Chapter 4

Preservation of Relevant Building Blocks

4.1 What Can Extended Selection Concepts Do to Avoid Premature Convergence?

The ultimate goal of the extended algorithmic concepts described in this chapter is to support crossover-based evolutionary algorithms, i.e., evolutionary algorithms that are ideally designed to function as building-block assembling machines, in their intention to combine those parts of the chromosomes that define high quality solutions. In this context we concentrate on selection and replacement which are the parts of the algorithm that are independent of the problem representation and the according operators. Thus, the application domain of the new algorithms is very wide; in fact, offspring selection and the RAPGA (a special variant of adaptive population sizing GA) can be applied to any application that can be treated by genetic algorithms (of course also including genetic programming).

The unifying purpose of the enhanced selection and replacement strategies is to introduce selection after reproduction in a way that checks whether or not crossover and mutation were able to produce a new solution candidate that outperforms its own parents. Offspring selection realizes this by claiming that a certain ratio of the next generation (pre-defined by the user) has to consist of child solutions that were able to outperform their own parents (with respect to their fitness values). The RAPGA, the second newly introduced selection and replacement strategy, ideally works in such a way that new child solutions are added to the new population as long as it is possible to generate unique and successful offspring stemming from the gene pool of the last generation. Both strategies imply a self-adaptive regulation of the actual selection pressure that depends on how easy or difficult it is at present to achieve evolutionary progress. An upper limit for the selection pressure provides a good termination criterion for single population GAs as well as a trigger for migration in parallel GAs.

4.2 Offspring Selection (OS)

As already discussed at length, the first selection step chooses the parents for crossover either randomly or in any other well-known way as for example roulette-wheel, linear-rank, or some kind of tournament selection strategy. After having performed crossover and mutation with the selected parents, we introduce a further selection mechanism that considers the success of the apparently applied reproduction. In order to assure that the progression of genetic search occurs mainly with successful offspring, this is done in such a way 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 is defined as the quotient of the next population members that have to be generated by successful mating in relation to the total population size. Our adaptation of Rechenberg's success rule ([Rec73], [Sch94]) for genetic algorithms 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 parent, the better parent, or some kind of weighted average of both?

In order to answer this question, we have borrowed an aspect from simulated annealing: The threshold fitness value that has to be outperformed lies between the worse and the better parent and the user is able to adjust a lower starting value and a higher end value denoted as comparison factor bounds; a comparison factor ($CompFactor$) of 0.0 means that we consider the fitness of the worse parent, whereas a comparison factor of 1.0 means that we consider the better of the two parents. During the run of the algorithm, the comparison factor is scaled between the lower and the upper bound resulting in a broader search at the beginning and ending up with a more and more directed search at the end; this procedure in fact picks up a basic idea of simulated annealing.

In the original formulation of the SASEGASA (which will be described in Chapter 5) we have defined that in the beginning of the evolutionary process an offspring only has to surpass the fitness value of the worse parent in order to be considered as "successful"; as evolution proceeds, the fitness of an offspring has to be better than a fitness value continuously increasing between the fitness values 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 using the success criterion defined above, the rest of the next generation ($(1 \cdot \cdot\ SuccRatio) \cdot |POP|$) is simply filled up with individuals randomly chosen from the pool of individuals that were also created by crossover, but did not reach the success criterion. The actual selection pressure $ActSelPress$ at the end of generation i 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:

$$ActSelPress = \frac{|POP_{i+1}| + |POOL_i|}{|POP_i|} \tag{4.1}$$

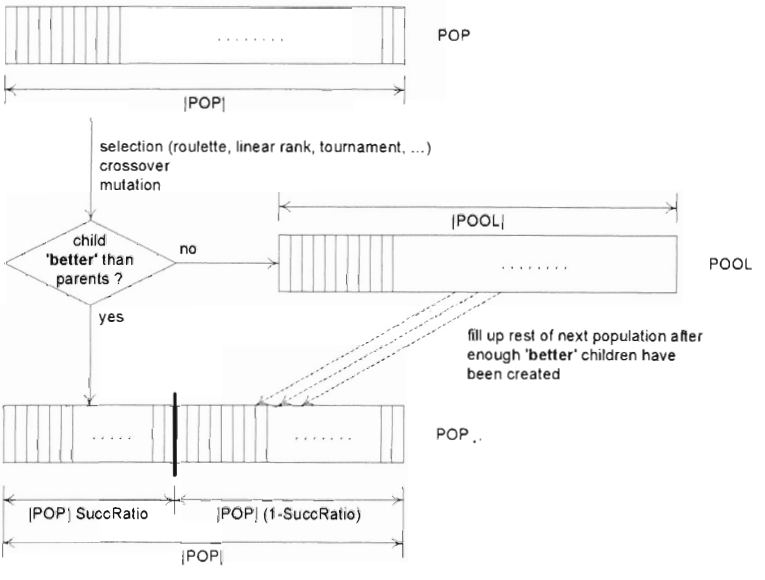

FIGURE 4.1: Flowchart of the embedding of offspring selection into a genetic algorithm. This figure is displayed with kind permission of Springer Science and Business Media.

