Multimodal Function Optimization Using Minimal Representation Size Clustering and Its Application to Planning Multipaths

Cem Hocaoğlu
Electronics Agile Manufacturing Research Institute
Electrical, Computer, and Systems Engineering Department
Rensselaer Polytechnic Institute
Troy, New York 12180-3590
hocaoglu@eamri.rpi.edu

Arthur C. Sanderson
Electronics Agile Manufacturing Research Institute
Electrical, Computer, and Systems Engineering Department
Rensselaer Polytechnic Institute
Troy, New York 12180-3590
acs@cat.rpi.edu

Abstract

A novel genetic algorithm (GA) using minimal representation size cluster (MRSC) analysis is designed and implemented for solving multimodal function optimization problems. The problem of multimodal function optimization is framed within a hypothesize-and-test paradigm using minimal representation size (minimal complexity) for species formation and a GA. A multiple-population GA is developed to identify different species. The number of populations, thus the number of different species, is determined by the minimal representation size criterion. Therefore, the proposed algorithm reveals the unknown structure of the multimodal function when a priori knowledge about the function is unknown. The effectiveness of the algorithm is demonstrated on a number of multimodal test functions. The proposed scheme results in a highly parallel algorithm for finding multiple local minima. In this paper, a path-planning algorithm is also developed based on the MRSC_GA algorithm. The algorithm utilizes MRSC_GA for planning paths for mobile robots, piano-mover problems, and N-link manipulators. The MRSC_GA is used for generating multipaths to provide alternative solutions to the path-planning problem. The generation of alternative solutions is especially important for planning paths in dynamic environments. A novel iterative multiresolution path representation is used as a basis for the GA coding. The effectiveness of the algorithm is demonstrated on a number of two-dimensional path-planning problems.

Keywords

Genetic algorithms, multimodal function optimization, minimal representation size, minimum description length, cluster analysis, species formation, path planning, multipaths.

1. Introduction

One of the well-known problems in genetic search is the phenomenon called *genetic drift* (De Jong, 1975). In multimodal functions with equal peaks, simple genetic algorithms (GAs) converge to only one of the peaks, and that peak is chosen randomly due to the stochastic variations associated with the genetic operators. We are interested in overcoming this problem. In addition, in multimodal functions of equal or unequal peaks, we may be interested in obtaining several best solutions rather than only the best one.

In path planning, it may be difficult to combine all the important factors into a single evaluation function. One way to approach this problem is to use a few important criteria in constructing the evaluation function and generating alternative solutions to the problem using evolutionary computation. From these selected paths, one can then choose the best one based on further analysis of the problem. For example, in dynamic environments one or more of these plans may become infeasible, so one of the feasible alternatives must then be chosen (Hocaoğlu & Sanderson, 1996).

Previously, several methods have been proposed to solve the problem of locating all the peaks of a multimodal function. In Section 2, we describe some of these methods and give their limitations.

In this paper, we present a new technique based on the concepts of minimal representation size criterion (Segen & Sanderson, 1981) and cluster analysis (Fukanaga, 1972; Hartigan 1975). In particular, we enforce species formation through minimal representation size clustering. We make use of multiple populations to track each species. Species evolve separately with occasional crossover-like interactions across populations. The proposed algorithm is highly parallel. Multiple populations on each processor may evolve separately while the minimal representation size clustering ensures that no two populations converge on the same optimum.

In this paper, we also present a new path planner that utilizes the minimal representation size cluster GA (MRSC_GA) (Hocaoğlu & Sanderson, 1995) to track various species for generating multiple paths.

Section 3 describes multipopulation GAs. In Section 4, we provide a brief description of cluster analysis techniques. Section 5 presents the minimal representation criterion principles. In Section 6, we formulate the problem of species formation based on the MRSC analysis. Section 7 describes the MRSC_GA for multimodal function optimization. In Section 8, we test the proposed algorithm on a number of popular multimodal functions. Section 9 provides background information on approaches for solving path-planning problems and describes the importance of developing a GA-based path planner. In Section 10, we formulate the path-planning problem using MRSC_GA analysis. Section 11 gives the experimental results for various path-planning problems. In Section 12, future extensions to the MRSC_GA-based path-planning algorithm are discussed.

