upper limit of selection pressure (*MaxSelPress*) defines the maximum number of offspring considered for the next generation (as a multiple of the actual population size) that may be produced in order to fulfill the success ratio. With a sufficiently high setting of (*MaxSelPress*), this new model also functions as a obvious heuristics for the detection of premature convergence:

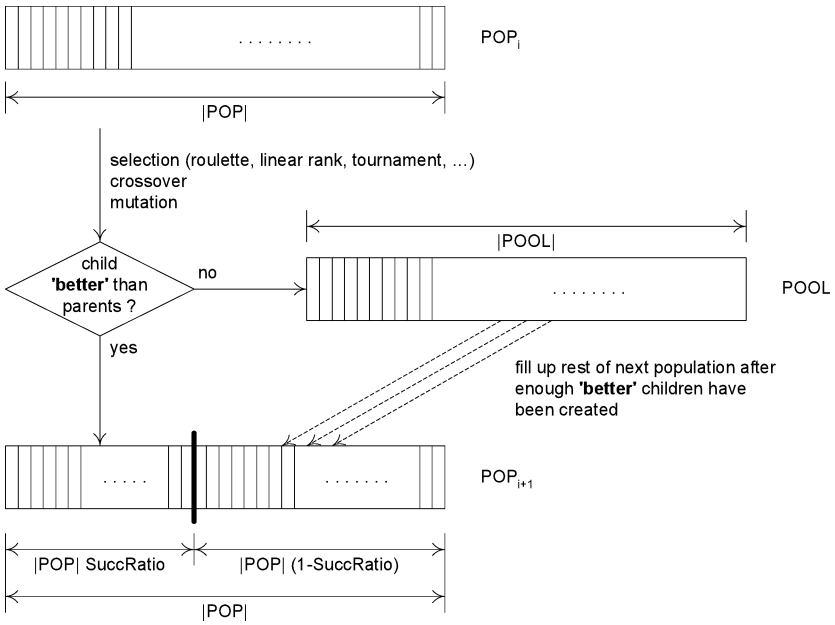

Figure 2. Flowchart for embedding the new offspring selection principle into a Genetic Algorithm.

If it is no longer possible to find a sufficient number ($SuccRatio \cdot |POP|$) of offspring outperforming their own parents even if ($MaxSelPress \cdot |POP|$) candidates have been generated, premature convergence has occurred.

Especially in the distributed version where a certain number of subpopulations evolve independently, local premature convergence can be detected efficiently in a certain subpopulation and the calculations for this subpopulation can be stopped. Then, if local premature convergence has occurred in all subpopulations, a new reunification phase has to be performed as described in detail in Section 3.1.

As a basic principle of this selection model, a higher success ratio causes higher selection pressure. Nevertheless, higher settings of success ratio, and therefore of selection pressure, do not necessarily cause premature convergence. The reason for this is mainly, that the new selection step (after crossover) per definition does not accept clones that emanate from two identical parents. In conventional GAs such clones represent a major reason for premature convergence of the whole population around a suboptimal value, whereas the new selection step specifically counteracts against this phenomenon.

A further aspect worth mentioning is that our additional offspring selection mechanism balances the disadvantages of certain crossover operators because corrupt descendants do not achieve the success criterion anyway. As long as at least ‘sometimes’ advantageous crossover result are produced, it is observable that provably worse operators (Larranaga et al., 1999) are able to achieve results in the range of the best operators for a certain problem. This is because the disadvantageous crossover results are simply not taken into account (cf. Section 4).

Furthermore, if more than one crossover operator is available for a certain problem representation, it is beneficial and recommendable to apply all contemplable crossover operators in parallel with the same probability. In this way all operators (even the worse) propagate only their advantageous crossover results into the next generation and as different crossover mechanisms tend to generate different offspring, this strategy increases the broadness of genetic search without counteracting the goal directedness, which also helps to retard premature convergence.

Experiments performed on the variable selection pressure model already indicate the supremacy of this approach (Affenzeller and Wagner, 2003b). Also the corresponding canonical Genetic Algorithm is fully included in our new superstructure if the success ratio is simply set to 0. Furthermore, we are going to discuss new aspects and models built up upon the described self adaptive selection model in the following section.

3. Parallel GA concepts in combination with self-adaptive selection pressure

When applying Genetic Algorithms to complex problems, one of the most frequent difficulties is the stagnation of solution quality due to premature convergence. Concisely speaking, premature convergence occurs when the population of a Genetic Algorithm reaches such a suboptimal state that the genetic operators can no longer produce offspring that outperform their parents (e.g. Fogel, 1994). The most frequent reason for premature convergence is the fixing of alleles to a characteristic that does not belong to an optimal solution (due to genetic drift).

Several methods have been proposed to combat premature convergence in Genetic Algorithms (e.g. Cobb and Grefenstette, 1993; De Jong, 1975; Goldberg, 1989). These include, for example, the restriction of the selection procedure, the operators and the according probabilities as well as the modification of the fitness assignment. However, all these methods are heuristic per definition. Their effects vary with different problems and their implementation strategies need ad hoc modifications with respect to different situations. Hence a critical problem in studying premature convergence is the identification of its occurrence and the characterization of its extent. Srinivas and Patnaik (1994), for example, use the difference between the average and maximum fitness as a standard to measure premature convergence, and adaptively vary crossover and mutation probabilities according to this measurement. On the other hand, the term 'population diversity' has been used in many papers to study premature convergence (e.g. Smith, Forrest, and Perelson, 1993; Yoshida and Adachi, 1994) where the decrease of population diversity (i.e. a homogeneous population) is considered as the primary reason for premature convergence. This situation shows some kind of analogy to the situation when a neighborhood-based search technique like Simulated Annealing is unable to escape from a local minimum.

In this paper we try to get to the bottom of premature convergence a little bit more by explaining and retarding premature convergence with the help of (well directed) genetic drift. Using the terminology of population genetics for the theoretical description of GA concepts, genetic drift is advantageous, if the allele that is fixed by genetic drift represents a part of a global optimal solution. However, when applying GAs to higher dimensional problems of combinatorial optimization, it unfortunately happens that genetic drift also causes a fixing

of alleles to non optimal properties, which causes a loss of optimal properties in the entire population. This effect is especially observable, if attributes of a global optimal solution with minor influence on the fitness function, are 'hidden' in individuals with a bad total fitness. In that case sexual selection additionally promotes the drop out of that attributes.