Figure 4.1 shows the operating sequence of the concepts described above.

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 detector for premature convergence:

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.

As a basic principle of this selection model, higher success ratios cause higher selection pressures. Nevertheless, higher settings of success ratio, and therefore also higher selection pressures, do not necessarily cause premature convergence. The reason for this is mainly that the new selection step does not accept clones that emanate from two identical parents per definition. In

conventional GAs such clones represent a major reason for premature convergence of the whole population around a suboptimal value, whereas the new offspring selection works against this phenomenon (see Chapters 7, 10, and 11).

With all strategies described above, finally a genetic algorithm with the additional offspring selection step can be formulated as stated in Algorithm 4.1. The algorithm is formulated for a maximization problem; in case of minimization problems the inequalities have to be changed accordingly.

Algorithm 4.1 Definition of a genetic algorithm with offspring selection.

Initialize total number of iterations $nrOfIterations \in \mathbb{N}$
Initialize actual number of iterations $i = 0$
Initialize size of population $|POP|$
Initialize success ratio $SuccRatio \in [0, 1]$
Initialize maximum selection pressure $MaxSelPress \in]1, \infty[$
Initialize lower comparison factor bound $LowerBound \in [0, 1]$
Initialize upper comparison factor bound $UpperBound \in [LowerBound, 1]$
Initialize comparison factor $CompFactor = LowerBound$
Initialize actual selection pressure $ActSelPress = 1$
Produce an initial population POP_0 of size $|POP|$

while $(i < nrOfIterations) \wedge (ActSelPress < MaxSelPress)$ **do**
  Initialize next population POP_{i+1}
  Initialize pool for bad children $POOL$

  while $(|POP_{i+1}| < (|POP| \cdot SuccRatio)) \wedge ((|POP_{i+1}| + |POOL|) < (|POP| \cdot MaxSelPress))$ **do**
    Generate a child from the members of POP_i based on their fitness values using crossover and mutation

    Compare the fitness of the child c to the fitness of its parents par_1 and par_2 (without loss of generality assume that par_1 is fitter than par_2)
    if $f_c \leq (f_{par_2} + |f_{par_1} - f_{par_2}| \cdot CompFactor)$ **then**
      Insert child into $POOL$
    else
      Insert child into POP_{i+1}
    end if
  end while
  $ActSelPress = \frac{|POP_{i+1}| + |POOL|}{|POP|}$

  Fill up the rest of POP_{i+1} with members from $POOL$
  while $|POP_{i+1}| \leq |POP|$ **do**
    Insert a randomly chosen child from $POOL$ into POP_{i+1}
  end while
  Adapt $CompFactor$ according to the given strategy
  $i = i + 1$
end while

For a detailed analysis of the consequences of offspring selection the reader is referred to Chapter 7 where the characteristics of a GA incorporating offspring selection will be compared to the characteristics of a conventional GA on the basis of a benchmark TSP.

4.3 The Relevant Alleles Preserving Genetic Algorithm (RAPGA)

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

The here presented variant of enhanced algorithmic concepts based upon GA-solution manipulation operators aims to achieve this goal by trying to bring out as much progress from the actual generation as possible and losing as little genetic diversity as possible at the same time.

This idea is implemented using ad hoc population size adjustment in the sense that potential offspring generated by the basic genetic operators are accepted as members of the next generation if and only if they are able to outperform the fitness of their own parents and if they are new in the sense that their chromosome consists of a concrete allele alignment that is not represented yet in an individual of the next generation. As long as new and (with respect to the definition given previously) "successful" individuals can be created from the gene pool of the actual generation, the population size is allowed to grow up to a maximum size. A potential offspring which is not able to fulfill these requirements is simply not considered for the gene pool of the next generation.

Figure 4.2 represents the gene pool of the alleles at a certain generation i and Figure 4.3 illustrates how this genetic information can be used in order to generate a next population $i + 1$ of a certain size which may be smaller or larger than that of the actual population i. Whether the next population becomes smaller or larger depends on the success of the genetic operators crossover and mutation in the above stated claim to produce new and successful chromosomes.

For a generic, stable, and robust realization of these RAPGA ideas some practical aspects have to be considered:

- The algorithm should offer the possibility to use different settings also for conventional parent selection, so that the selection mechanisms for the two parents do not necessarily have to be the same. In many exam-

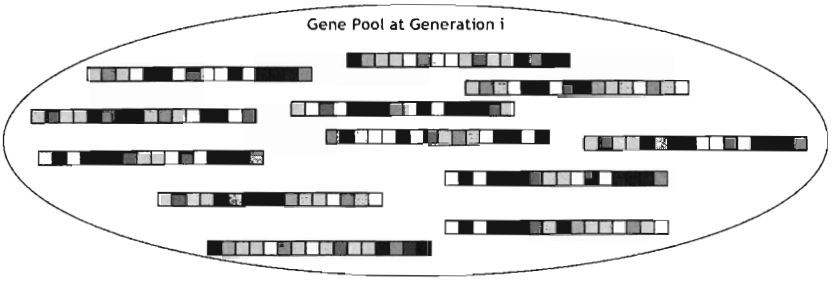

FIGURE 4.2: Graphical representation of the gene pool available at a certain generation. Each bar represents a chromosome with its alleles representing the assignment of the genes at the certain loci.

