A Species Conserving Genetic Algorithm for Multimodal Function Optimization

Jian-Ping Li Jian-Ping.Li@umist.ac.uk
Department of Mechanical, Aerospace and Manufacturing Engineering, UMIST, PO Box 88, Manchester M60 1QD, UK

Marton E. Balazs BALAZSM@Richmond.ac.uk
Department of Computing, Mathematics and Sciences, Richmond the American International University in London, Queens Road, Richmond upon Thames TW10 6JP, UK

Geoffrey T. Parks gtp@eng.cam.ac.uk
Engineering Design Centre, Cambridge University Engineering Department, Trumpington Street, Cambridge CB2 1PZ, UK

P. John Clarkson pjc10@eng.cam.ac.uk
Engineering Design Centre, Cambridge University Engineering Department, Trumpington Street, Cambridge CB2 1PZ, UK

Abstract

This paper introduces a new technique called species conservation for evolving parallel subpopulations. The technique is based on the concept of dividing the population into several species according to their similarity. Each of these species is built around a dominating individual called the species seed. Species seeds found in the current generation are saved (conserved) by moving them into the next generation. Our technique has proved to be very effective in finding multiple solutions of multimodal optimization problems. We demonstrate this by applying it to a set of test problems, including some problems known to be deceptive to genetic algorithms.

Keywords

Genetic algorithms, multimodal functions, niching, species, species conservation.

1 Introduction

Over the years, *genetic algorithms* (GAs) have been proven effective in solving a variety of search and optimization problems (Goldberg, 1989; Gen and Cheng, 1997; Parmee, 1999). When attempting to optimize a multimodal function, the *Simple Genetic Algorithm* (SGA) converges to a single solution (Goldberg and Richardson, 1987). The intrinsic parallelism in a GA suggests, however, that this method should be able to locate several optima of a multimodal function.

There are two good, practical reasons why it may be desirable to locate multiple optima of an optimization problem. First, by encouraging the GA to locate multiple optima, the chances of locating the global optimum may be improved. Second, in a design context, identifying a diverse set of high-quality solutions will provide the designer with insight into the nature of the design space and, perhaps, suggest innovative alternative solutions.

Only a limited amount of research has been done into the use of GAs to locate multiple optima of a multimodal function. The techniques developed for solving problems of this type fall into two broad categories: iterative methods and parallel subpopulation methods.

Iterative methods address the problem of locating multiple optima of a multimodal function by repeatedly applying the same optimization algorithm. To prevent repeated convergence to the same solution, iterative methods use various techniques to prohibit the underlying optimization method from searching again those portions of the search space that have already been explored. Tabu Search (Glover, 1989) and the Sequential Niche Technique (Beasley et al., 1993) fall into this category.

Parallel subpopulation methods attempt to produce multiple solutions to a multimodal optimization problem by dividing the population into subpopulations that evolve in parallel. If there is no communication between these subpopulations, such a method is equivalent to iterating the evolution of a single, smaller population several times (Beasley et al., 1993). In consequence, parallel subpopulation methods use some communication between the subpopulations to allow "good characteristics" of individuals to spread.

One important class of parallel subpopulation methods is *island model parallel GAs* (IMGAs) (Gordon et al., 1992). IMGAs exploit the concept of punctuated equilibria in an evolutionary computation context (Cohoon et al., 1987). In IMGAs the overall population is partitioned into subpopulations, which evolve in isolation for a period known as an epoch. At the end of each epoch, solutions migrate between subpopulations. Migration must be carefully controlled, as communication between subpopulations has potential drawbacks, such as the reduction of diversity of solutions (Davidor, 1991). A variety of migration schemes, which are well reviewed by Martin et al. (1997), have been developed. IMGAs have been successfully applied to a variety of design problems (Cohoon et al., 1991; Lienig and Thulasiraman, 1993; Eby et al., 1999), but, by their nature, they introduce a number of additional control parameters that need careful selection to ensure good algorithm performance.

This paper introduces *species conservation*, a new technique for evolving parallel subpopulations. This technique introduces just one control parameter, the *species distance*, in addition to those needed to control any GA.

To describe our technique we will consider unconstrained optimization problems of real-valued functions, defined over arrays of real numbers. Where no confusion could occur we denote the objective function by f. The GA using species conservation (SCGA) presented in this paper makes no distinction between genotypes and phenotypes. Thus, genetic operators will be applied directly to individuals represented by arrays of real numbers. Note that none of the above restrictions are required for our technique to be applicable. The only reason for imposing them is for simplicity of presentation.

The next section describes the related work that is relevant to our proposed technique. Section 3 introduces the species conservation technique and describes the algorithm that implements it. Section 4 presents the results from a series of experiments on a set of test functions, comparing our results with other results reported in the literature where this is possible. Section 5 draws some conclusions and proposes further directions of research.

2 Related Work

The preservation of good individuals from one generation to the next and the maintaining of diversity are two very important and apparently contradictory requirements when applying GAs to multimodal optimization problems. In this section we briefly review the methods developed to address these issues that are most relevant to our research: elitism and niching.

2.1 Elitism

It is important to prevent promising individuals from being eliminated from the population during the application of genetic operators. To ensure that the best chromosome is preserved, elitist methods copy the best individual found so far into the new population (De Jong, 1975).

Different GA variants achieve this goal of preserving the best solution in different ways. For instance, Whitley's GENITOR (Whitley, 1989) creates just one child each cycle which then replaces the worst member of the population. In Eshelman's CHC (Eshelman, 1991) the offspring and parent populations are merged and the best M (the population size) individuals selected.

The effects of elitism have been widely researched, and it is used in most GA implementations (Gen and Cheng, 1997). However, "elitist strategies tend to make the search more exploitative rather than explorative and may not work for problems in which one is required to find multiple optimal solutions" (Sarma and De Jong, 1997).

2.2 Evolving Parallel Subpopulations by Niching

In common with other techniques used in evolutionary computation, the idea of niching was inspired by nature. In natural ecosystems there are many different ways in which individuals may survive by taking on different roles. Each of these roles is called an ecological niche. Rather than evolving a single population of identical (or very similar) individuals, ecosystems evolve subpopulations to fill different niches.

Niching was introduced into GAs primarily to maintain the diversity in a population. Later the same techniques were extended to design GAs capable of retrieving multiple optimal solutions.

A number of means of implementing niching in GAs have been devised (Deb and Goldberg, 1989; Goldberg, 1989; Beasley et al., 1993). In the literature Cavicchio's (1970) dissertation was one of the first studies to attempt to induce niching behavior in a GA by introducing a mechanism called preselection. Preselection is a tournament approach where a child replaces an inferior parent and induces niching by letting similar individuals compete for a place in the population.

De Jong (1975) generalized Cavicchio's preselection technique in a scheme he called crowding. In crowding an individual is compared to a randomly drawn subpopulation and the most similar member of that subpopulation is replaced. Later two further variants of crowding, deterministic crowding (Mahfoud, 1995) and probabilistic crowding (Mengshoel and Goldberg, 1999), were introduced. Both of these use Boltzmann tournament selection (Goldberg, 1990) for handling children and parents. The main difference between the two is that the former uses a deterministic acceptance rule, while the latter uses a probabilistic one. Recently Hughes and Leyland (2000) have used a GA with a fixed number of species to locate the radar scattering centers in a missile-target engagement simulator.

Another way of inducing niching behavior in a GA is to use fitness sharing. Goldberg and Richardson (1987) used Holland's sharing concept (Holland, 1975) to divide

the population into different subpopulations according to the similarity of the individuals. They introduced a sharing function that defined the degradation of the fitness of an individual due to the presence of neighboring individuals. The sharing function is used during selection. Its effect is such that when many individuals are in the same neighborhood they degrade each other's fitness values, thus limiting the uncontrolled growth of a particular species.

Spears (1994) used tag bits to identify different species — individuals with the same tag bit value belong to the same species. The tag bits are used to restrict mating and to perform fitness sharing, and can be changed, i.e., a solution can change species, through mutation.

All the niching techniques we have found described in the literature try to give all local or global optimal solutions an equal opportunity to survive. Sometimes, however, survival of low fitness but very different individuals may be as, if not more, important than that of some highly fit ones. The purpose of this paper is to present a new technique called species conservation that addresses this problem. We show that using this technique, a simple GA will converge to multiple solutions of a multimodal optimization problem.

3 Species Conservation

The technique for multimodal function optimization presented in this paper achieves niching by exploiting the notion of species.

A species is a class of individuals with common characteristics. In their approach to niching by fitness sharing, Goldberg and Richardson (1987) divided the population into different subpopulations according to the similarity of the individuals. They used the Euclidean distance between two individuals to measure their dissimilarity. The larger the distance between two individuals, the more dissimilar they are.