This is exactly the point where considerations about multiple subpopulations that systematically exchange information come into play:

Splitting the entire population into a certain number of subpopulations (demes) causes the separately evolving subpopulations to explore different genetic information in the certain demes due to the stochastic nature of genetic drift. Especially in the case of multimodal problems the different subpopulations tend to prematurely converge to different suboptimal solutions. The idea is that the building blocks of a global optimal solution are scattered in the single subpopulations, and the aim is to develop concepts to systematically bring together these essential building blocks in one subpopulation in order to make it possible to find a global optimal solution by crossover. In contrast to the various coarse- and fine-grained parallel GAs that have been discussed in the literature (a good review is given in Alba and Troya (1999) for instance), we decided to take a different approach by letting the demes grow together step by step in case of local premature convergence. Even if this property does not support parallelization as much as established parallel GAs, we decided to introduce this concept of migration as it proved to support the localization of global optimal solutions to a greater extent. Of course the concept of self adaptive selection pressure steering is essential, especially in the migration phases, because these are exactly the phases where the different building blocks of different subpopulations have to be unified. In classical parallel GAs it has been tried to achieve this behavior just by migration which is basically a good, but not very efficient idea, if no deeper thoughts about selection pressure are spent at the same time.

As first experiments have already shown, there also should be a great potential in equipping established parallel GAs with our newly developed self adaptive selection pressure steering mechanism (more stability in terms of operators and migration rates, automated detection of migration date, etc.). Anyway, the combination of self-adaptive offspring selection (in the sense of Section 2) with established parallel GA concepts deserves closer attention and is planned to be undertaken within a concrete research project.

In the following we describe the generic extensions, that are built upon the self-adaptive offspring selection model, which we have introduced and generically brought together in the so called SASEGASA algorithm.

3.1. SASEGASA: The core algorithm

In principle, the new SASEGASA (Self Adaptive SEgregative Genetic Algorithm with Simulated Annealing aspects) introduces two enhancements to the basic concept of Genetic Algorithms. Firstly, we bring in a variable selection pressure, as described in Section 2, in order to self-adaptively control the goal-orientedness of genetic search. The second concept introduces a separation of the population to increase the broadness of the search process, and joins the subpopulation after their evolution (after local premature convergence in the subpopulations) in order to end up with a population including all genetic information sufficient for locating a global optimum.

The basic properties (in terms of solution quality) of this segregation and reunification approach, have already proven their potential in overcoming premature convergence in the so-called SEGA algorithm (SEgregative GA) (Affenzeller, 2001a, 2001c).

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 for a specified number of iterations that is considered to be sufficient for the subpopulations to develop. Then a reunification from n to $(n-1)$ subpopulations is performed.

Roughly speaking this means, that there are a certain number of villages at the beginning of the evolutionary process that are slowly growing together to bigger towns, ending up with one big city containing the whole population at the end. By using this approach of width-search, building blocks in different regions of the search space are evolved at the beginning and during the evolutionary process. In the case of standard Genetic Algorithms these would disappear early and their genetic information could not be provided at a later date of evolution when the search for global optima is of paramount importance.

However, within the SEGA algorithm there is no criterion to detect premature convergence, and there is also no self-adaptive selection pressure steering mechanism. Even if the results of SEGA are quite good with regard to global convergence, it requires an experienced user to adjust the selection pressure steering parameters, and as there is no criterion to detect premature convergence the dates of reunification have to be implemented statically.

Equipped with our new self adaptive selection technique we have both: A self-adaptive selection pressure, as well as an automated detection heuristics for local premature convergence. Therefore, a date of reunification has to be set, if local premature convergence has been detected within all subpopulations, in order to increase genetic diversity again. Figure 3 shows a schematic diagram of the migration policy in the case of a reunification phase of the SASEGASA algorithm. The grey shaded subpopulations stand for already prematurely converged subpopulations. If all subpopulations are prematurely converged (grey shaded) a new reunification phase is initiated.

Figures 4 and 5 show the typical shape of the fitness curve, and the selection pressure curve representative for a SASEGASA test run. Due to better presentability the number of subpopulations is restricted to 10. The vertical lines indicate a date of reunification. In the quality diagram (figure 4) the lines give the fitness value of the best member of each deme as the relative distance to the best known solution, which is represented by the horizontal line. In the selection pressure diagram (figure 5) the lines stand for the actual selection pressure in the certain demes, as the actual quotient of evaluated solution candidates per round (in a deme) to the subpopulation size. The bottom of the diagram represents a selection pressure of 1 and the upper horizontal line represents the maximum selection pressure. If the actual selection pressure of a certain deme exceeds the maximum selection pressure, local premature convergence is detected in this subpopulation and evolution is stopped in this deme (which can be seen in the constant value of the corresponding fitness curve) until the next reunification phase is started (if all demes are prematurely converged).

With all the above described strategies, the complete SASEGASA algorithm can be stated as described in figure 6.

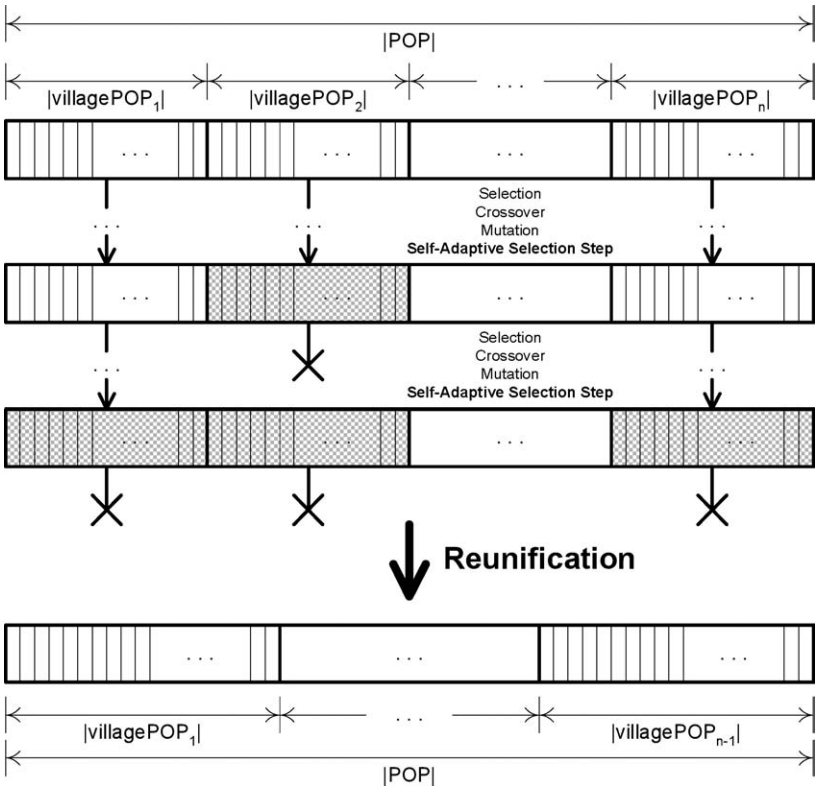

Figure 3. Flowchart for the reunification of subpopulations.