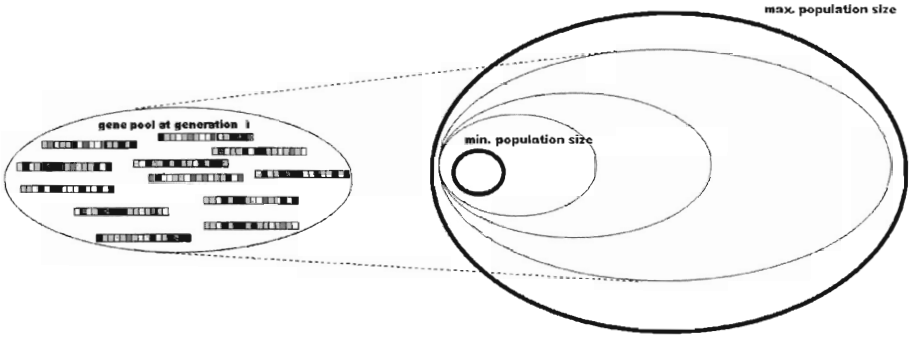

FIGURE 4.3: The left part of the figure represents the gene pool at generation i and the right part indicates the possible size of generation $i+1$ which must not go below a minimum size and also not exceed an upper limit. These parameters have to be defined by the user.

ples a combination of proportional (roulette wheel) selection and random selection has already shown a lot of potential (for example in combination with GP-based structure identification as discussed in [AWW08], e.g.). The two different selection operators are called male and female selection. It is also possible and reasonable in the context of the algorithmic concepts described here to disable parent selection totally, as scalable selection pressure comes along with the selection mechanisms after reproduction. This can be achieved by setting both parent selection operators to random.

- Due to the fact that reproduction results are only considered in case they are successful recombinations (and maybe mutations) of their parents' chromosomes, it becomes reasonable to use more than one crossover op-

erator and more than one mutation operator at the same time. The reason for this possibility is given by the fact that only successful offspring chromosomes are considered for the ongoing evolutionary process; this allows the application of crossover and mutation operators which do not produce good results mostly as long as they are still able to generate good offspring at least sometimes. On the one hand the insertion of such operators increases the average selection pressure and therefore also the average running time, but on the other hand these operators can help a lot to broaden evolutionary search and therefore retard premature convergence. If more than one crossover and mutation operator is allowed, the choice occurs by pure chance which has proven to produce better results than a preference of more successful operators [Aff05].

- As indicated in Figure 4.3, a lower as well as an upper limit of population size are still necessary in order to achieve efficient algorithmic performance. In case of a missing upper limit the population size would snowball especially in the first rounds which is inefficient; a lower limit of at least 2 individuals is also necessary as this indicates that it is no more possible to produce a sufficient amount of chromosomes that are able to outperform their own parents and therefore acts as a good detector for convergence.

- Depending on the problem at hand there may be several possibilities to fill up the next population with new individuals. If the problem representation allows an efficient check for genotypical identity, it is recommendable to do this and accept new chromosomes as members for the next generation if there is no structurally identical individual included in the population yet. If a check for genotypical identity is not possible or too time-consuming, there is still the possibility to assume two individuals are identical if they have the same fitness values as an approximative identity check. However, the user has to be aware of the fact that this assumption may be too restrictive in case of fitness landscapes with identical fitness values for a lot of different individuals; in such cases it is of course advisable to check for genotypical identity.

- In order to terminate the run of a certain generation in case it is not possible to fill up the maximally allowed population size with new successful individuals, an upper limit of effort in terms of generated individuals is necessary. This maximum effort per generation is the maximum number of newly generated chromosomes per generation (no matter if these have been accepted or not).

- The question, whether or not an offspring is better than its parents, is answered in the same way as in the context of offspring selection.

Figure 4.4 shows the typical development of the actual population size during an exemplary run of RAPGA applied to the ch130 benchmark instance of