The distance between two individuals $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{in}]$ and $\mathbf{x}_j = [x_{j1}, x_{j2}, ..., x_{jn}]$ was defined by:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \tag{1}$$

Note that this is not the only way in which the distance, and hence the dissimilarity, between two individuals represented by vectors of real numbers could be defined.

In this paper we use the above definition of distance to characterize the dissimilarity between two individuals, but the method we describe will work for other distance definitions as well.

3.1 The Definition of a Species

Our definition of a species, as well as the operation of the SCGA, depends on a parameter we call the species distance, which we denote by σ_s. The species distance specifies the upper bound on the distance between two individuals for which they are considered to be similar. In our approach we propose that the species distance also be used to determine which individuals are worth preserving from one generation to the next.

In this work we define a species with respect to a finite population $P_N = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$. This is an intuitively reasonable view, because a species consists of actual individuals and is by no means just a region of feasible space. One consequence of this is that we do not define a notion of *absolute species*, that is, species defined only with respect to feasible space and the objective (fitness) function.

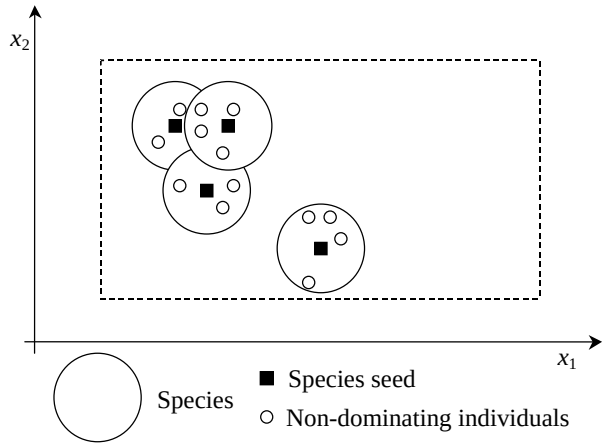

Figure 1: A sample distribution of species in a two-dimensional domain.

A species will be a subset S_i of P_N in which the distance between any two individuals is less than the species distance. Note that we do not require that any two individuals satisfying the condition that the distance between them is less than the species distance belong to the same species.

There are obviously many ways in which a population can be partitioned into species. In this paper we propose to construct the species in a population around certain individuals we call *dominating individuals* or *species seeds*.

Let $S_1, S_2, ..., S_k$ be a partitioning of population P_N into species. An individual $\mathbf{x}^* \in S_i$ is a dominating individual in its species if, for every individual $\mathbf{y} \in S_i$,

$$f(\mathbf{x}^*) \geq f(\mathbf{y}) \tag{2}$$

Note that the equality in this relation ($\geq$ rather than $>$) means that there may be more than one dominating individual in a species.

A species S_i is *centered on* its dominating individual $\mathbf{x}^*$ if, for every individual $\mathbf{y} \in S_i$,

$$d(\mathbf{x}^*, \mathbf{y}) \leq \sigma_s/2 \tag{3}$$

It is important to note that this definition does not mean that if a species S_i is centered on its dominating individual $\mathbf{x}^*$, all the individuals within a distance $\sigma_s/2$ of $\mathbf{x}^*$ are in S_i. To illustrate this, Figure 1 shows a possible partitioning of a population in a two-dimensional domain into species centered on their dominating individuals.

It should be clear that according to the above definitions, a dominating individual may dominate several species — indeed a single globally optimal individual will dominate all species. Nevertheless, the notions of a dominating individual and of a species dominated by a dominating individual are helpful to our technique, as they allow us to define a criterion for selecting individuals for conservation.

3.2 The Need for Conservation

In natural environments some species may become extinct because they could not adapt to a changing environment. Nevertheless, they may be useful (to humanity or to

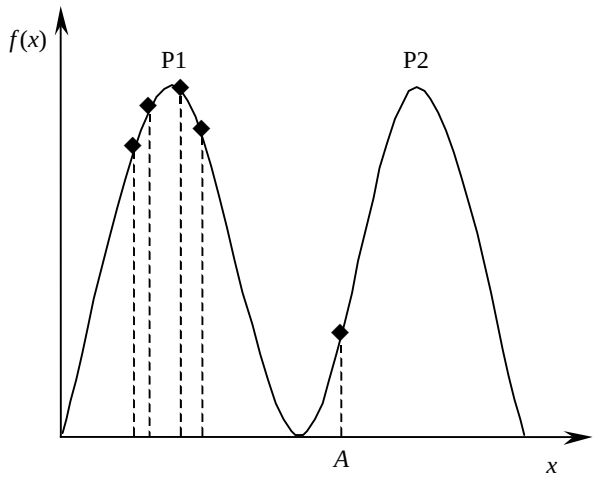

Figure 2: An illustration of the need for species conservation.

some other ecosystems) in the future. Under these conditions one could intervene to preserve a few individuals.

Introducing elitism into a GA will often lead to one highly fit individual gradually replacing all competing rivals. Thus, in addition to introducing the possibility of premature convergence, elitism also prevents a GA from finding multiple optima of a multimodal optimization problem. The question we seek to answer is: "How can the idea of elitism be transferred to a GA using some form of niching in such a way that several possibly useful individuals are copied into the next generation (conserved)?"

The answer to this question is not trivial, because there may be several reasons why an individual should be conserved.

First, we would like to conserve highly fit individuals. The question that needs to be answered here is: "Which individuals should be conserved?" Fitness sharing addresses this issue by adjusting the fitness of individuals based on the size of the niche they are in. While this gives a higher probability of selection to individuals belonging to smaller niches (or even to isolated individuals), it does not guarantee their survival.

Second, we would like to conserve individuals that, while not highly fit, are different enough from the current best individuals to be worth keeping.

Suppose, for example, that our problem is to maximize the function shown in Figure 2, and assume that, after some iterations of the GA, the population consists of the points shown. Most of the individuals have converged on peak P1, but at the same time, a low fitness individual A has appeared. Because the fitness of A is very small, the probability of this individual surviving to the next generation is low. This is true even for a GA using fitness sharing, unless a sharing function is specifically designed for this problem. However, this individual is very important to the search, if the other global optimum (P2) is to be found. In order to preserve the "good quality" of A, that of being far from the other members of the population and thus possibly close to an unexplored region, we need a special method to conserve it.

```
begin
    X_S = ∅;
    while (no more unmarked individuals in G(t)) do
        Search for the best unmarked x* ∈ G(t);
        Mark x* as processed;
        found ← FALSE;
        for all x ∈ X_S do
            if (d(x*, x) ≤ σ_s/2) then
                found ← TRUE;
                break;
            end (if)
        end (for)
        if (not found) then
            let X_S ← X_S ∪ {x*};
        end (if)
    end (while)
end
```

Figure 3: The algorithm for determining the species seeds.

None of the techniques we have encountered in the literature address this problem.

We propose to solve the problems described above by partitioning the population into a set of dominated species and copying the dominating individual of each of these species (the species seeds) into the next generation.

3.3 Determining the Species in a Population

To determine the individuals that will be copied into the next generation, we need to partition the current generation into a set of dominated species and determine the dominating individual in each of these species. In Figure 3 we show the algorithm that accomplishes this. In the algorithm, X_S denotes the set of species seeds found in generation $G(t)$.

The algorithm builds the set X_S by successively considering each of the individuals in $G(t)$, in decreasing order of fitness. When an individual is considered, it is checked against the species seeds found so far. If X_S does not contain any seed that is closer than half the species distance $(\sigma_s/2)$ to the individual considered, then the individual will be added to X_S.

The procedure for determining the species seeds has to be performed for every generation in a GA run, and, as a consequence, it will introduce some overhead to the computation. The complexity of this additional computation can be characterized in terms of the number of times a distance between two individuals needs to be evaluated. We can analyze this complexity by considering the algorithm given in Figure 3:

- The **while** loop is executed for each individual in the population, that is, a total of N times.
- Assume that, when considering the ith individual, X_S contains r_i species seeds. In this case the **for** loop will be performed at best 1 and at worst r_i times. The former will happen when the individual considered is within a distance $\sigma_s/2$ of the best

```
begin
    Mark all individuals as unprocessed;
    for all x ∈ X_S do
        Select the worst unmarked y ∈ S'(x, σ_s);
        if (y exists) then
            if (f(y) < f(x)) then
                y = x;
            end (if)
        else
            Select the worst unmarked y ∈ G(t + 1);
            y = x;
        end (if)
        Mark y as processed;
    end (for)
end
```

Figure 4: The algorithm for conserving species.

individual, while the latter will happen if the individual considered is not within a distance $\sigma_s/2$ of any of the species seeds found so far.