2. Previous Methods in Species Formation

A number of methods have been described in the literature for inducing niche and species in GAs. Cavicchio's *preselection* scheme (Cavicchio, 1970), where an offspring replaces one of its parents if the offspring's fitness is higher than that of its parent is one of the first attempts to induce niche and species in GAs.

Crowding and sharing are two other methods that are used to induce species in GA search. We briefly discuss these methods.

2.1 Crowding Method

Genetic algorithms that attempt to generate stable niches by replacing population elements with similar individuals are referred to as *crowding* methods. In his crowding scheme, DeJong (1975) extended the preselection scheme of Cavicchio (1970). Only some proportion of the population is allowed to reproduce each generation, and the new offspring replaces the most similar chromosome in a pool of sampled individuals from the population. In this method, the

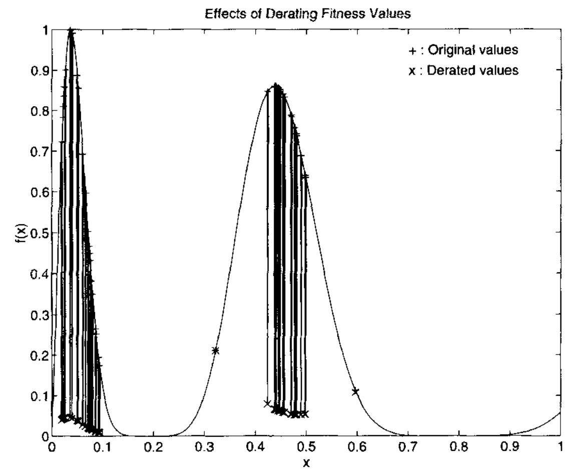

Figure 1. Effects of derating fitness-values.

niche and species formation is limited by the stochastic replacement errors (Mahfoud, 1992). However, crowding methods still remain promising in maintaining stable subpopulations.

2.2 Sharing Method

In sharing methods (Goldberg & Richardson, 1987; Deb & Goldberg, 1989), the individuals in a population are grouped together according to the similarity of the individuals in the decoded parameter space or the gene space. Individuals within the same *niche radius* must share their fitnesses with individuals belonging to the same niche according to a sharing function. The sharing function is defined by a sharing parameter σ_{share}, for controlling the extent of sharing, and a power law function $\text{Sh}(d)$ having a distance metric d between two individuals as a parameter:

$$\text{Sh}(d) = \begin{cases} 1 - \left(\dfrac{d}{\sigma_{\text{share}}}\right)^{\alpha}, & \text{if } d < \sigma_{\text{share}} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

The parameter α determines the degree of convexity of the sharing function. The sharing takes place by derating an individual's fitness by its niche count. The niche count for an individual is taken to be the sum of all sharing function values over the entire population. The method of sharing is sensitive to the sharing parameter σ_{share}. This parameter is dependent on the number of peaks of the multimodal function and the lower and upper bounds of the solution space (Deb & Goldberg, 1989). In Deb & Goldberg (1989), a simple method for computing the niche radius is given. However, that method requires the number of peaks in the function to be known a priori. In addition, it assumes that all the peaks are equally spread throughout the search space.

Figure 1 shows that a correct choice of the niche radius is more difficult to obtain when the peaks are unequally spread throughout the search space. For example, derating the fitnesses using a niche radius of 0.1 has generated two *erroneous* species in the function structure (individuals at x values of 0.3 and 0.6).