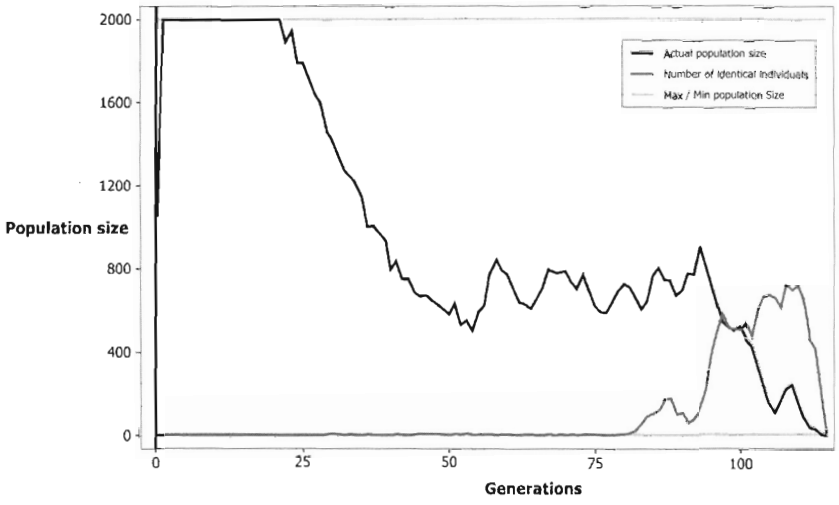

FIGURE 4.4: Typical development of actual population size between the two borders (lower and upper limit of population size) displaying also the identical chromosomes that occur especially in the last iterations.

the traveling salesman problem taken from the TSPLib [Rei91]. More sophisticated studies analyzing the characteristics of RAPGA will be presented in Chapter 7.

4.4 Consequences Arising out of Offspring Selection and RAPGA

Typically, GAs operate under the implicit assumption that parent individuals of above average fitness are able to produce better solutions as stated in Holland's schema theorem and the related building block hypothesis. This general assumption, which ideally holds under the restrictive assumptions of a canonical GA using binary encoding, is often hard to fulfill for many practical GA applications as stated in the questions of Chapter 3 which shall be rephrased and answered here in the context of offspring selection and RAPGA:

ad 1. *Is crossover always able to fulfill the implicit assumption that two above-average parents can produce even better children?*

Unfortunately, the implicit assumption of the schema theorem, namely that parents of above average fitness are able to produce even better children, is not accomplished for a lot of operators in many theoretical as well as practical applications. This disillusioning fact has several

reasons: First, a lot of operators tend to produce offspring solution candidates that do not meet the implicit or explicit constraints of certain problem formulations. Commonly applied repair strategies included in the operators themselves or applied afterwards have the consequence that alleles of the resulting offspring are not present in the parents which directly counteracts the building block aspect. In many problem representations it can easily happen that a lot of highly unfit child solution candidates arise even from the same pair of above average parents (think of GP crossover for example, where a lot of useless offspring solutions may be developed, depending on the concrete choice of crossover points). Furthermore, some operators have disruptive characteristics in the sense that the evolvement of longer building block sequences is not supported. By using offspring selection (OS) or the RAPGA the necessity that almost every trial is successful concerning the results of reproduction is not given any more; only successful offspring become members of the active gene pool for the ongoing evolutionary process.

ad 2. *Which of the available crossover operators is best suited for a certain problem in a certain representation?*

For many problem representations of certain applications a lot of crossover concepts are available where it is often not clear a priori which of the possible operators is suited best. Furthermore, it is often also not clear how the characteristics of operators change with the remaining parameter settings of the algorithm or how the characteristics of the certain operators change during the run of the algorithm. So it may easily happen that certain, maybe more disruptive operators perform quite well at the beginning of evolution whereas other crossover strategies succeed rather in the final (convergence) phase of the algorithm. In contrast to conventional GAs, for which the choice of usually one certain crossover strategy has to be done in the beginning, the ability to use more crossover and also mutation strategies in parallel is an important characteristic of OS-based GAs and the RAPGA as only the successful reproduction results take part in the ongoing evolutionary process. It is also an implicit feature of the extended algorithmic concepts that when using more operators in parallel only the results of those will succeed which are currently able to produce successful offspring which changes over time. Even the usage of operator concepts that are considered evidentially weak for a certain application can be beneficial as long as these operators are able to produce successful offspring from time to time [Aff05].

ad 3. *Which of the resulting children are "good" recombinations of their parents' chromosomes?*

Both OS and RAPGA have been basically designed to answer this question in a problem independent way. In order to retain generality, these

algorithms have to base the decision if and to which extent a given reproduction result is able to outperform its own parents by comparing the offspring's fitness with the fitness values of its own parent chromosomes. By doing so, we claim that a resulting child is a good recombination (which is a beneficial building block mixture) worth being part of the active gene pool if the child chromosome has been able to surpass the fitness of its own parents in some way.

ad 4. *What makes a child a "good" recombination?*

Whereas question 3 motivates the way, how the decision may be carried out whether or not a child is a good recombination of its parent chromosomes, question 4 intuitively asks why this makes sense. Generally speaking, OS and RAPGA direct the selection focus after reproduction rather than before reproduction. In our claim this makes sense, as it is the result of reproduction that will be part of the gene pool and that has to keep the ongoing process alive. Even parts of chromosomes with below average fitness may play an important role for the ongoing evolutionary process, if they can be combined beneficially with another parent chromosome which motivates gender specific parent selection [WA05b] as is for example applied in our GP experiments shown in the practical part (Chapter 11) of this book. With this gender specific selection aspect, which typically selects one parent randomly and the other one corresponding to some established selection strategy (proportional, linear-rank, or tournament strategies) or even both parents randomly, we decrease selection pressure originating from parent selection and balance this by increasing selection pressure after reproduction which is adjusted self-adaptively depending on how easy or difficult it is to achieve advancement.

ad 5. *Which parts of the chromosomes of parents of above-average fitness are really worth being preserved?*

Ideally speaking, exactly those parts of the chromosomes of above-average parents should be transferred to the next generation that make these individuals above average. What may sound like a tautology at the first view cannot be guaranteed for a lot of problem representations and corresponding operators. In these situations, OS and RAPGA are able to support the algorithm in this goal which is essential for the building block assembling machines GAs and GP.