- For a given i the value of r_i will range from 0 (no seeds have yet been found) to $i-1$ (all the individuals considered so far were found to be species seeds).

The number of distance evaluations performed when determining the species seeds in one generation $T_s(N)$ can thus be characterized by the following relation:

$$N \leq T_s(N) \leq \sum_{i=1}^{N}(i-1) = \frac{N(N-1)}{2} \tag{4}$$

That is, the number of distance evaluations performed for each generation is $O(N^2)$.

For the algorithm given in Figure 3 we can give a much tighter upper bound on the number of distance evaluations required when determining the species seeds in a generation by taking into account the value of the species distance σ_s. First, let us note that the actual upper bound is given by:

$$N \leq T_s(N) \leq \sum_{i=1}^{N} N_S = N_S \cdot N \tag{5}$$

where N_S is the number of species that will be found for the generation under consideration. An upper bound on N_S can be given based on the size of the search space and σ_s. If we denote this upper bound by $\overline{N}_\sigma$, the above inequalities become

$$N \leq T_s(N) \leq \overline{N}_\sigma \cdot N \tag{6}$$

This means that, in practice, the overhead for finding species seeds is of linear order with the coefficient of the linear term depending on σ_s.

The average complexity is much harder to calculate, because we cannot make any assumptions about the distribution of the individuals in a generation.

```
begin
    t = 0;
    Initialize G(t);
    Evaluate G(t);
    while (not termination condition) do
        Identify species seeds X_S;
        Select G(t + 1);
        Crossover G(t + 1);
        Mutate G(t + 1);
        Evaluate G(t + 1);
        Conserve species from X_S in G(t + 1);
        t = t + 1;
    end (while)
    Identify species seeds X_S;
    Identify global optima;
end
```

Figure 5: The structure of the SCGA.

3.4 Conserving Species

Once all the species have been found, the new population is constructed by applying the usual genetic operators: selection, crossover, and mutation. Since some species may not survive following these operations, we copy them into the new population and thus enable them to survive. The species conservation process used is shown in Figure 4 and works as follows:

- The new generation $G(t+1)$ is searched for solutions belonging to the same species (designated $S'(\mathbf{x}, \sigma_s)$) as each species seed $\mathbf{x} \in X_S$ identified in generation $G(t)$, i.e., the solutions $\mathbf{y} \in G(t+1)$ for which $d(\mathbf{x}, \mathbf{y}) < \sigma_s/2$ are found.
- Species seed $\mathbf{x}$ replaces the worst of these "similar" solutions as long as it is better (fitter).
- If there are no solutions in the same species as $\mathbf{x}$ in $G(t+1)$, $\mathbf{x}$ replaces the worst unmarked solution in $G(t+1)$.
- As the species seeds are drawn from the previous generation, the number of species seeds N_S is always less than the population size N, and therefore unmarked solutions must always exist.

Thus, all the species seeds are either conserved or superseded by better examples of the same species.

Note that it is possible that none of the species seeds selected for conservation will be copied into the next generation. However, this will only happen if the new generation created by applying the genetic operators contains at least one individual from each of the species defined by the seeds in X_S, and if each of these individuals has higher fitness than the corresponding species seed.

Conserving the species seeds adds another overhead to the computations performed. Again, we can characterize the complexity of this overhead by the number

of times a distance between two individuals is computed. Considering the algorithm given in Figure 4:

- The **for** loop is executed for each species seed, that is, a total of N_S times.
- When the ith species seed is conserved, there are $i-1$ marked individuals in the population. Therefore $N-(i-1)$ distance evaluations are required between the $N-(i-1)$ unmarked members of the population and the species seed $\mathbf{x}$ to identify the members of the species $S'(\mathbf{x}, \sigma_s)$ and hence the worst individual in this species.

The number of distance computations performed when conserving the species seeds in one generation $T_c(N)$ can be characterized by the following relation:

$$N < T_c(N) = \sum_{i=1}^{N_S}(N-i+1) = N_S\left[N - \frac{1}{2}(N_S-1)\right] < N_S \cdot N \leq \overline{N}_\sigma \cdot N \qquad (7)$$

Differentiating this expression for $T_c(N)$ with respect to N_S, it is easily shown that $T_c(N)$ is maximized when $N_S = N + \frac{1}{2}$. N_S must be an integer, in practice, and, as the species seeds are identified from the population, cannot exceed N. Hence, substituting $N_S = N$ in Equation 7:

$$T_c(N) \leq \frac{1}{2}N(N+1) \qquad (8)$$

Thus, combining the results in Equations 6 and 7, the total overhead introduced by our species determination and conservation processes, as measured by the number of distance calculations performed per generation, $T_{sc}(N) = T_s(N) + T_c(N)$, will be:

$$2N \leq T_{sc}(N) \leq 2N_S \cdot N \leq 2\overline{N}_\sigma \cdot N \qquad (9)$$

Thus, the complexity of the number of distance computations performed for each generation is between $O(N)$ and $O(N^2)$. If the species distance σ_s is set small, there are potentially many species seeds in each generation, and the complexity is $O(N^2)$; if σ_s is sufficiently large, there are few species seeds, and the complexity is approximately $O(N)$. Overall, even with a large number of species seeds, this overhead is no worse than the overhead introduced by any other parallel subpopulation techniques.

The reader may ask the obvious question: "Why would we want to perform this complicated procedure instead of simply copying the species seeds into the new generation before the application of the genetic operators?" The rationale for doing this is that we want to allow a newly generated individual similar to a conserved species seed to replace that seed in the new generation. This will result in each species containing individuals of increasing (or at least not lower) fitness from one generation to the next.

Obviously just copying the conserved species seeds into the new generation and then "filling it up" with individuals created using genetic operators is also a viable approach. We still need to investigate whether this alternative approach affects the behavior of the SCGA significantly.

3.5 The SCGA

In this section we present the structure of the SCGA. The algorithm is based on the structure of a classical SGA and is shown in Figure 5.

The only significant differences between the SCGA and the SGA are that:

- Within the generation loop, first the species seeds are determined;

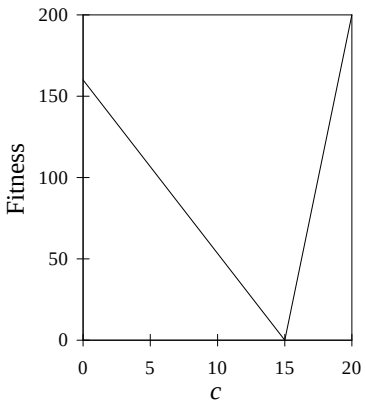

Figure 6: The two-peak trap function.

- After the genetic operators have been applied and the population evaluated, the species conservation process is performed, and, thus, the new generation is constructed.

3.6 Identifying Global Optima

There is one more question that we need to answer with respect to the operation of the SCGA: "How are the global optima[1] (the solutions to the optimization problem) identified after the generation loop is exited?"

First, recall that in X_S we collected all the dominating individuals that were sufficiently different from each other. This suggests that X_S should contain (among other individuals) the global optima found, if any. Unfortunately, X_S may contain both low fitness individuals that were stored because they were sufficiently different from all the other individuals in the previous generation, and high (but not necessarily equal) fitness individuals. This leaves us with only one option — to select from X_S those individuals that have a "high enough" fitness.

In our implementation of the SCGA we consider that the global optima to the problem at hand are the most fit individuals in X_S; these being all the individuals in X_S that have a fitness "close to" the fitness $f_{\max}$ of the most fit species seed. For this purpose we define a *solution acceptance threshold* r_f ($0 < r_f \leq 1$) and identify as global optima all the individuals $\mathbf{x} \in X_S$ that satisfy the following inequality:

$$f(\mathbf{x}) \geq (f_{\max} - f_{\min}) \times r_f \tag{10}$$

where $f_{\min}$ is the minimal fitness in the final population. Identifying the global optima in this way is straightforward, since the individuals are stored in X_S in decreasing order of fitness. As r_f is used in post-processing the results of the SCGA run, its value has no effect on algorithm performance. In processing these results the user can interactively

[1]Here we use "global optima" to mean "dissimilar, high-quality solutions." If one is tackling a multimodal optimization problem with equal-valued optima, then one would probably be seeking to find the true global optima. If, as is much more likely to be the case in a design context, one is tackling a problem the optimal solutions of which are not known, then one would be interested in identifying dissimilar, high-quality solutions.

Table 1: Comparison of the performance of different algorithms applied to the two-peak trap function.