Beasley et al. (1993) proposed a sequential niche technique for multimodal function optimization. Their algorithm is based on iterating a traditional GA. At each iteration, a fitness derating function is applied to suppress the fitness values in the regions where solutions have already been found. In this way, they may locate different local maxima. But this method is also sensitive to the value of niche radius. Problems will arise if an inappropriate (small or large) niche radius is used. Beasley et al. (1993) demonstrated that with a small niche radius, artificial peaks are introduced, whereas with a large niche radius, the maxima of interest may be lost, or an incorrect solution may be obtained. They proposed solutions to the niche radius problems. Beasley et al. (1993) assumed that there are a known number of maxima of *interest*, all having fitness values greater than the solution threshold. In addition, they made the assumption that the peaks are evenly distributed throughout the search space.

Yin and Noël (1993) introduced a sharing scheme using cluster analysis that makes use of clustering algorithms to cluster the individuals into different niches. The niches are then used for derating the fitness values. The use of cluster analysis eliminates the requirement of knowing the number of peaks. Although the proposed algorithm is more efficient than the usual sharing scheme, two additional distances are introduced in the algorithm. The algorithm is very sensitive to these distances. If they are too small, some peaks carry a large number of clusters; thus, the number of niches is very large. If the distances are too large, the number of clusters becomes too small, so some fine peaks are lost. The use of the work described by Yin and Noël (1993) remains substantially different from that introduced in this paper.

3. Multipopulation GAs

Parallel GAs (PGAs) with multiple populations have been the subject of study for a number of years (Mühlenbein, Schomisch, & Born, 1991; Cohoon, Martin, & Richards, 1991; Fan, 1993; Levine, 1994; Chen, Nakao, & Fang, 1996). They are usually designed for parallel/distributed computers with multiple interacting populations. These are called *island* model GAs, where a GA population is divided into several subpopulations, and each subpopulation evolves independently in a sequential GA cycle. Occasionally, highly fit individuals migrate among subpopulations. The migration of individuals among subpopulations allows sharing of highly fit individuals and helps to maintain genetic diversity. Individuals that are fit for migration are allocated additional reproductive trials that increase the overall selective pressure (Levine, 1994; Whitley, 1993). In Mühlenbein et al. (1991), a PGA is applied to continuous function optimization. They use a mixed strategy. Subpopulations try to locate good local minima. If a subpopulation does not progress after a number of generations, hill climbing is used. Local minima found by subpopulations are migrated to neighboring subpopulations. Chen et al. (1996) applied the island model PGA to image restoration problems.

4. Cluster Analysis

Cluster analysis (Hartigan, 1975) refers to the unsupervised classification of objects into subsets that have meaning in the context of a particular problem. The unsupervised classification means that there is no a priori knowledge about the data samples. The organization of the objects into an efficient representation characterizes the sampled population. The fundamental issue in clustering problems is the definition of a cluster itself (Fukanaga, 1972). The definition of a cluster provides the means of distinguishing between good and bad classification of samples. In this paper, the *minimal representation size criterion* described in the next section and in Segen and Sanderson (1981) and Rissanen (1978) will be used as the basis for identifying *good* clusters.

In this paper, we make use of a KMEANS clustering algorithm. The KMEANS algorithm (Hartigan, 1975) is an iterative algorithm that is based on minimizing the ratio of within-class scatter to the mixture scatter. In general, the KMEANS algorithm requires a priori knowledge of the number of clusters. In this paper, we describe a modified KMEANS clustering algorithm for clustering the data and determining the optimal number of clusters based on the minimal representation size criterion (Segen & Sanderson, 1981). The resulting clusters will correspond to different species and the resulting number of clusters, ideally, to the number of local minima in the multimodal function.

5. Minimal Representation

The minimal representation criterion was introduced as a criterion for model inference by Segen and Sanderson (1981). The ideas of Solomonoff (1964), Kolmogorov (1968), Akaike (1974), Chaitin (1975), Rissanen (1978), and Wallace and Boulton (1968) resulted in the development of algorithmic information theory to which the method of minimal representation size is related. The approach was demonstrated on problems in attributed image matching (Sanderson & Foster, 1989), clustering and polynomial fitting (Segen & Sanderson, 1981). Iba, de Garis, and Sato (1994) introduced a minimum description in genetic programming. They applied the minimum description length (MDL)-based fitness functions to problems in pattern recognition.