Chapter 5

SASEGASA – More than the Sum of All Parts

The concept of offspring selection as described in Chapter 4 is very well suited to be transferred to parallel GA concepts. In the sense of parallel GA nomenclature, our proposed variant called SASEGASA (which stands for self-adaptive segregative genetic algorithm with simulated annealing aspects) is most closely related to the class of coarse-grained parallel GAs. The well-known island model supports global search by taking advantage of the steady pulsating interplay between breadth search and depth search supported by the forces of genetic drift and migration.

SASEGASA acts differently by allowing the certain subpopulations to drift a lot longer through the solution space; exactly until premature convergence is detected in each of the subpopulations, which will be denoted as local premature convergence. Then the algorithm aims very carefully to bring together the essential genetic information evolved in the certain demes individually.

Concretely, the following main distinguishing features can be pointed out comparing SASEGASA to a coarse-grained island model GA:

Dynamic Migration Intervals

Migration happens no longer in predefined fixed intervals but at those points in time when local premature convergence is detected in the certain subpopulations. The indicator of local premature convergence is the exceeding of a certain amount of selection pressure which can be measured in an offspring selection GA. New genetic information is then added from adjacent subpopulations that suffer from local premature convergence themselves, but have evolved different alleles due to the undirected forces of genetic drift. By this strategy the genetic search process can be initiated again until local premature convergence is detected next time.

From Islands to Growing Villages

The most important difference from SASEGASA to the well known island model is given by the fact that in case of SASEGASA the size of the subpopulations is slowly growing by decreasing the number of subpopulations. Therefore, the migration aspect of SASEGASA can rather be associated with

a village-town-city model than with an island model. By this means the certain (at the beginning rather small) villages can drift towards different regions of the search space until they are all prematurely converged. Then the total number of subpopulations is decreased by one and the individuals are regrouped as sketched in Figure 5.1; then the new subpopulations evolve independently until local premature convergence is detected again for each subpopulation. So the initially rather small villages become larger and larger towns, finally forming a large panmictic population. The main idea of this strategy is that the essential alleles which may be shared over many different villages can slowly and carefully be collected in a single population resulting in a high quality solution. As a consequence, parallelization is no more that efficient as in the island model, due to the changing number of subpopulations. More sophisticated communication protocols between the involved CPUs become necessary for efficient parallel implementations.

5.1 The Interplay of Distributed Search and Systematic Recovery of Essential Genetic Information

When applying GAs to higher dimensional problems in combinatorial optimization, it happens that genetic drift also causes alleles to fix to suboptimal 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 parental selection additionally promotes the drop out of those 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 population 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 (good reviews are given in [AT99] and [Alb05] for instance), we have 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 have

decided to introduce this concept of migration as it proved to support the localization of global optimal solutions to a greater extent [Aff05]. Of course, the concept of self-adaptive selection pressure steering is essential, especially immediately after 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 mechanisms which leads to more stability in terms of operators and migration rates, automated detection of migration interval, etc.

5.2 Migration Revisited

In nature the fragmentation of the population of a certain species into more subpopulations of different sizes is a commonly observable phenomenon. Many species have a great area of circulation of various environments which leads to the formation of subpopulations. An important consequence of the population structure is the genetic differentiation of subpopulations, i.e., the shift of allele frequencies in the certain subpopulations. The reasons for genetic differentiation are:

- Local adjustment of different genotypes in different populations
- Genetic drift in the subpopulations
- Random differences in the allele frequency of individuals which build up a new subpopulation

The structure of the population is hierarchically organized in different layers[1]:

- Individual
- Subpopulation
- Local population
- Entire population (species)

[1]The concept of hierarchical population structures has been introduced by Wright [Wri43].

An important goal of population genetics is the detection of population structures, the analyses of consequences and the location of the layer with most diverse allele frequencies. In this context a deeper consideration of genetic drift and its consequences is of major interest. The aspect of local adaptation of different genotypes in different populations should give useful hints for multi-objective function optimization or for optimization in changing environments.

One consequence of the population structure is the loss of heterozygosity (genetic variation). The Swedish statistician and geneticist Wahlund [HC89] described that genetic variation rises again, if the structure is broken up and mating becomes possible in the entire population. The SEGA and the SASEGASA algorithm, which will be described later, systematically take advantage of this effect step by step.