Again, like in the context of the new selection model which is included in SASEGASA as well, it should be pointed out that a corresponding Genetic Algorithm is unrestrictedly included in SASEGASA, when the number of subpopulations (villages) is set to 1 and the success ratio is set to 0 at the beginning of the evolutionary process. Moreover, the introduced techniques also do not use any problem specific information.

4. Experimental results

Empirical studies with different problem classes and instances are the most effective way to analyze the potential of heuristic optimization searches like Evolutionary Algorithms.

In our experiments, all computations are performed on a Pentium 4 PC with 1 GB RAM under Windows XP. The programs are written in the C# programming language using the Microsoft .NET framework 1.1. All values presented in the following tables are the best resp. average values of five independent test runs executed for each test case. The average number of evaluated solutions gives a quite objective measure of the computational effort.

We have selected the Travelling Salesman Problem (TSP) as a test instances because of several reasons:

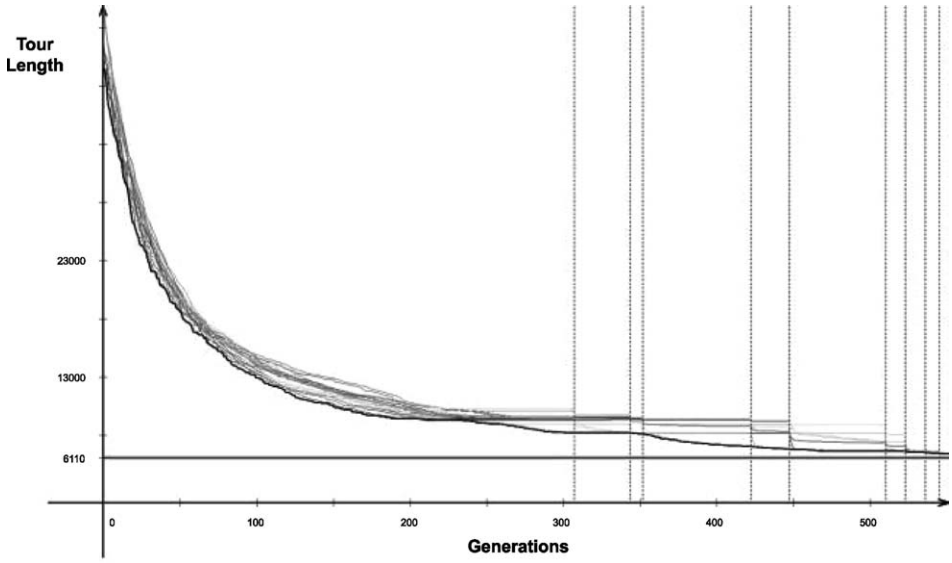

Figure 4. Fitness curves for a typical run of the SASEGASA algorithm.

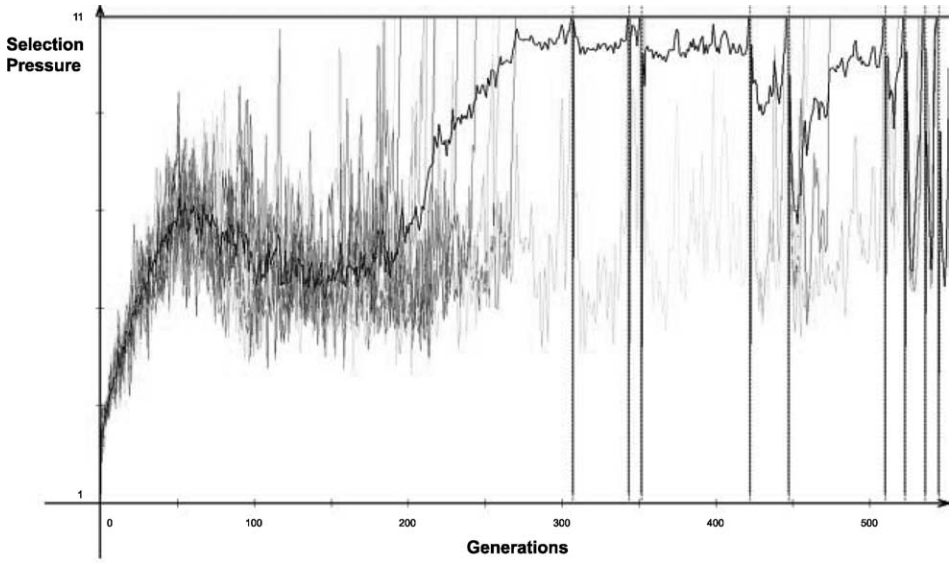

Figure 5. Selection pressure curves for a typical run of the SASEGASA algorithm.

Firstly, there are a plenty of standardized benchmark test cases (of various degree of difficulty) available for this, possibly most established, instance of an NP-complete combinatorial optimization problem which allows to draw fair comparisons between different optimization techniques.