Algorithm	Success rate	Evaluations expected
Iterated Genetic Search (Ackley, 1987)	0.00%	> 1000000
Stochastic Iterated Genetic Hillclimbing (SIGH)	100%	780
Iterated traditional GA (Beasley et al., 1993)	0.01%	2300000
Sequential Niche GA (SNGA)	77.60%	4900
SCGA	100%	935

adjust r_f. The values of r_f quoted in Section 4 are therefore included merely to enable others to reproduce our results.

3.7 Choosing the Parameter σ_s

The species distance parameter σ_s introduced in our technique plays a crucial role in our definition of the species and species seeds. If we choose it too small, many species may be found in every generation. This will increase the overhead associated with our technique, thus reducing efficiency, without necessarily increasing effectiveness. On the other hand, a large value of σ_s will make many solutions indistinguishable from the point of view of species conservation. This means that too few species will be conserved. If σ_s is so large that only one species seed is conserved, the SCGA will degenerate into an elitist SGA with all its disadvantages (plus the overhead introduced by species conservation).

Deb and Goldberg (1989) proposed that the species distance be estimated on the following basis. The radius of the smallest hypersphere containing feasible space is given by:

$$r = \frac{1}{2}\sqrt{\sum_{k=1}^{n}\left(x_k^u - x_k^l\right)^2} \tag{11}$$

where $x_1^l, x_2^l, ..., x_n^l$ and $x_1^u, x_2^u, ..., x_n^u$ are respectively the lower and upper bounds on the control variables. If each species (or niche) is enclosed by an n-dimensional hypersphere of radius σ_s, and if we know N_g, the number of global optima, the species distance can be estimated as:

$$\sigma_s = \frac{r}{\sqrt[n]{N_g}} \tag{12}$$

Even in the context of Deb and Goldberg's work, this approach has a very strong limitation — it can only be used in cases where the global optima are evenly distributed over the feasible region. In any practical application we will not know how many global optima the objective function has, nor can we assume that the global optima are evenly distributed.

The question then is: "How should one choose the species distance parameter?"

We suggest that σ_s be selected so that the solutions found will be sufficiently diverse. More precisely, if the SCGA user considers that a solution is significantly novel compared to another one if the distance between them is at least d, then we suggest that σ_s be chosen such that $\sigma_s \geq 2d$. Thus, all species seeds selected will be sufficiently

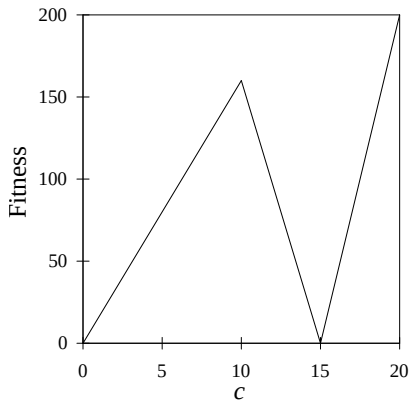

Figure 7: The central two-peak trap function.

different, as defined by the user, and, as a consequence, so will the solutions found by the SCGA.

It is also possible to conceive of mechanisms by which σ_s could be determined automatically and adaptively. For instance, it is easy, at the cost of additional distance evaluations, to vary σ_s iteratively so that, say, a specified proportion of the initial (and subsequent) population(s) are designated as species seeds. Methods for automatically determining suitable values of σ_s are the subject of ongoing research.

4 Experimental Results

4.1 Preamble

The evaluation of any optimization algorithm is, of necessity, an extended process. The true test of our SCGA will, of course, be on real-world design problems for which the number, distribution, and quality of optima are unknown. However, first the algorithm's performance must be investigated on suitable test problems that are well understood. The latter process is made difficult by the fact that there are only limited published results arising from the application of comparable techniques to the sort of problems for which the SCGA has been designed.

When testing the algorithm on well understood problems, there are two measures of performance:

- The consistency with which all known optima are located;
- The average number of objective function evaluations required to find these optima.

In Section 4.2 we give details of the SCGA implementation we used to conduct the performance investigations reported here.

In Section 4.3 we report the results of applying the SCGA to some simple trap functions. This enables us to demonstrate that the SCGA can reliably locate the optima of these deceptive problems, and do so in an average number of evaluations comparable to, and in some cases, significantly better than the number required by other techniques designed for multimodal problems.

Table 2: Comparison of the performance of different algorithms applied to the central two-peak trap function.

Algorithm	Evaluations expected
Stochastic Iterated Genetic Hillclimbing (SIGH)	> 1000000
Sequential Niche GA (SNGA)	3000
SCGA	625

In Section 4.4 we report the results of applying the SCGA to some multimodal problems with multiple global optima from the literature. This enables us to demonstrate that the SCGA can reliably locate all the optima of these problems.

In Section 4.5 we report the results of applying the SCGA to the two-dimensional Shubert function. This is a problem from the literature with 760 local minima, of which 18 are unevenly spaced global optima. We show that the SCGA can reliably locate all these global optima, and investigate the effect on the performance of the algorithm of varying the population size and the species distance. In the absence of any published results concerning the performance of other methods on this problem, we can only draw tentative conclusions about the SCGA's computational efficiency here.

Finally, in Section 4.6 we report the results of applying the SCGA to higher dimension versions of the Shubert function. This enables us to make some observations about how the algorithm's performance scales with problem size.

4.2 SCGA Implementation

The SCGA implementation used in these tests employed:

- proportional, roulette-wheel selection;
- intermediate recombination to perform crossover (with probability p_c), so that the offspring O of randomly chosen parents S and T is:

$$\mathbf{x}_O = \mathbf{x}_T + U \times (\mathbf{x}_S - \mathbf{x}_T) \tag{13}$$

  where U is a uniformly distributed random number over $[0, 1]$;

- uniform neighborhood mutation (with probability p_m):

$$x'_j = x_j + r_m \times R \times \left(x_j^u - x_j^l\right) \tag{14}$$

  where R is a uniformly distributed random number over $[-1, 1]$.

It is important to recognize that our species conservation technique can be employed with any combination of standard selection, crossover, and mutation operators. The operators chosen for this SCGA implementation were chosen specifically because they would not by themselves contribute significantly to the maintenance of diversity:

- proportional, roulette-wheel selection is well known to be susceptible to scaling problems;
- intermediate recombination introduces a bias towards the center of mass of the population;

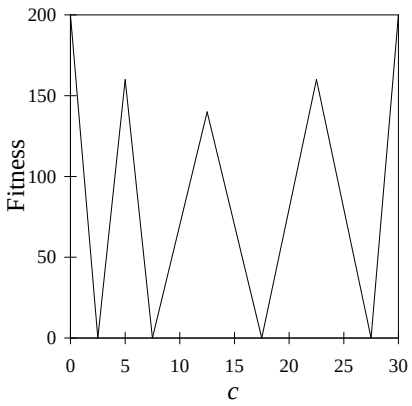

Figure 8: The five-peak uneven trap function.

- uniform neighborhood mutation strictly limits the size of any mutations introduced.

Thus, if the SCGA does succeed in reliably locating all the global optima of multimodal problems, then this success can, with certainty, be attributed to the species conservation technique.

The only control parameters that are important to the performance of the species conservation technique are the population size N and the species distance σ_s. The values of the other control parameters used are given in the following sections in order to aid reproducibility of results but are not significant to our argument.

4.3 Trap Functions

Trap functions are deceptive. That is, they are functions for which finding the global optimum using GAs is hard, because they feature a local optimum or local optima that can "attract" the population away from the true global optimum. Thus, trap functions represent useful first tests for techniques aimed at multimodal function optimization.

Several trap functions have been introduce as benchmarks for evaluating the effectiveness of various kinds of GAs (Ackley, 1987; Deb and Goldberg, 1991; Goldberg, 1992). We tested the SCGA on some of these, as well as on a specially constructed trap function that has multiple global optima. We derived the latter function from "classical" trap functions.

4.3.1 Two-Peak Trap

The fitness function of the two-peak trap is defined by:

$$F(c) = \begin{cases} \frac{160}{15}(15 - c) & \text{for } 0 \leq c < 15 \\ \frac{200}{5}(c - 15) & \text{for } 15 \leq c \leq 20 \end{cases} \tag{15}$$

The global maximum of this function is at $c = 20$ and has a fitness of 200, but there is a "false maximum" at $c = 0$, with a fitness of 160. This function is shown in Figure 6.

In applying the SCGA to this problem, we used the following values for the algorithm control parameters: $N = 50$, $p_c = 0.6$, $p_m = 0.05$, $r_m = 0.15$, $r_f = 0.9999$, and

Table 3: Comparison of the success rates of the SCGA and SGA in finding both the global maxima of the five-peak uneven trap function.