The minimal representation criterion is based on the principle of minimum complexity of a program that would reproduce the observed data on the output tape when executed on a deterministic Universal Turing Machine (UTM). The length of such a program in bits is referred to as the *representation size* of the observed data. The program constitutes a model, data residuals, and a procedure. The complexity of the representation is measured in terms of the size of the overall program. In general, there is a trade-off between the model size and data residual size. The data D can originate from one of several available models.

Assume that there are N individuals, $X_1, \ldots, X_N$ that we want to classify or cluster. Denote these individuals by

$$X^N = X_1, \ldots, X_N$$

where each X_j can be an n-dimensional individual in parametric space.

Let Q be a set of models, and let $q \in Q$ be some model describing the individuals X^N. It has been shown (Segen & Sanderson, 1981) that for statistical models, the representation size for the model q with respect to the data X^N is

$$S[q, X^N] = S[q] - \log_2 P_q(X^N) \tag{2}$$

where $P_q(X^N) = P_q(X_1, \ldots, X_N)$, and $S[q, X^N]$ denotes the representation size of the data X^N when described by the model q. The size of the procedure that tells the UTM how to interpret the program is represented by a fixed number of bits, so it is not included in the above formula. The size of the model q is denoted as $S[q]$; $P_q(X^N)$ is the joint density function under the model q evaluated for the individuals X^N; and $-\log_2 P_q(X^N)$ denotes the number of bits required to encode the data residuals.

According to the minimal representation size criterion, the best description of the data is given by the smallest size program. This corresponds to the model with the minimal representation size:

$$q^* = \arg\min_{q \in Q} S[q, X^N]$$

Such a q^* always exists (Segen & Sanderson, 1981). In Section 6, we formulate a representation size expression for the formation of species in GAs.

6. Formulation of Speciation Using MRSC Analysis

We support speciation in GAs by classifying the individuals into different clusters. Each cluster will correspond to a different species.

Let N individuals X^N be classified into L clusters $c_1, \ldots, c_L$. The number of clusters L is to be determined. We pose the clustering problem as a statistical model identification problem, where within-class probability distributions and the distribution of each class are known in a general form. In this paper, the *mixture probability* density $p(X)$ is assumed to be the statistical model. We specify $p(X)$ in terms of the conditional probability density function of class c_i, $p(X|c_i)$, and the a priori probability of class c_i, denoted by $P(c_i)$. The model q is represented by L, $p(X|c_i)$, and $P(c_i)$ for $i = 1, \ldots, L$.

Under the assumed model, we have

$$p(X) = \sum_{i=1}^{L} p(X|c_i)P(c_i) \tag{3}$$

Assume that the class density function for each class c_i is normally distributed with mean M_i and covariance Σ_i. That is,

$$p(X|c_i) = N(M_i, \Sigma_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(X - M_i)^T \Sigma_i^{-1}(X - M_i)\right) \tag{4}$$

where d is the problem dimension.

In this case, the model q is represented by L, M_i, Σ_i, and $P(c_i)$ for $i = 1, \ldots, L$. Denote the class to which X_j is assigned by $c_{k_j}, j = 1, \ldots N$, $k_j \in \{1, \ldots, L\}$. Let $\Omega_q(X)$ be a decision function that assigns X to one of the classes c_i under the model q. Then,

$$\Omega_q(X_j) = c_{k_j}, \qquad k_j \in \{1, \ldots, L\}$$

Assume that X can only be assigned to one class. That is,

$$p(X|c_i) = 0 \qquad \text{if} \quad c_i \neq \Omega_q(X) \tag{5}$$

Assuming that the observations X_j are independent, $P_q(X_j) = p(X_j)$, and making use of Equations 3 and 5, we get

$$P_q(X^N) = P_q(X_1, \ldots, X_N) = \prod_{j=1}^{N} P_q(X_j) = \prod_{j=1}^{N} p(X_j|c_{k_j})P(c_{k_j}) \tag{6}$$