When talking about migration it is essential to consider some distinction concerning the genetic connection between the subpopulations which mainly depends on the gene flow (the exchange of alleles between subpopulations). Migration, the exchange of individuals, causes gene flow if and only if the exchanged individuals produce offspring. The most important effect of migration and gene flow is the introduction of new alleles into the subpopulations. In that sense migration has effects similar to mutation, but can occur at much higher rates. If the gene flow between subpopulations is high, they become genetically homogeneous; in the case of little gene flow, the subpopulations may diverge due to selection, mutation, and drift. Population genetics provides a set of models for the theoretical analysis of gene flows. The most popular migration models of population genetics are the mainland-island model, which considers migration in just one direction, and the island model, that allows migration in both directions. As discussed in parallel GA theory, the migration rate is an essential parameter for the description of migration. In population genetics the migration rate describes the ratio of chromosomes migrating among subpopulations.

5.3 SASEGASA: A Novel and Self-Adaptive Parallel Genetic Algorithm

We have already proposed several new EA-variants. The first prototype of this new class of evolutionary search which considers the concept of controllable selection pressure ([Aff01c], [Aff02) for information exchange between independently evolving subpopulations has been introduced with the Segregative Genetic Algorithm (SEGA) [Aff01a], [Aff01b]. Even if the SEGA is already able to produce very high quality results in terms of global solution quality, selection pressure has to be set by the user which is a very time con-

suming and difficult challenge. Further research, which aimed to introduce self-adaptive selection principles for the steering of selection pressure ([AW03], [AW04a]), resulted in the so-called SASEGASA-algorithm ([AW03], [Aff05]), which already represents a very stable and efficient method for producing high quality results without introducing problem-specific knowledge or local search.

So far parallelism has "only" been introduced for improving global solution quality and all experiments have been performed on single-processor machines. Nevertheless, there is nothing to be said against transforming the concepts evolved in the parallel GA community to also improve the quantitative performance of our new methods. Empirical studies have shown that the theoretical concepts of the SASEGASA-algorithm [AW03] allow global solution quality to be steered up to the highest quality regions by just increasing the number of demes involved. The algorithm turned out to find the global optimum for all considered TSP benchmarks as well as for all considered benchmark test functions up to very high problem dimensions [Aff05].

Therefore an enormous increase of efficiency can be expected when applying concepts of supercomputing, allowing us to also attack much higher dimensional theoretical and practical problems efficiently in a parallel environment. Because of the problem independency of all newly proposed theoretical concepts there is no restriction to a certain class of problems that allows the attack of all problems for which GA theory (and also GP theory) offers adequate operators.

5.3.1 The Core Algorithm

In principle, the SASEGASA introduces two enhancements to the basic concept of genetic algorithms. Firstly, the algorithm makes use of variable selection pressure, as introduced as offspring selection (OS) in Chapter 4, 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 so that the subpopulations are joined after local premature convergence has occurred. This is done in order to end up with a population including all genetic information sufficient for locating a global optimum.

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 (reunification) is to combat premature convergence which is the source of GA-difficulties. The basic properties (in terms of solution quality) of this segregation and reunification approach have already proven their potential in overcoming premature convergence [Aff01a], [Aff01b] in the so-called SEGA algorithm.

By using this approach of breadth search, essential building blocks can evolve independently in different regions of the search space. In the case of standard GAs those relevant building blocks are likely to disappear early on due to genetic drift and, therefore, their genetic information can not be

provided at a later phase of evolution, when the search for a global optimum 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 [Aff01b], 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.

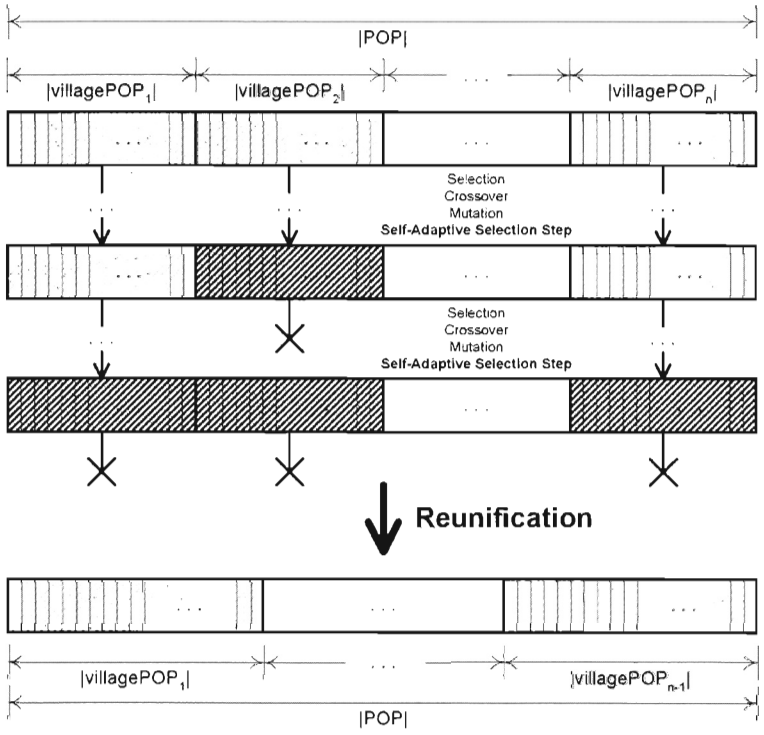

FIGURE 5.1: Flowchart of the reunification of subpopulations of a SASEGASA (light shaded subpopulations are still evolving, whereas dark shaded ones have already converged prematurely). This figure is displayed with kind permission of Springer Science and Business Media.