	Optimum at $c = 0$	Optimum at $c = 30$	Both optima
SCGA	100%	100%	100%
SGA	20%	14%	0%

Table 4: SCGA performance on multimodal optimization problems.

Problem	Number of global optima	SCGA parameters used	Evaluations per solution required	
			Mean	σ
Deb's 1st function (Deb, 1989) $f_1(x) = \sin^6(5\pi x)$ where $0 \le x \le 1$	5	$N = 50$, $p_c = 0.6$, $p_m = 0.05$, $\sigma_s = 0.1$, $r_m = 0.1$, $r_f = 0.99$	662	191
Six-hump camel back function (Michalewicz, 1996) $f(\mathbf{x}) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right) x_1^2 + x_1x_2 + \left(-4 + 4x_2^2\right) x_2^2$ where $-3 \le x_1 \le 3$, $-2 \le x_2 \le 2$	2	$N = 50$, $p_c = 0.6$, $p_m = 0.05$, $\sigma_s = 2.0$, $r_m = 0.1$, $r_f = 0.9999$	918	274
Brannin RCOS function (Michalewicz, 1996) $f(\mathbf{x}) = a\left(x_2 - bx_1^2 + cx_1 - d\right)^2 + e(1 - f)\cos(x_1) + e$ where $-5 \le x_1 \le 10$, $0 \le x_2 \le 15$, $a = 1$, $b = 5.1/(4\pi^2)$, $c = 5/\pi$, $d = 6$, $e = 10$, $f = 1/(8\pi)$	3	$N = 100$, $p_c = 0.6$, $p_m = 0.05$, $\sigma_s = 1.0$, $r_m = 0.2$, $r_f = 0.9999$	2843	445

$\sigma_s = 2.0$. We ran the SCGA 30 times and averaged the number of function evaluations needed over these runs. The algorithm found the global maximum on average after 27 generations. The average number of objective function evaluations required was 935 (with a standard deviation of 362). Note that the algorithm produces similar results using any value less than 20 for σ_s.

These results, together with those reported by Ackley (1987) and Beasley et al. (1993), are presented in Table 1. Note that in every SCGA run the global optimum was located. The SCGA comfortably outperforms all but Ackley's *Stochastic Iterated Genetic Hillclimbing* (SIGH) algorithm on this problem. The latter requires on average about 20% fewer function evaluations than the SCGA.

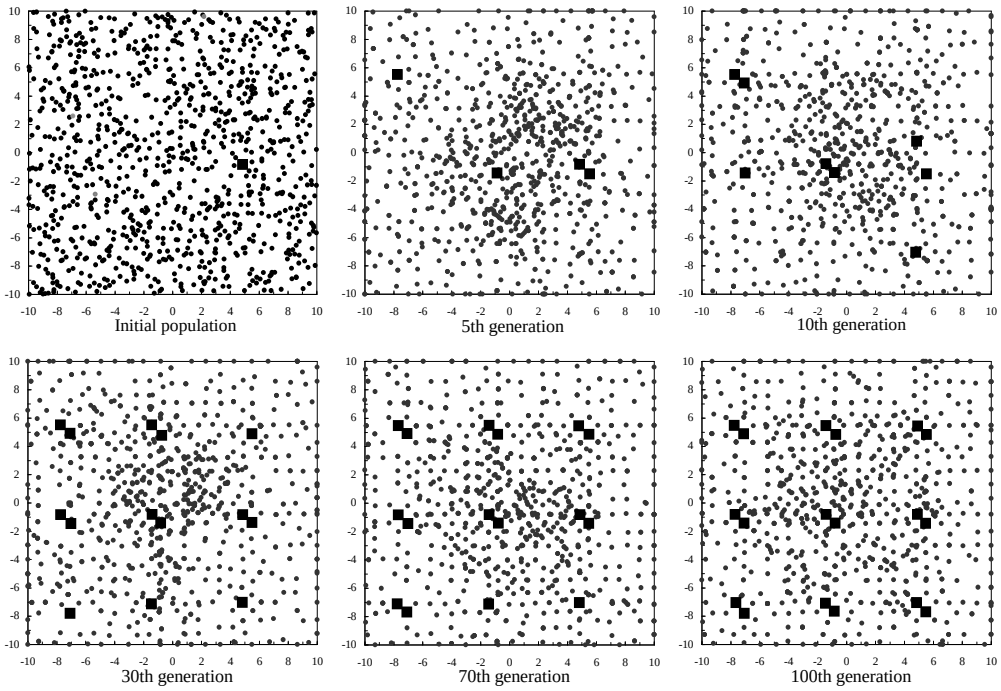

Figure 9: Evolution of an SCGA population when applied to the 2-D Shubert function.

4.3.2 Central Two-Peak Trap

Another trap function for which both Ackley (1987) and Beasley et al. (1993) report results is the Central Two-Peak trap (Figure 7). Its fitness function is given by:

$$F(c) = \begin{cases} \frac{160}{10}c & \text{for } 0 \leq c < 10 \\ \frac{160}{5}(15-c) & \text{for } 10 \leq c < 15 \\ \frac{200}{5}(c-15) & \text{for } 15 \leq c \leq 20 \end{cases} \tag{16}$$

This function has a global maximum of 200 at $c = 20$, but it has a "central" false maximum of 160 for $c = 10$.

With the same parameter settings as for the two-peak trap function, the SCGA found the global maximum of the central two-peak trap function on average after 17 generations. The average number of objective function evaluations required was 625 (with a standard deviation of 144). Note that, although a species distance $\sigma_s = 2.0$ was used, the algorithm is similarly successful using any species distance less than 10, as this value of σ_s allows the SCGA to clearly distinguish between the true and false maxima.

Table 2 compares the performance of the SCGA with results reported by Ackley (1987) and Beasley et al. (1993). Note that the SIGH algorithm, which performed best on the two-peak trap function, exceeded its 1 million evaluation limit on this problem. The superior performance of the SCGA on this problem is readily apparent.

Table 5: Results for the SCGA applied to the 2-D Shubert function.

Population size N	Average number of					
	Generations $\overline{N}_g$		Function evaluations $\overline{N}_f$		Function evaluations per solution $\overline{N}_f^1$	
	Mean	σ	Mean	σ	Mean	σ
90	1170	144	74181	9101	4121	505
100	837	85	59030	6006	3279	333
200	250	18	35647	2663	1980	147
300	167	13	35747	2825	1985	156
400	144	7	41180	2005	2287	111
500	123	10	44178	3575	2454	198
600	111	8	48017	3688	2667	204
700	105	10	52905	5281	2939	293
800	107	7	61840	4258	3435	236
900	86	5	55753	6006	3097	333
1000	89	6	64178	4491	3565	249
1100	90	4	71198	3066	3955	170
1200	86	4	74612	4125	4145	229
1300	93	7	87046	6501	4835	361
1400	77	6	78375	6038	4354	335

4.3.3 Five-Uneven-Peak Trap

While Beasley's *Sequential Niche GA* (SNGA) (Beasley et al., 1993) was reported to be effective for solving a number of trap functions, none of those functions had multiple global optima. Moreover, for both of the trap functions presented above, several SNGA runs were needed to find the single global optimum. For both these trap problems the SCGA was able to find the global optimum in a single run with an average number of function evaluations significantly lower than that required by the SNGA. In this section we present the results from a set of experiments performed to demonstrate that the SCGA can effectively solve deceptive optimization problems with multiple global optima. To do this we introduce a new trap function that we call the *five-uneven-peak function*, defined by:

$$F(c) = \begin{cases} 80(2.5 - c) & \text{for } 0 \le c < 2.5 \\ 64(c - 2.5) & \text{for } 2.5 \le c < 5.0 \\ 64(7.5 - c) & \text{for } 5.0 \le c < 7.5 \\ 28(c - 7.5) & \text{for } 7.5 \le c < 12.5 \\ 28(17.5 - c) & \text{for } 12.5 \le c < 17.5 \\ 32(c - 17.5) & \text{for } 17.5 \le c < 22.5 \\ 32(27.5 - c) & \text{for } 22.5 \le c < 27.5 \\ 80(c - 27.5) & \text{for } 27.5 \le c \le 30 \end{cases} \tag{17}$$

This function has five peaks, as shown in Figure 8. However, it has only two global maxima (with fitnesses of 200) on the borders of the feasible region, at $c = 0$ and $c = 30$, respectively.