We can now write the expression for the representation size $S[q, X^N]$. The representation size for the model q is given by

$$S[q] = S[L] + \sum_{i=1}^{L} (S[M_i] + S[\Sigma_i] + S[P(c_i)]) \tag{7}$$

From Equations 2 and 6, we get

$$S[q, X^N] = S[q] - \sum_{j=1}^{N} [\log_2 p(X_j|c_{k_j}) + \log_2 P(c_{k_j})] \tag{8}$$

After some algebraic manipulations and removal of constant terms, we can write

$$S[q, X^N] = S[q] + \frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{N_i}(X_j^{(i)} - M_i)^T \Sigma_i^{-1}(X_j^{(i)} - M_i) + \sum_{i=1}^{L} N_i\left[\frac{1}{2}\log_2|\Sigma_i| - \log_2 P(c_i)\right] \tag{9}$$

where $X^{(i)}$ represents the individuals assigned to class c_i, and N_i is the number of individuals in cluster c_i.

The representation size for an object n can be defined as in Segen and Sanderson (1981). In this work, it is defined by

$$S[n] = \lceil \log_2(|n| + 1) \rceil$$

To select the best model q^* and the best classification $\Omega^*_{q_*}$ for this model, the representation size $S[q, X^N]$ is minimized over all possible classifications Ω_q and over all possible models q. The one having the global minimum provides us with the optimal clustering. In practice, we hypothesize a partial model q, the number of clusters L, and the a priori class probabilities $P(c_i)$. Then, the KMEANS clustering algorithm is used to find the best classification Ω^*_q. It is assumed that the KMEANS algorithm gives the minimal representation size for the hypothesized partial model q. To establish a closer connection with GAs, we note that the data in cluster analysis corresponds to the individuals in GAs and the clusters are analogous to the niche and species in GAs. Therefore, the species formation in GAs is achieved through the minimal representation size clustering. Through experimentation, we have verified that the Gaussian assumption in the formulation of species formation, in general, gives good results even when the clusters are not normally distributed.

7. High-Level MRSC_GA

In this section, we describe the MRSC_GA. The MRSC_GA incorporates a simple GA. The simple GA implemented uses a binary string representation to encode the problem parameters. One generation of the simple GA involves selection, mutation, crossover, elitism, and evaluation procedures. A proportional selection algorithm based on stochastic universal sampling (Baker, 1987) or a linear ranking selection algorithm is used to create the next generation. The mutation operator is applied to each individual in the new population. Each position in the binary representation of an individual has a certain probability of undergoing mutation. A two-point crossover operator exchanges information among pairs of individuals in a pool of individuals from the population. The elitist strategy guarantees that the best individual of the last population is copied to the current population. The evaluation procedure assigns fitnesses to each individual in the population.

The high-level MRSC_GA is as follows:

1. Form uniformly sampled initial population(s) having *PopSize* individuals.
2. Set the current number of populations: *CurNumPop* = *Initial_No_Pops*. In general, *Initial_No_Pops* = 1.
3. Run the simple GA with elitist strategy until the generation counter has value