Equipped with offspring selection we have both: A self-adaptive selection pressure (depending on the given success ratio), as well as an automated detection of local premature convergence, if the current selection pressure becomes higher than the given maximal selection pressure parameter ($MaxSelPress$). Therefore, a date of reunification has to be set, if local premature convergence

has occurred within all subpopulations, in order to increase genetic diversity again. Figure 5.1 shows a schematic diagram of the migration policy in the case of a reunification phase of the SASEGASA algorithm. The dark shaded subpopulations stand for already prematurely converged subpopulations. If all subpopulations are prematurely converged (dark shaded) a new reunification phase is initiated.

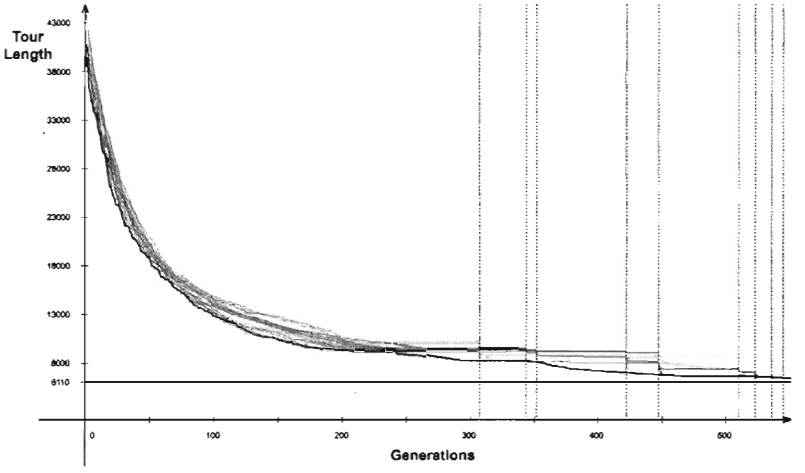

FIGURE 5.2: Quality progress of a typical run of the SASEGASA algorithm. This figure is displayed with kind permission of Springer Science and Business Media.

Figures 5.2 and 5.3 show typical shape of the fitness curves and selection pressure progresses of a SASEGASA test run. The number of subpopulations is in this example set to 10. The vertical lines indicate dates of reunification. In the quality diagram (Figure 5.2) the lines give the fitness value of the best member of each deme; the best known solution is represented by the horizontal line. In the selection pressure diagram (shown in Figure 5.3) 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 lower horizontal line 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

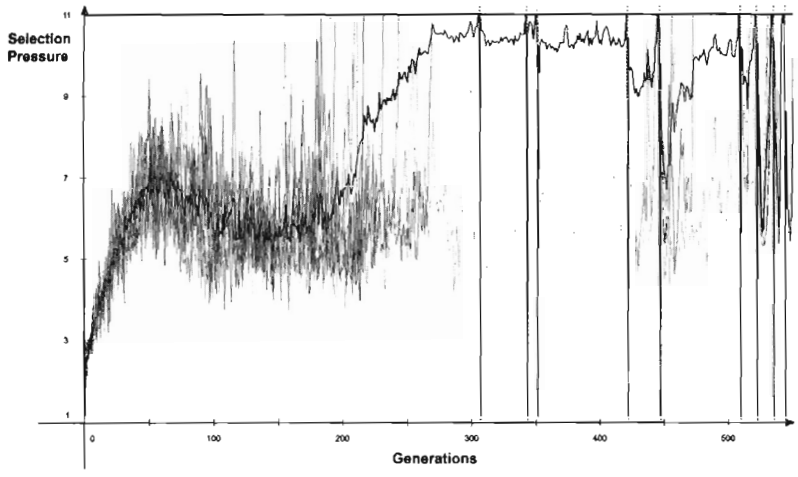

FIGURE 5.3: Selection pressure curves for a typical run of the SASEGASA algorithm. This figure is displayed with kind permission of Springer Science and Business Media.

can be stated as described in Figure 5.4.

Again, similar as in the context of offspring selection, 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.

5.4 Interactions among Genetic Drift, Migration, and Self-Adaptive Selection Pressure

Using all the introduced generic algorithmic concepts combined in SASEGASA, it becomes possible to utilize the interactions between genetic drift and the SASEGASA specific dynamic migration policy in a very advantageous way in terms of achievable global solution quality:

Initially a certain number of subpopulations evolve absolutely independently from each other until no further evolutionary improvement is possible when using a genetic algorithm with offspring selection in the subpopulations, i.e., until local premature convergence is detected in all subpopulations.