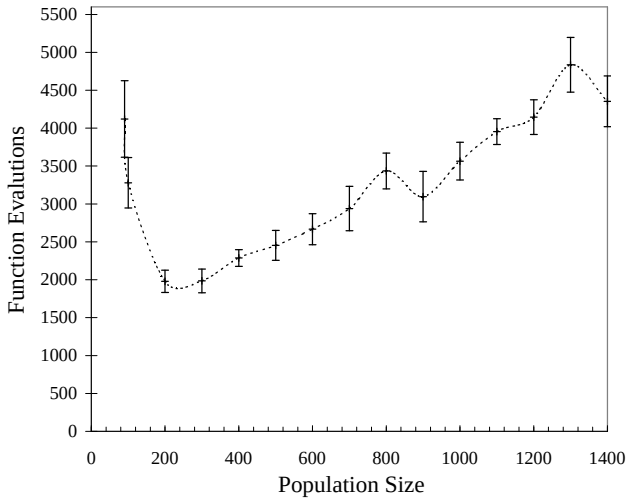

Figure 10: Variation in the average number of function evaluations per solution with population size for the 2-D Shubert function, with one standard deviation (σ) error bars.

With the same parameter settings as before, the SCGA found both the global optima on average after 85 generations. As in the previous cases, the value of the species distance used ($\sigma_s = 2.0$) allowed the SCGA to distinguish between the false and true global maxima. The average number of objective function evaluations required was 2811 (with a standard deviation of 1186), i.e., 1406 evaluations per global optimum (with a standard deviation of 593).

To evaluate the performance of the SCGA on this trap function, we compared it with a SGA using the same operators as the SCGA but with no species conservation. We ran both GAs 100 times. Runs were terminated when both global optima had been found or after 1000 generations. The statistics of the results obtained are summarized in Table 3.

Table 3 clearly shows that the three local maxima of this function did not prevent the SCGA from consistently finding both the global maxima. At the same time, the SGA did not find either of the global maxima in 66% of the runs and only one of the optima in each of the other runs.

4.4 Other Multimodal Problems

Table 4 summarizes the performance of the SCGA (averaged over 30 runs in each case) on three other multimodal optimization problems from the literature. It can be seen that the algorithm reliably locates all the global optima in these problems.

Beasley et al. (1993) report results for the SNGA on the first of these problems (Deb's 1st function). The SNGA found the five global optima in an average of 380 evaluations per optimum. This compares favorably with the SCGA's 662 evaluations per optimum. However, on this problem the SNGA employed a 30-bit binary-coded chromosome rather than a real-valued one, as used in the SCGA, so the spaces being searched by the two algorithms are not identical.

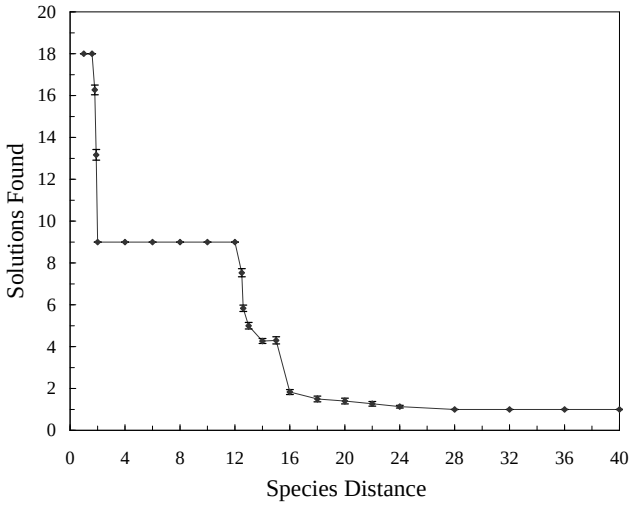

Figure 11: Variation in the average number of optima found with species distance for the 2-D Shubert function, with one standard deviation (σ) error bars.

We have been unable to find any results reported in the literature for the other two problems.

4.5 Two-Dimensional Shubert Function

We now present results we obtained minimizing the two-dimensional (2-D) Shubert function (Michalewicz, 1996) defined by:

$$f(x_1, x_2) = \prod_{i=1}^{2} \sum_{j=1}^{5} j \cos[(j+1)x_i + j] \tag{18}$$

where $-10 \leq x_i \leq 10$ for $i = 1, 2$.

This is a very interesting function. It has 760 local minima, 18 of which are global minima with an objective function value of -186.73. The global optima are unevenly spaced. The multiplicity of global optima and their uneven distribution present the SCGA with a greater challenge than the problems studied in the preceding sections, and are characteristics with which the SCGA must be able to cope if it is to tackle difficult, real-world design optimization problems.

Despite an extensive search, we could find no published reports detailing results for this optimization problem. In a private communication Ken Price (University of Berkeley) reported that he has managed to find all 18 global optima using their highly successful *Differential Evolution* method (Storn and Price, 1997). 50 trials were needed to find these solutions. Each trial required about 6,000 function evaluations, so, in total, about 300,000 function evaluations were needed. Averaging these over the number of global optima found (18) gives an average of 16,667 function evaluations per solution.

The smallest distance between any two global solutions of the 2-D Shubert function is about 0.98, so, to find all the global optima, the species distance σ_s was set to 1.6. Using a population size $N = 1000$, $p_c = 0.6$, $p_m = 0.05$, and $r_m = 0.15$, the SCGA suc-

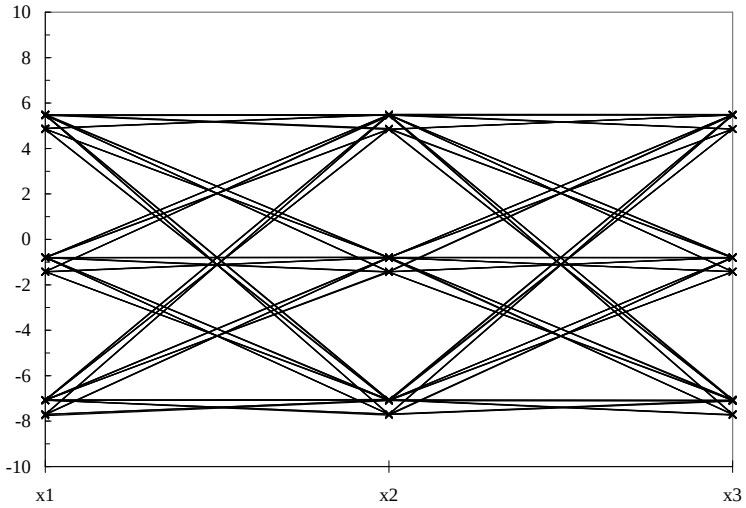

Figure 12: The distribution of the global optima of the 3-D Shubert function. Values of x_1, x_2, and x_3 constituting a global optimum are joined by straight lines.

cessfully identified all the global optima within 100 generations. Figure 9 illustrates the pattern of evolution of the population during a typical SCGA run on this problem. The black squares represent the best individuals found in that generation. Note that here "best individuals" means all the individuals with fitness higher than $(f_{\max} - f_{\min}) \times r_f$, where $r_f = 0.95$, and $f_{\max}$ and $f_{\min}$ are the maximal and minimal fitnesses in the current population.

4.5.1 Effect of Population Size

Next, to analyze the effect of the population size on the performance of the SCGA, the algorithm was run for population sizes (N) ranging from 90 to 1400 individuals. For each population size, the observations were averaged over 30 runs. Table 5 summarizes the results obtained. Note that each run continued until all the 18 global optima were found. The results presented in Table 5 show that, for all the population sizes considered, the average number of function evaluations per solution found ($\overline{N}_f^1$) required by the SCGA is much smaller than the number required by Differential Evolution.

Figure 10 illustrates graphically the variation in $\overline{N}_f^1$ with population size. The number of function evaluations per solution is lowest for population sizes between 200 and 300. This observation is consistent with our discussion in Section 3 concerning the relationship between the species distance σ_s, the number of species seeds, and the population size.

For the given species distance ($\sigma_s = 1.6$), the maximum possible number of species in the domain of the 2-D Shubert function under consideration is about 600. When the population size is smaller than 200, the average number of function evaluations per solution, $\overline{N}_f^1$, rises dramatically. This is because, for small population sizes, the population consists almost entirely of species seeds. This makes it more difficult for the SCGA to generate new individuals that are better than the conserved species seeds,

Table 6: Experimental results for the SCGA applied to the n-D Shubert functions.

Dimension n	Population size N	Average number of generations $\overline{N}_g$		Number of solutions found	Average number of function evaluations $\overline{N}_f$		Average number of function evaluations per solution $\overline{N}_f^1$	
		Mean	σ		Mean	σ	Mean	σ
1	15	33	13	3	324	123	107	41
2	45	447	51	9	14098	1607	1566	179
3	105	1487	136	27	118077	10853	4373	402
4	405	2861	147	81	938707	48473	11589	598

and, although the population is genetically diverse, it evolves slowly as new individuals only survive if they are better than existing species seeds. This represents the most significant danger associated with this form of distributed elitism; rather than premature genetic convergence occurring (the principal danger associated with traditional elitism), a form of genetic stagnation occurs.