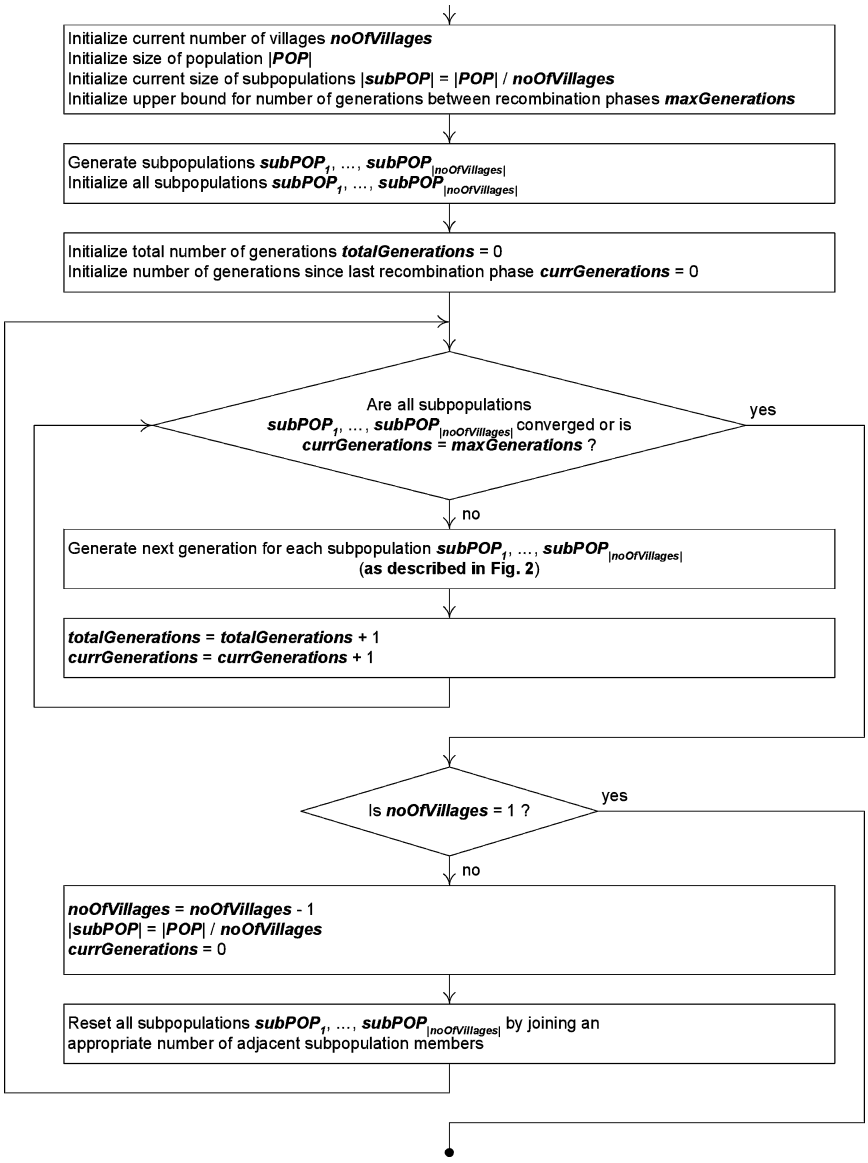

Figure 6. Flowchart of the core algorithm (SASEGASA).

Secondly, the TSP is discussed in great detail for GA applications. There are a lot of GA representations and GA operators available for this problem instance which is of great interest especially for our considerations and analyses.

However, it is to be pointed out, that the newly introduced methods are by no means restricted or somehow optimized to routing problems like the TSP. Further experimental tests

on other combinatorial optimization problems like the JSSP (Job Shop Scheduling Problem), the MPSP (Multi Processor Scheduling Problem), or the TTP (Timetabling Problem) have already been performed and show the same effects and improvements as are being described here for the TSP.

Furthermore, the newly developed theories are already applied in practical projects like the structure identification of formulae based on time series data by using Genetic Programming (GP) (Winkler, Affenzeller, and Wagner, 2004) and can also come up to one's expectations for these application areas because of the generic design.

In order to circumstantiate the generality of our methodologies, the new concepts are also tested for a collection of difficult high dimensional real-valued test functions. Also for the test functions the SASEGASA algorithm is able to find the global optimum even in higher dimensions.

4.1. Experimental results for the TSP

The TSP benchmark problems are taken from the TSPLIB (Reinelt, 1991) using updated results[2] for the best or at least the best known solutions. The results for the TSP are represented as the relative difference to the best known solution defined as $relativeDifference = (\frac{Result}{Optimal} - 1) \cdot 100\%$.

To visualize the positive effects of the new methods in a more obvious way we also present results that were generated by a classical GA with generational replacement and 1-elitism. Algorithm 1 shows the used algorithm in detail which is one of the very commonly used GA variants.

Algorithm 1 Classical Genetic Algorithm

Initialize total number of generations $noOfGenerations \in \mathbb{N}$
Initialize actual number of generations $i = 0$
Initialize size of population $|POP| \in \mathbb{N}$
Initialize mutation rate $mutRate \in [0, 1]$
Initialize elitism rate $elitism \in [0, |POP|]$
Produce an initial population $|POP_0|$ with size $|POP|$ randomly
while $i \leq noOfGenerations$ **do**
  Initialize next population POP_{i+1}
  while $|POP_{i+1}| < |POP| - 1$ **do**
    Select two individuals from the members of POP_i
    Generate new child by crossing the two selected parents
    Mutate child with probability $mutRate$
    Insert child into POP_{i+1}
  end while
  Insert $elitism$ best individuals from POP_i into POP_{i+1}
  $i = i + 1$
end while

Table 1. Parameter values used in the test runs of the several algorithms.

Parameters for the classical GA (Results presented in Table 2)	
Generations	100'000
Population size	120
Elitism rate	1
Mutation rate	0.05
Selection operator	Roulette
Mutation operator	Inversion & translocation
Parameters for SASEGASA with 1 subpopulation (Results presented in Table 3)	
Population size	500
Elitism rate	1
Mutation rate	0.05
Selection operator	Roulette
Mutation operator	Inversion & translocation
Success ratio	0.7
Maximum selection pressure	250
Parameters for SASEGASA (Results presented in Table 4)	
Initial subpopulation size	100
Elitism rate	1
Mutation rate	0.05
Selection operator	Roulette
Crossover operators	OX, ERX, MPX
Mutation operator	Inversion & translocation
Success ratio	0.8
Maximum selection pressure	15

In Table 2 the results achieved with the classical GA are listed. The fixed parameter values for all algorithms that were used in the different test runs are given in Table 1.

Especially we aim to point out two major qualities of the new algorithm: First, we illustrate the effects of the new selection on the basis of a number of experiments which were performed on a single population in order not to dilute the effects of the new selection principles with the effects of the segregation and reunification strategy. Table 3 recapitulates the results for a variety of crossover operators suggested for the TSP (Michalewicz, 1996; Larranaga et al., 1999) each on its own as well as one combination of more effective crossover operators.

Remarkable in this context is the effect that also crossover operators that are considered as rather unsuitable for the TSP (Larranaga et al., 1999) achieve quite good results in

Table 2. Experimental results achieved with the classical GA.

Problem	Crossover	Best	Average	Evaluated solutions
berlin52	OX	0.00	3.76	12'000'000
berlin52	ERX	5.32	7.73	12'000'000
berlin52	MPX	21.74	26.52	12'000'000
ch130	OX	3.90	5.41	12'000'000
ch130	ERX	142.57	142.62	12'000'000
ch130	MPX	83.57	85.07	12'000'000
kroa200	OX	3.14	4.69	12'000'000
kroa200	ERX	325.92	336.19	12'000'000
kroa200	MPX	146.94	148.08	12'000'000

Table 3. Experimental results achieved with the new selection principle.

Problem	Crossover	Best	Average	Evaluated solutions	Change to classical GA
berlin52	OX	0.00	1.90	14'250'516	−1.86
berlin52	ERX	0.00	1.97	6'784'626	−5.76
berlin52	MPX	0.00	0.76	6'825'199	−25.76
berlin52	OX, ERX, MPX	0.00	0.90	7'457'451	–
ch130	OX	1.54	2.26	13'022'207	−3.15
ch130	ERX	0.57	2.11	4'674'485	−140.51
ch130	MPX	1.11	3.18	9'282'291	−81.89
ch130	OX, ERX, MPX	0.68	1.18	5'758'022	–
kroa200	OX	2.73	3.51	15'653'950	−1.18
kroa200	ERX	3.21	5.40	19'410'458	−330.79
kroa200	MPX	3.28	4.65	13'626'348	−143.43
kroa200	OX, ERX, MPX	2.34	3.04	9'404'241	–

combination with the new selection model. The reason for this behavior is given by the fact that in our selection principle only children that have emerged as a good combination of their parents' attributes are considered for the further evolutionary process. In combination with a higher upper value for the maximum selection pressure genetic search can therefore be guided advantageously also for poor crossover operators as the larger amount of handicapped offspring is simply not considered for the further evolutionary process. Figure 7 shows this effect in detail for the berlin52 TSP. Quite good results in terms of global convergence could also be achieved with a combination of different randomly selected crossover operators as additional genetic diversity is brought to the population by this action.