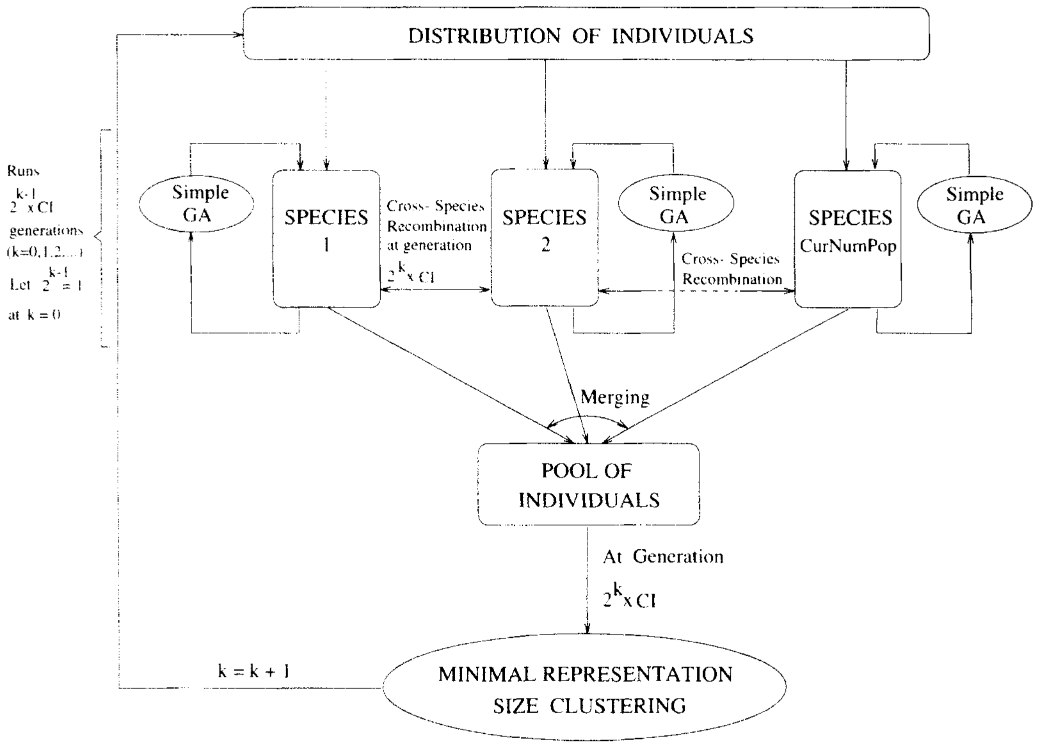

Figure 2. Architecture of MRSC_GA. CI ≡ Cluster_Interval.

Cluster-Interval. At this point, perform a *Cross-Species Interaction* among individuals from different species (populations) (see Fig. 2).

4. If the maximum number of clustering, *Max-Cluster-Count*, is performed, go to step 10.
5. Merge the individuals of each population (see Fig. 2).
6. Cluster individuals into optimal number *Lopt* populations using minimal representation size clustering (see Fig. 2).
7. Set *CurNumPop* = *Lopt*.
8. Update cluster interval: *Cluster-Interval* = K**Cluster-Interval*. We use K = 2.
9. Distribute chromosomes into *CurNumPop* populations (see Fig. 2). For populations having more than *PopSize* individuals, discard the worst chromosomes from the population. For populations having less than *PopSize* individuals, generate individuals that are perturbed around the cluster means. The maximum perturbation along each dimension is equal to the standard deviation of the corresponding variable.
10. If the termination condition is not reached, go back to step 3. The algorithm terminates when the maximum number of function evaluations is performed or when no function evaluations are performed for *NoEval* number of generations.

Figure 2 illustrates the key components of the MRSC_GA architecture. It also demonstrates the evolutionary behavior of the MRSC_GA. As shown in Figure 2, the MRSC_GA allows one to run multiple populations of chromosomes that evolve separately, with occasional cross-species interactions among populations. The simple GA runs sequentially, processing each species until the next cluster interval is reached. Cross-species interaction is an operator similar to the crossover operator in a single population. In cross-species interaction, the individuals are selected from different species, and then crossover operator is applied to produce two new offspring. The cross-species interaction is controlled by *Cross-Species-Rate*. The cross-species interaction is performed for all pairs of populations. *Cross-Species-Rate* is usually very low so as not to destroy the resulting speciation. It may prove to be useful when the function structure has plateaus on which *pseudo-species* may be formed. A *Cross-Species-Rate* of zero means that no cross-species interactions ever happen.

MRSC analysis is performed at each *Cluster-Interval* (see Fig. 2). After clustering, no elitist strategy is used for one generation for each population. Cluster analysis is a basis for forming species. Once speciation is achieved, clustering does not contribute much. Since it increases the computational cost of the algorithm, it is performed less frequently as the algorithm converges. The individuals are then distributed into their respective locations according to the minimal representation size classification algorithm.