For population sizes above 300, the number of function evaluations per solution increases with population size. As the population size increases above its optimal value, more function evaluations are required each generation, independent of the species distance value chosen. Recognizing that the error bars shown are $\pm$ one standard deviation, it can be seen that $\overline{N}_f^1$ increases approximately linearly with N for values of N above 300.

4.5.2 The Effect of the Species Distance Parameter

As mentioned earlier, the choice of the species distance σ_s has a significant effect on the performance of the SCGA. In Section 3.7 we gave an intuitive explanation of why a small value of σ_s would result in the location of many solutions (but possibly at a high cost), while a large value of σ_s would result in the location of too few solutions. To examine this claim, we conducted a series of experiments, again using the 2-D Shubert function, in which we varied the value of σ_s and counted the number of global optima found. For these runs we used $p_c = 0.6$, $p_m = 0.05$, $r_m = 0.15$, $r_f = 0.95$ again, set the population size $N = 300$ (around the optimal value found in Section 4.5.1), and set the number of generations $N_g = 1000$. We averaged the results over 30 SCGA runs for each value of σ_s. The results obtained are shown in Figure 11.

Figure 11 shows that, as expected, as the species distance is increased, the numbers of global optima found decreases. The reason is, of course, that when σ_s is large enough, some global optima will be deemed to belong to another species and therefore be discarded. The SCGA can consistently find all 18 global optima of the 2-D Shubert function in the time allowed in these tests when σ_s is set to 1.6 or below. Above this value the average number of solutions identified drops from 18 to 9 and stabilizes at this level until σ_s reaches 12.5.

As can be seen in the 100th generation population shown in Figure 9, the global optima of the 2-D Shubert function are clustered in pairs. For values of σ_s between 1.96 and 12.5 these pairs of solutions are indistinguishable (deemed to belong to the same

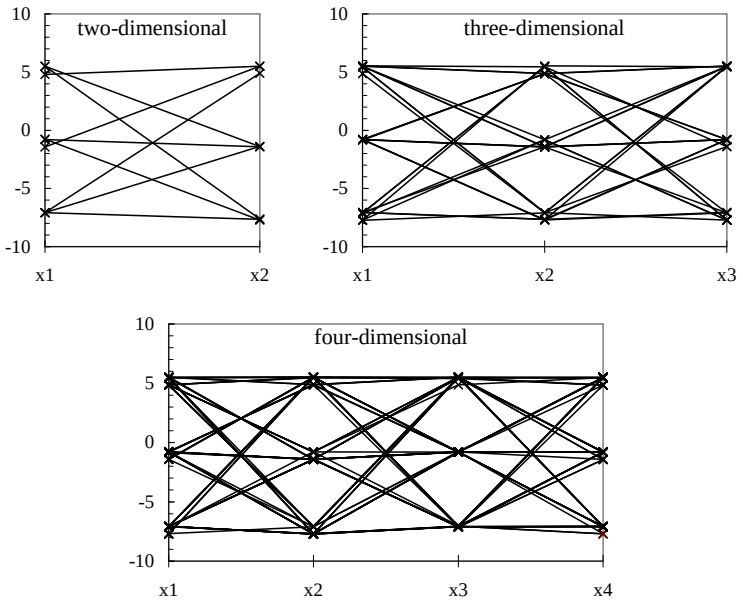

Figure 13: The distributions of solution groups for three Shubert functions. Values of x_1, ... x_n constituting a global optimum in a solution group are joined by straight lines.

species), thus halving the number of global optima that can be identified by the SCGA, even if the algorithm is run indefinitely. From $\sigma_s \geq 12.5$ the average number of global optima identified in these tests decreases further with increasing σ_s, until when $\sigma_s \geq 28$ just one solution is identified.

4.6 The Generalized Shubert Function

To further test the effectiveness of the SCGA, and, in particular, to investigate how its performance scales with problem size, we defined a generalization of the Shubert function for higher dimensions as follows:

$$f(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} \sum_{j=1}^{5} j \cos[(j+1)x_i + j] \tag{19}$$

where $-10 \leq x_i \leq 10$ for $i = 1, 2, ..., n$.

The SCGA was tested on the 3-D Shubert function with $p_c = 0.6$, $p_m = 0.05$, $N = 4000$, $\sigma_s = 1.6$, $r_m = 0.15$, and $r_f = 0.95$. This problem has 81 global optima, which, as in the 2-D Shubert function, are clustered in groups. To distinguish optima within these groups a small value of σ_s must be used. In order to overcome the problem of genetic stagnation identified in the previous section, a large population size is therefore required; hence the choice of $N = 4000$. When tackling real-world problems, such a large population size may not be practical, but, when tackling real-world problems, rather than trying to distinguish between very similar global optima, one is almost certainly going to be trying to identify dissimilar high-quality solutions. In

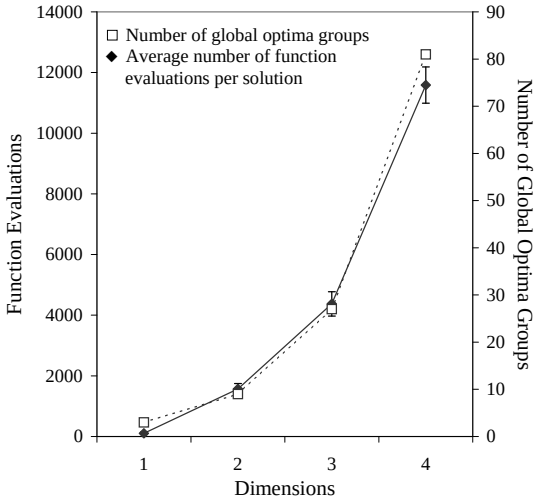

Figure 14: Variation in the number of global optima groups and the average number of function evaluations per solution for n-D Shubert functions.

such a situation a much larger value of σ_s is appropriate, and the population size can be correspondingly smaller.

After 280 generations the SCGA had successfully located the 81 global optima at a computational cost of 10,498 function evaluations per solution. Figure 12 shows the distribution of these global optima of the 3-D Shubert function.

We conjectured that the n-dimensional (n-D) Shubert function has $n \cdot 3^n$ unevenly distributed global optima and that these global optima can be divided into 3^n groups of n, with the members within each group being close to each other. Based on our estimates in Section 3, for the SCGA to find all $n \cdot 3^n$ global optima at minimal computational cost, the population size would need to be very large, and, in consequence, so would the number of function evaluations.

Rather than attempting to find *all* the global optima of the n-D Shubert function, we propose finding the 3^n groups. The SCGA offers a very convenient way of finding these groups. Given the maximum distance d between any two solutions in a group, setting the species distance σ_s to a value greater than $2d$ will result in the SCGA finding at most one member (global optimum) from each group.

We need to note two points about this approach. First, if further solutions need to be found in any of the groups, this can be done by restricting the search to a hypersphere of radius $\sigma_s/2$ centered around the solution representing the group. Second, the approach proposed is reasonable from a practical point of view, as in real-world design problems it is more important to find a few significantly different solutions than many very similar ones.

For the purpose of demonstrating the above approach, let us consider again the 2-D Shubert function. We know that it has 18 global optima that can be divided into 9 pairs. The members of each pair are a distance 0.98 apart, while solutions in different groups are much further apart. In order to locate the 9 groups in this problem, σ_s needs

to be set to a value between 2 and 12, as shown in Figure 11.

Based on these observations and with the parameters set to $p_c = 0.6$, $p_m = 0.05$, $r_m = 0.30$, $r_f = 0.85$, and $\sigma_s = 6.0$, the SCGA was tested (30 times in each instance) on Shubert functions of dimensions 1, 2, 3, and 4. In each case the population size N was set to be 5 times the number of global optima groups. All the global optima groups were located in each run. The results obtained are summarized in Table 6. These show that, at least for the dimensions considered, our conjecture about the number of groups of global optima in n-D Shubert functions can be verified using the SCGA, and demonstrate that the SCGA can reliably locate these groups of high-quality solutions. The distributions of the solution groups found for the 2-, 3-, and 4-dimensional Shubert functions are illustrated in Figure 13.

Figure 14 shows that the average number of function evaluations required by the SCGA to find each solution group scales at approximately the same rate as the number of solution groups for these Shubert functions. Thus, at least on n-D Shubert functions, the computational cost of the SCGA (as measured in function evaluations) is $O(M^2)$, where M is the number of high-quality solutions to be found.