Secondly, we point out the main effects of the present contribution—namely that an increasing number of subpopulations at the beginning of the evolutionary process allows to

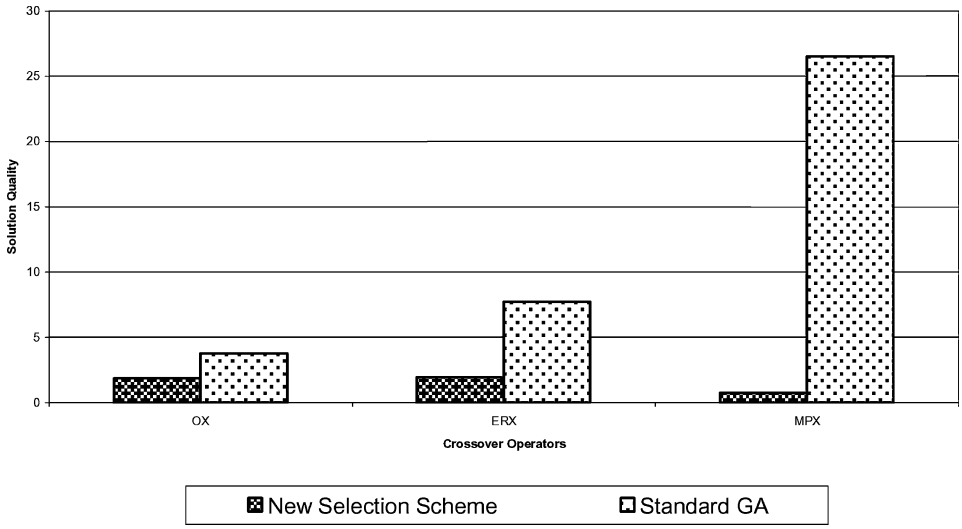

Figure 7. Improvement of various crossover operators due to the new selection principle.

achieve scalable improvements in terms of global convergence. As many different crossover operators are discussed for the path representation for the TSP and the selfadaptive selection of SASEGASA offers the opportunity, the tests for SASEGASA have been performed with a combination of crossover operators, whereby the crossover operator was selected randomly over the three choices. As Table 4 shows, the global solution can be scaled up to highest quality regions by just increasing the number of evolving subpopulations. This definitely represents a new achievement in the area of Evolutionary Algorithms with distributed subpopulations.

Indeed, as Table 4 shows, the optimal solution could be found for all benchmark test cases if the initial number of subpopulations is set high enough. Of course, a greater number of subpopulations also implies a greater number of solutions, that have to be evaluated. Nevertheless, the scalability of global solution quality with a greater number of generated solutions is a real advantage to a classical GA where a greater number of evaluated solutions can not improve global solution quality anymore after the GA has prematurely converged.

Even if the achieved results are clearly superior to most of the results reported for applications of Evolutionary Algorithms to the TSP (Larranaga et al., 1999), it has to be pointed out again that all introduced and applied additions to a standard evolutionary algorithm are generic and absolutely no problem specific local pre- or post-optimization techniques have been applied in our experiments.

4.2. Empirical analysis of standardized test functions

In addition to the empirical analysis of the TSP, this section discusses the performance of SASEGASA for some of the more popular real valued test functions in different dimensions.

Table 4. Results showing the scaling properties of SASEGASA.

Problem	Sub-populations	Best	Average	Evaluated solutions
berlin52	1	4.39	7.51	74’203
berlin52	5	0.00	0.00	906’555
berlin52	10	0.00	0.00	1’742’478
berlin52	20	0.00	0.00	4’719’682
berlin52	40	0.00	0.00	38’669’859
berlin52	80	0.00	0.00	96’222’047
berlin52	160	0.00	0.00	309’898’466
ch130	1	26.28	31.13	140’732
ch130	5	4.96	5.35	1’273’962
ch130	10	2.16	2.78	3’669’370
ch130	20	0.54	1.37	38’533’693
ch130	40	0.74	1.23	79’717’698
ch130	80	0.00	0.19	223’721’674
ch130	160	0.00	0.08	298’820’514
kroa200	1	43.86	45.32	285’330
kroa200	5	10.58	10.73	2’049’418
kroa200	10	4.06	7.37	5’382’084
kroa200	20	2.18	2.37	61’362’510
kroa200	40	0.9	2.42	78’447’908
kroa200	80	0.6	1.14	171’520’767
kroa200	160	0.00	0.23	320’857’902
lin318	1	64.78	77.56	403’431
lin318	5	17.60	20.53	3’292’861
lin318	10	7.95	10.93	8’093’264
lin318	20	2.38	4.35	26’534’811
lin318	40	1.97	3.19	200’885’952
lin318	80	1.54	2.34	624’986’088
lin318	160	0.80	1.12	959’258’717
lin318	320	0.00	0.52	2’116’724’528
fl417	1	60.31	89.11	585’102
fl417	5	9.86	13.78	5’104’971
fl417	10	2.70	5.14	26’586’653
fl417	20	1.50	2.35	106’892’925
fl417	40	0.29	0.91	664’674’431
fl417	80	0.21	0.27	1’310’193’035
fl417	160	0.00	0.11	2’122’856’932
fl417	320	0.00	0.00	4’367’217’575

The main purpose of this additional analysis is to underpin the generality of the newly introduced concepts and methods. As no problem specific knowledge or local search is involved, the identical SASEGASA is applied as in the tests for the TSP—just the problem representation and the crossover and mutation operators have been exchanged with standard operators known from GA literature for real valued encoding (e.g. Dumitrescu et al., 2000). This analysis for the standard test functions should demonstrate the generic capability of the newly introduced concepts to achieve highest quality solutions for various very difficult and tricky problems (especially in the higher dimensional cases), in order to avoid the impression that the new concepts operate especially well for routing problems like the TSP.

The test functions that are used in these sections have been designed by several authors for analyzing and comparing different optimization algorithms. Most of the test functions have especially been designed to detect weak points of the different methods. Test functions have for instance been published by De Jong (1975), Goldberg (1989), Schwefel (1994), or Mühlenbein et al. (1991). We have selected the following test functions that are considered to be more difficult and whose degree of difficulty can by scaled up by increasing the dimension of the search space due to the with the problem dimension exponentially increasing number of local minima (except the unimodal Rosenbrock function).