As a matter of principle, it is also the case that primarily those alleles are fixed in the certain subpopulations which currently influence the fitness

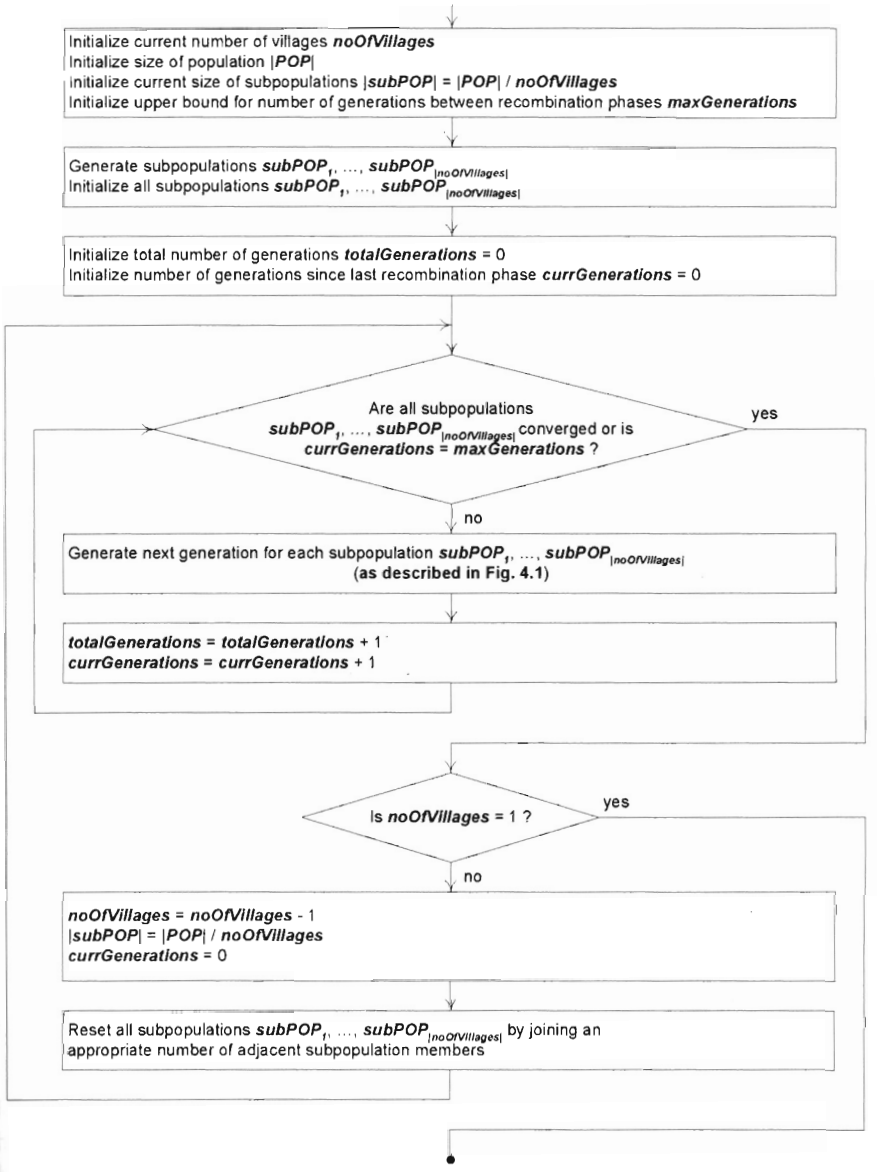

FIGURE 5.4: Flowchart showing the main steps of the SASEGASA. This figure is displayed with kind permission of Springer Science and Business Media.

function to a greater extent. The rest of the essential alleles with a currently low influence on the fitness value, which may be required for a global optimal solution later, are unlikely to be combined in any single subpopulation –

because basically these alleles are distributed over all subpopulations.

As after this first optimization stage all subpopulations are prematurely converged to solutions with comparable fitness values, genetic information that has not been considered before is suddenly taken into account by selection after a reunification phase. We so establish a step-by-step reunification of independently evolving subpopulations, which is triggered by the detection of convergence; in this way it becomes possible to systematically consider the essential alleles at exactly those points in time when they are in the position to become accepted in the newly emerging greater subpopulations. By means of this procedure smaller building blocks[2] of a global optimal solution are step-by-step enriched with new alleles, in order to evolve to larger and larger building blocks ending up in one single subpopulation, containing all genetic information being required for a global optimal solution.

With an increasing number of equally sized subpopulations, the probability that essential alleles are barred from dying off increases due to the greater total population size; this results in a higher survival probability of the alleles which are not yet considered.

As empirically demonstrated in Chapters 7, 10, and 11, the procedure described above makes it possible to find high quality solutions for more and more difficult problems by simply increasing the number of subpopulations. Of course this causes increasing computational costs; still, this growth in computational costs is not exponential, but linear.

[2]The notation of building blocks is considered in a more general interpretation than in Holland's definition.