5 Conclusions and Future Work

5.1 Conclusions

In this paper we presented species conservation, a new technique for evolving parallel subpopulations for multimodal function optimization. The technique is based on distributed elitism, which is achieved by selecting from each generation a set of seeds that are considered to be worth preserving into the next generation. The selection of these seeds is based on the notion of dominated species.

The only difference between the GA obtained by introducing species conservation and a classical GA, such as Goldberg's SGA, is the introduction into the generation loop of two processes: the selection of seeds and the conservation of species. It is shown in this paper that the additional overhead associated with these two processes is no higher than that introduced by fitness sharing.

The species conservation technique can be implemented with any combination of standard selection, crossover, and mutation operators. Its use introduces just one additional control parameter, the species distance, and this and the population size are the only parameters important to the performance of the species conservation technique.

A simple SCGA implementation has been tested on standard trap functions from the literature and has proved able to solve these deceptive problems consistently and at computational costs similar to, or substantially lower than, those associated with other multimodal optimization techniques. Because of the paucity of published results from applications of other techniques to the sort of problems we are interested in solving, these results on trap functions are the only concrete performance comparisons we have been able to make. It is worth reiterating that the genetic operators used in our SCGA implementation were chosen specifically because they are "diversity unfriendly." It is highly likely that the SCGA performance reported here could be improved through the use of more "diversity friendly" selection, crossover, and mutation operators.

Tests on other well understood multimodal problems from the literature have enabled us to demonstrate that the SCGA can reliably find all the global optima of these problems. The tests on the Shubert functions have enabled us to demonstrate this ability further, and also to examine features of the SCGA's performance as its control parameters are changed, and as the problem size is changed. Lack of other published results for these problems means our evaluation of the SCGA is necessarily incomplete.

The demonstrated ability of the SCGA to locate all the global optima in these test problems is promising but ultimately of little practical utility. The true test of this technique must be on problems for which the number, distribution, and quality of optima are unknown. This represents a much stiffer challenge because of the difficulty inherent in identifying appropriate values of the algorithm's control parameters, particularly the species distance, in the absence of a priori problem knowledge.

In conclusion, the species conservation technique, incorporated into the SCGA presented in this paper, seems to offer the promise of being an effective and efficient method for inducing niching behavior into GAs with the purpose of finding all the global optima of a multimodal optimization problem or diverse, high-quality solutions of difficult, real-world problems, but there is much work in development and evaluation still to be done.

5.2 Future Work

Our first objective for the future is to develop means of automatically, adaptively identifying suitable values of the key control parameters, the species distance and population size, for problems where there is no a priori knowledge about the distribution of optima available.

Our second task is to apply our technique to hard multimodal engineering design problems with the expectation of discovering novel solutions. We will also perform further empirical studies on the effectiveness and efficiency of the SCGA in solving problems described in the literature, and, in particular, investigate the effects of using more "diversity friendly" genetic operators than those used here.

On the more theoretical side, we need to investigate the effects on SCGA behavior of using different definitions of similarity and of different settings of parameters.

Acknowledgments

We are most grateful to the anonymous referees who reviewed the first version of this paper for their invaluable comments and suggestions of improvements.

References

Ackley, D. H. (1987). An empirical study of bit vector function optimization. In Davis, L., editor, *Genetic Algorithms and Simulated Annealing*, pages 170–204, Pitman, London, UK.

Beasley, D., Bull, D. R., and Martin, R. R. (1993). A Sequential Niche Technique for Multimodal Function Optimization. *Evolutionary Computation*, 1(2):101–125.

Cavicchio, D. J. (1970). *Adaptive Search Using Simulated Evolution*. Ph.D. thesis, University of Michigan, Ann Arbor, Michigan.

Cohoon, J. P. et al. (1987). Punctuated equilibria: a parallel genetic algorithm. In Grefenstette, J. J., editor, *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 148–154, Lawrence Earlbaum, Hillsdale, New Jersey.

Cohoon, J. P. et al. (1991). Distributed Genetic Algorithms for the Floorplan Design Problem. *IEEE Transactions on Computer-Aided Design*, 10(4):483–492.

Davidor, Y. (1991). A naturally occurring niche and species phenomenon: the model and first results. In Belew, R. K. and Booker, L. B., editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 257–263, Morgan Kaufmann, San Mateo, California.

Deb, K. (1989). *Genetic Algorithms in Multimodal Function Optimization*. Master's thesis, University of Alabama, Tuscaloosa, Alabama.

Deb, K. and Goldberg, D. E. (1989). An investigation of niche and species formation in genetic function optimization. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, Morgan Kaufmann, San Mateo, California.

Deb, K. and Goldberg, D. E. (1991). Analyzing deception in trap functions. IlliGAL Technical Report 91009, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana, Illinois.

De Jong, K. A. (1975). *An Analysis of Behavior of a Class of Genetic Adaptive Systems*. Ph.D. thesis, University of Michigan, Ann Arbor, Michigan.

Eby, D. et al. (1999). The optimization of flywheels using an injection island genetic algorithm. In Bentley, P. J., editor, *Evolutionary Design by Computers*, pages 167–190, Morgan Kaufmann, San Francisco, California.

Eshelman, L. J. (1991). The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination. In Rawlins, G. J. E., editor, *Foundations of Genetic Algorithms*, pages 265–283, Morgan Kaufmann, San Mateo, California.

Gen, M. and Cheng, R. (1997). *Genetic Algorithms and Engineering Design*. John Wiley and Sons, New York, New York.

Glover, F. (1989). Tabu Search – Part I. *ORSA Journal on Computing*, 1(3):190–206.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, Massachusetts.

Goldberg, D. E. (1990). A Note on Boltzmann Tournament Selection for Genetic Algorithms and Population-oriented Simulated Annealing. *Complex Systems*, 4(4):445–460.

Goldberg, D. E. (1992). Construction of High-order Deceptive Functions using Low-order Walsh Coefficients. *Annals of Mathematics and Artificial Intelligence*, 5:35–48.

Goldberg, D. E. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In Grefenstette, J. J., editor, *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49, Lawrence Earlbaum, Hillsdale, New Jersey.

Gordon, V. S., Whitley, D., and Böhn, A. (1992). Dataflow parallelism in genetic algorithms. In Männer, R. and Manderick, B., editors, *Parallel Problem Solving from Nature 2*, pages 533–542, Elsevier Science, Amsterdam, The Netherlands.

Holland, J. H. (1975). *Adaptation in Natural and Artificial System*. University of Michigan Press, Ann Arbor, Michigan.

Hughes, E. J. and Leyland, M. (2000). Using Multiple Genetic Algorithms to Generate Radar Point-scatterer Models. *IEEE Transactions on Evolutionary Computation*, 4(2):147–163.

Lienig, J. and Thulasiraman, K. (1993). A Genetic Algorithm for Channel Routing in VLSI Circuits. *Evolutionary Computation*, 1(4):293–311.

Mahfoud, S. W. (1995). Niching methods for genetic algorithms. IlliGAL Technical Report 95001, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana, Illinois.

Martin, W. N., Lienig, J., and Cohoon, J. P. (1997). Island (migration) models: evolutionary algorithms based on punctuated equilibria. In Bäck, T., Fogel, D. B., and Michalewicz, Z., editors, *Handbook of Evolutionary Computation*, pages C6.3:1–C6.3:16, Institute of Physics Publishing, Bristol, UK.

Mengshoel, O. J. and Goldberg, D. E. (1999). Probability crowding: deterministic crowding with probabilistic replacement. In Banzhaf, W., Daida, J., and Eiben, A. E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference 1999*, pages 409–416, Morgan Kaufmann, San Francisco, California.

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, New York, New York.

Parmee, I. C. (1999). A Review of Evolutionary/Adaptive Search in Engineering Design. *Evolutionary Optimization*, 1(1):13–39.

Sarma, J. and De Jong, K. (1997). Generation gap methods. In Bäck, T., Fogel, D. B., and Michalewicz, Z., editors, *Handbook of Evolutionary Computation*, pages C2.7:1–C2.7:5, Institute of Physics Publishing, Bristol, UK.

Spears, W. M. (1994). Simple subpopulation schemes. In Sebald, A. V. and Fogel, L. J., editors, *Proceedings of the Third Annual Conference on Evolutionary Programming*, pages 296–307, World Scientific, Singapore.

Storn, R. and Price, K. (1997). Differential Evolution — a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11(4):341–359.

Whitley, D. (1989). The GENITOR algorithm and selection pressure: why rank-based allocation of reproductive trials is best. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 116–121, Morgan Kaufmann, San Mateo, California.