– *The n-dimensional Rosenbrock function*:

$$f(\vec{x}) = \sum_{i=1}^{n-1} 100 \cdot \left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2$$

for $-2.048 \leq x(i) \leq 2.048$ with a global minimum of $f(\vec{x}) = 0$ at $\vec{x} = (1, 1, 1, \ldots\ldots 1)$.

– *The n-dimensional Rastrigin function*:

$$f(\vec{x}) = 10 \cdot n + \sum_{i=1}^{n} x_i^2 - 10\cos(2 \cdot \pi \cdot x_i)$$

for $-5.12 \leq x(i) \leq 5.12$ with a global minimum of $f(\vec{x}) = 0$ at $\vec{x} = (0, 0, 0, \ldots\ldots 0)$.

– *The n-dimensional Schwefel function (Sine Root)*:

$$f(\vec{x}) = 418.982887272433 \cdot n + \sum_{i=1}^{n} -x_i \sin\left(\sqrt{|x_i|}\right)$$

for $-500 \leq x(i) \leq 500$ with a global minimum of $f(\vec{x}) = 0$ at $\vec{x} = (420.968746453712, 420.968746453712, \ldots\ldots 420.968746453712)$.

– *The n-dimensional Griewangk function*:

$$f(\vec{x}) = \frac{\cdot 1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

for $-600 \leq x(i) \leq 600$ with a global minimum of $f(\vec{x}) = 0$ at $\vec{x} = (0, 0, 0, \ldots\ldots 0)$.

– *The n-dimensional Ackley function*:

$$f(\vec{x}) = 20 + e - 20 \cdot e^{-0.2} \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} - e^{\frac{1}{n}} \sum_{i=1}^{n} \cos(2 \cdot \pi \cdot x_i)$$

for $-32.768 \leq x(i) \leq 32.768$ with a global minimum of $f(\vec{x}) = 0$ at $\vec{x} = (0, 0, 0, \ldots\ldots 0)$.

As all considered test functions have a unique global minimum of 0 we state the results of the following tables as the absolute difference between the achieved result and 0. Many papers interpret a solution to be globally optimal if it is in a certain range (e.g. 10^{-5}) around the global optimum. For reasons of objectiveness we state the exact values of the achieved solutions as it frequently happens that it still takes plenty of iterations to improve the solution quality from a value of 10^{-5} to 10^{-20} for example. Consequently, a displayed result of 0.000000000 has to be interpreted to be smaller than the lower limit of the C# double data type.

Table 5 shows the SASEGASA parameter settings for the different benchmark test functions.

Even if the Rosenbrock function is a unimodal function it is considered to be rather hard to solve with GA concepts. The Rosenbrock function is characterized by an extremely deep valley along a parabolic curve. Optimization literature often discusses the 2-dimensional Rosenbrock function (e.g. Potter et al., 1994). More directed empirical analysis, applying a special distributed GA variant to the 5-dimensional Rosenbrock function, are performed in Hiroyasu et al. (2000). Takahashi et al. analyze the performance of a new hybrid GA variant on the example of the 20-dimensional Rosenbrock function. The SASEGASA algorithm is able to find the global optimum of the Rosenbrock function up to a dimension of 1000 without any problem specific information of local search, i.e. with the same algorithmic concepts as used before for the TSP.

The primary characteristic of the highly multimodal Rastrigin function is the existence of many suboptimal peaks whose values increase as the distance from the global optimum increases. The number of minima in the typically used interval of $[-5.12, 5.12]^n$ is given by 11^n (Liang, Yao, and Newton, 2000).

The interesting characteristic of the Schwefel sine root function is the presence of a second-best minimum far away from the global minimum—intended to trap optimization algorithms on a suboptimal peak. The number of minima in the typically used interval of $[-500, 500]^n$ is given by 8^n (Liang, Yao, and Newton, 2000) for the Schwefel sine root function.

The Griewangk function has a product term, introducing an interdependency between the variables. This is intended to disrupt optimization techniques that work on one function variable at a time.

The landscape of the Ackley function seems unimodal at a low resolution. However, the second exponential term covers the landscape with many small peaks and valleys. In the general (n-dimensional) case the number of minima is given by 65^n (Liang, Yao, and Newton, 2000) for the Ackley function in the typically considered interval of $[-32, 32]^n$.

Table 5. Parameter values used in the test runs of SASEGASA for the different test functions.

SASEGASA Parameters for the n-dimensional Rosenbrock function (Results presented in Table 6)	
Subpopulation size	100
Number of subpopulations	1
Elitism rate	1
Mutation rate	0.05
Selection operator	Roulette
Crossover operators	Discrete, Average, Heuristic
Mutation operator	One-position Uniform Mutation
Success ratio	0.6
Comparison factor	1.0 (constantly)
Maximum selection pressure	30
SASEGASA Parameters for the n-dimensional Rastrigin, Schwefel (Sine Root), Griewangk, and Ackley functions (Results presented in Table 7–10)	
Initial subpopulation size	100
Number of subpopulations	10
Elitism rate	1
Mutation rate	0.05
Selection operator	Roulette
Crossover operators	Discrete, average, Heuristic
Mutation operator	One-position uniform mutation
Success ratio	0.8
Comparison factor	0.9–1.0
Maximum selection pressure	12

Table 6. SASEGASA results for the Rosenbrock function.

Problem dimension	Best	Average	Evaluated solutions
$n = 20$	$3.11 \cdot 10^{-25}$	$5.71 \cdot 10^{-25}$	2'926'331
$n = 50$	$3.42 \cdot 10^{-14}$	$3.56 \cdot 10^{-12}$	7'383'413
$n = 100$	$2.18 \cdot 10^{-8}$	$2.21 \cdot 10^{-5}$	16'859'987
$n = 200$	$9.84 \cdot 10^{-10}$	$9.52 \cdot 10^{-9}$	55'165'719
$n = 500$	$4.86 \cdot 10^{-9}$	$1.01 \cdot 10^{-8}$	181'099'934
$n = 1000$	$5.12 \cdot 10^{-8}$	$2.62 \cdot 10^{-7}$	493'559'822

Table 7. SASEGASA results for the Rastrigin function.

Problem dimension	Best	Average	Evaluated solutions
$n = 20$	0.000000000	0.000000000	1'390'178
$n = 50$	0.000000000	0.000000000	3'672'293
$n = 100$	0.000000000	0.000000000	8'449'073
$n = 200$	0.000000000	0.000000000	19'629'709
$n = 500$	0.000000000	$3.03 \cdot 10^{-13}$	59'908'514
$n = 1000$	$1.48 \cdot 10^{-11}$	$3.52 \cdot 10^{-11}$	109'216'384
$n = 2000$	$4.26 \cdot 10^{-9}$	$2.61 \cdot 10^{-8}$	169'469'734

Table 8. SASEGASA results for the Schwefel (Sine Root) function.

Problem dimension	Best	Average	Evaluated solutions
$n = 20$	$-2.00 \cdot 10^{-11}$	$-2.00 \cdot 10^{-11}$	1'227'355
$n = 50$	$-5.09 \cdot 10^{-11}$	$-5.09 \cdot 10^{-11}$	3'198'299
$n = 100$	0.000000000	0.000000000	7'323'552
$n = 200$	$1.02 \cdot 10^{-10}$	$1.02 \cdot 10^{-10}$	15'384'177
$n = 500$	$5.24 \cdot 10^{-10}$	$5.24 \cdot 10^{-10}$	50'489'780
$n = 1000$	$4.07 \cdot 10^{-10}$	$4.30 \cdot 10^{-8}$	90'737'007
$n = 2000$	$4.19 \cdot 10^{-8}$	$5.32 \cdot 10^{-7}$	157'636'676

Table 9. SASEGASA results for the Griewangk function.

Problem dimension	Best	Average	Evaluated solutions
$n = 20$	0.000000000	0.000000000	1'712'422
$n = 50$	0.000000000	0.000000000	4'238'489
$n = 100$	0.000000000	0.000000000	10'061'281
$n = 200$	0.000000000	0.000000000	29'016'670
$n = 500$	$2.22 \cdot 10^{-16}$	$4.07 \cdot 10^{-16}$	72'977'814
$n = 1000$	$3.51 \cdot 10^{-16}$	$8.21 \cdot 10^{-16}$	134'165'002
$n = 2000$	$6.42 \cdot 10^{-15}$	$1.14 \cdot 10^{-14}$	228'741'378

Many new theoretical serial and parallel GA concepts are analyzed considering the highly multimodal Rastrigin, Ackley, Griewangk or Schwefel's Sine Root function (e.g. Potter and De Jong, 1994; Hiroyasu et al., 2000, Takahashi, Kita, and Kobayashi, 1999; Liang, Yao, and Newton, 2000; Gordon and Whitley, 1993; Yao, 1997). However, these papers analyze rather low problem dimensions around $n = 10$ for the Rosenbrock function and about

Table 10. SASEGASA results for the Ackley function.

Problem dimension	Best	Average	Evaluated solutions
$n = 20$	$3.11 \cdot 10^{-15}$	$3.11 \cdot 10^{-15}$	2'024'674
$n = 50$	$3.16 \cdot 10^{-15}$	$5.47 \cdot 10^{-15}$	6'296'073
$n = 100$	$6.66 \cdot 10^{-15}$	$9.03 \cdot 10^{-15}$	16'015'281
$n = 200$	$3.86 \cdot 10^{-14}$	$4.34 \cdot 10^{-14}$	41'855'649
$n = 500$	$2.73 \cdot 10^{-13}$	$4.64 \cdot 10^{-13}$	110'402'886
$n = 1000$	$1.12 \cdot 10^{-12}$	$5.97 \cdot 10^{-12}$	194'382'779
$n = 2000$	$9.51 \cdot 10^{-11}$	$4.19 \cdot 10^{-10}$	295'716'608

$n = 20$ or $n = 30$ for the other test functions. The SASEGASA algorithm is able to find the global optimum of all those test functions up to a problem dimension of $n = 1000$ for the Rosenbrock functions, and even up to $n = 2000$ for Rastrigin, Ackley, Griewangk and Schwefel without any additional algorithmic adaptations, i.e. with absolutely the same algorithm as used before for the TSP. Just the operators (crossover and mutation) have been replaced by standard crossover operators for real-valued encoding.

This exemplarily con.rms the general claim for the generality of the newly introduced concepts which should allow to attack and improve plenty of problems considered in GA literature by just inserting the certain problem specific standard operators into the generic SASEGASA algorithm.

5. Conclusion

In this paper an enhanced Genetic Algorithm and two upgrades have been presented and exemplarily tested on some TSP benchmarks as well as on the optimization of some high dimensional real-valued test functions. The proposed EA-based techniques couple aspects from Evolution Strategies (adjustable selection pressure), Simulated Annealing (growing directedness of the search) as well as a special segregation and reunification strategy with crossover and mutation in the general model of a Genetic Algorithm. Therefore, 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 by considering the desired and also the unwanted effects of genetic drift. From a more practical point of view we introduce methods which make the algorithm more open for scalability in terms of solution quality versus computation time.

Concerning the speed of SASEGASA, it has to be pointed out that the superior performance concerning global convergence requires a higher computation time, mainly because of the greater total population size $|POP|$ and because of the increase of generated individuals in our new self adaptive selection mechanism under higher selection pressure. Nevertheless, in contrast to other implementations in the field of evolutionary computation,

it is absolutely remarkable, that it has become possible to almost linearly improve the global solution quality by introducing greater population sizes and an accordingly greater number of subpopulations, whereas the computational costs are 'only' increasing linearly due to the greater number of individuals having to be taken into account. This allows to transfer already developed GA-concepts to increasingly powerful computer systems in order to achieve better results. In case of conventional GAs no more increase in solution quality is possible once the algorithm is prematurely converged independent of the computational power being used. Similar to the motivation of diverse coarse- or fine-grained parallel GAs using parallel computer architectures is very well suited to increase the performance of SASEGASA.

Anyway, with special parameter settings the corresponding Genetic Algorithm is fully included within the introduced concepts achieving a runtime performance only marginally worse than the performance of the equivalent Genetic Algorithm. In other words, the introduced models can be interpreted as a superstructure of the GA model or as a technique downwards compatible to Genetic Algorithms. Therefore, an implementation of the new algorithm(s) for a certain problem is quite easy to do, presumed that the corresponding Genetic Algorithm (coding, operators) is known.

Even if this paper only shows results for the TSP and for real-valued test functions simply because a lot of standardized benchmarks are available for these classical optimization problems, we could already observe the same effects and improvements when applying the new concepts to other problems of combinatorial optimization like the Job Shop Scheduling Problem (JSSP), Multiprocessor Scheduling (MS) or the Timetabling Problem (TTP). Furthermore, the described techniques are on the verge to prove their potential for Genetic Programming applications: In a recently starting industrial project we adopt the described principles for structure identification of mechatronic systems by means of Genetic Programming (GP) (Winkler, Affenzeller, and Wagner, 2004). Especially the success criterion in our self adaptive selection pressure mechanism is expected to balance the unwanted effects of useless crossover results which occur very frequently in GP.

Because of better comparability, no additional problem specific methods like commonly used hill-climbing or certain other pre- or postoptimization techniques have been considered in the examples presented in the experimental results section. However, there exists absolutely no objection against doing so in order to improve the solution quality for a certain problem.

As we have already shown exemplarily a main area of applications for the new Evolutionary Algorithm is given by problems 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 self-adaptive selection pressure, segregation and reunification quite easily.

Notes

1. Taken from the introduction to the proceedings of the First Workshop on Parallel Problem Solving from Nature (Schwefel and Maenner, 1990).
2. Updates for the best (known) solutions can for example be found on http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

References

Affenzeller, M. (2001). "A New Approach to Evolutionary Computation: Segregative Genetic Algorithms (SEGA). Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence." *Lecture Notes of Computer Science* 2084, 594–601.

Affenzeller, M. (2001). "Transferring the Concept of Selective Pressure from Evolutionary Strategies to Genetic Algorithms." In *Proceedings of the 14th International Conference on Systems Science* 2, pp. 346–353.

Affenzeller, M. (2001). "Segregative Genetic Algorithms (SEGA): A Hybrid Superstructure Upwards Compatible to Genetic Algorithms for Retarding Premature Convergence." *Internatinal Journal of Computers, Systems and Signals (IJCSS)* 2(1), 18–32.

Affenzeller, M. (2002). "A Generic Evolutionary Computation Approach Based Upon Genetic Algorithms and Evolution Strategies." *Systems Science* 28(2), 59–71.

Affenzeller, M. and S. Wagner. (2003). "SASEGASA: An Evolutionary Algorithm for Retarding Premature Convergence by Self-Adaptive Selection Pressure Steering." Computational Methods in Neural Modeling, Lecture Notes of Computer Science—IWANN 2003. *Lecture Notes in Computer Science* 2686, 438–445.

Affenzeller, M. and S. Wagner. (2003). A Self-Adaptive Model for Selective Pressure Handling within the Theory of Genetic Algorithms." Computer Aided Systems Theory: EUROCAST 2003, *Lecture Notes in Computer Science* 2809, 384–393.

Alba, E. and J.M. Troya. (1999). "A Survey of Parallel Distributed Genetic Algorithms." *Complexity* 4(4), 31–52.

Baeck, T. (1993). "Selective Pressure in Evolutionary Algorithms: A Characterization of Selection Mechanisms." In *Proceedings of the First IEEE Conference on Evolutionary Computation* 1994, pp. 57–62.

Cantu-Paz, E. (2001). *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers.

Cobb, H.J. and J.J. Grefenstette. (1993). "Genetic Algorithms for Tracking Changing Environment." In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 523–530.

DeJong, K.A. (1975). "An Analysis of the Behaviour of a Class of Genetic Adaptive Systems." PhD thesis. Department of Computer Science. University of Michigan.

Dumitrescu, D., B. Lazzerini, L.C. Jain. and A. Dumitrescu. (2000). *Evolutionary Computation*. CRC Press.

Fogel, D.B. (1994). "An Introduction to Simulated Evolutionary Optimization." *IEEE Trans. on Neural Networks* 5(1), 3–14.

Goldberg, D.E. (1989). *Genetic Alogorithms in Search, Optimization and Machine Learning*. Addison Wesley Longman.

Gordon, V.S. and D. Whitley. (1993). "Serial and Parallel Genetic Algorithms as Function Optimizers." In *Proceedings of the Fifth International Conference on Genetic Algorithms. Morgan Kaufman*, pp. 177–183.

Hajek, B. (1988). "Cooling Schedules for Optimal Annealing." *Operations Research* 13, 311–329.

Hiroyasu, T., M. Miki, M. Hamasaki, and Y. Tanimura. (2000). "A New Model of Distributed Genetic Algorithm for Cluster Systems: Dual Individual DGA." *High Performance Computing. Springer, LNCS* 1940, 374–383.

Holland, J.H. (1975). *Adaption in Natural and Artificial Systems*. University of Michigan Press.

Larranaga, P., C.M.H. Kuijpers, R.H. Murga, I. Inza, and S. Dizdarevic. (1999). "Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators." *Artificial Intelligence Review* 13, 129–170.

Liang, K., X. Yao, and C. Newton. (2000). "Evolutionary Search of Approximated N-Dimensional Landscapes." *International Journal of Knowledge-Based Intelligent Engineering Systems* 4(3), 172–183.

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd edn. Berlin Heidelberg, New York: Springer-Verlag.

Mühlenbein, H., M. Schomisch, and J. Born. (1991). "The Parallel Genetic Algorithm as Function Optimizer." *Parallel Computing* 17, 619–631.

Potter, M.A. and K. De Jong. (1994). "A Cooperative Coevolutionary Approach to Function Optimization. Parallel Problem Solving from Nature—PPSN III." *LNCS* 866, pp. 249–257, Springer.

Rechenberg, I. (1973). *Evolutionsstrategie. Friedrich Frommann Verlag*.

Reinelt, G. (1991). "TSPLIB—A Traveling Salesman Problem Library." *ORSA Journal on Computing* 3, 376–384.

Schoeneburg, E., F. Heinzmann, and S. Feddersen. (1994). *Genetische Algorithmen und Evolutionsstrategien*. Addison-Wesley.

Schwefel, H.-P. (1994). *Numerische Optimierung von Computer-Modellen mittels Evolutionsstrategie*. Birkhäuser Verlag. Basel.

Smith, R.E., S. Forrest, and A.S. Perelson. (1993). "Population Diversity in an Immune System Model: Implications for Genetic Search." *Foundations of Genetic Algorithms* 2, 153–166.

Schwefel, H.-P. and R. Maenner. (Eds). (1999). *Proceedings of the First International Conference on Parallel Problem Solving from Nature (PPSN)*. Berlin Heidelberg, New York: Springer-Verlag.

Srinivas, M. and L. Patnaik. (1994). Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms. *IEEE Transactions on Systems, Man, and Cybernetics* 24(4), 656–667.

Takahashi, O., H. Kita, and S. Kobayashi. (1999). A Distance Dependent Alternation Model on Real-Coded Genetic Algorithms. *IEEE Transactions on Systems, Man, and Cybernetics* 619–624.

Yao, X. (1997). "A Diploid Genetic Algorithm for Preserving Population Diversity—Pseudo-Meiosis GA." *Lecture Notes in Computer Science* 866, 36–45.

Winkler, S., M. Attenzeller, and S. Wagner. (2004). "Identifying Nonlinear Model Structures Using Genetic Programming Techniques." Accepted to be published in: *Proceedings of the European Meeting on Cybernetics and Systems Research—EMCSR 2004*.

Yoshida, Y. and N. Adachi. (1994). "Global Optimisation by Evolutionary Algorithms." In *Proceedings of the Second AIZU International Symposium on Parallel Algorithms/Architecture Synthesis*. IEEE Computer Society Press, pp. 282–291.