The algorithm developed herein is a powerful one. It can incorporate several variations of this algorithm:

- single-population GA:
  - This is the same as the simple GA, which the MRSC_GA incorporates. A brief description of this algorithm was provided at the beginning of this section. This version is only capable of finding a single near-optimal solution.
- multiple-population GA:
  - This allows one to run multiple populations of individuals that evolve separately in parallel. There is no interaction among populations. This is equivalent to iterating a single population many times. There is no explicit scheme to force the populations to converge to different local minima. So, different populations may converge to the same local minimum.
- multiple-population GA with cross-species interaction:
  - In this scheme, multiple populations evolve in parallel. Populations interact with the *CrossSpecies* operator. There is no guarantee that populations will converge to different local minima.
- multiple-population GA with MRSC and cross-species interaction (MRSC_GA):
  - The algorithm described in this paper.

8. Simulation Results

The MRSC_GA presented in this paper was tested on a variety of multimodal functions with different degrees of complexity. In this paper, we report the results of experiments on the following multimodal functions:

- F1: Modified Himmelblau's function (Deb, 1989).

Table 1. GA parameters used for each experiment.

	F1	F2	F3
Initial No Pops	1	1	1
Max No Pops	8	8	8
Population Size	70	30	30
Crossover Rate	0.6	0.8	0.6
CrossSpecies Rate	0.0	0.0	0.0
Mutation Rate	0.005	0.0005	0.0005
Cluster Interval	1	1	1
Max Cluster Count	10	10	10

- F2: Decaying sinusoids (Goldberg & Richardson, 1987; Deb & Goldberg, 1989).
- F3: Valleys function.

The GA is implemented in C and C++ programming languages on a SPARC-10 workstation. In Table 1, we give the set of parameters used to run the experiments. For these experiments, we have run the algorithm for an extended number of generations to demonstrate the stability of subpopulations.

8.1 F1: Himmelblau's Function

This is a function of two variables that have been modified (Deb, 1989) to be a maximization problem. The modified Himmelblau's function is expressed by

$$F1(x,y) = 200 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2 \tag{10}$$

Figure 3 shows a shaded surface plot of the modified Himmelblau's function. This function has four maxima of equal height having value 200. The maxima are approximately at locations $(3.584, -1.848)$, $(3.0, 2.0)$, $(-2.805, 3.131)$, and $(-3.779, -3.283)$.

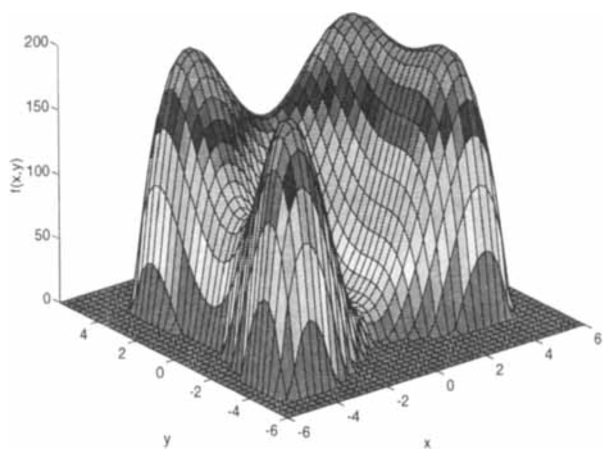

Figure 3. Modified Himmelblau's function.

Table 2. Solution points given by the MRSC_GA for the modified Himmelblau's function.

	(x,y)	$f(x,y)$
1	$(3.579, -1.820)$	199.9882
2	$(2.999, 1.996)$	199.9995
3	$(-2.805, 3.133)$	199.9998
4	$(-3.749, -3.274)$	199.9508

Table 1 (column F1) shows the parameters with which the GA was run. For this experiment, the species formation takes place at generations 2, 4, 8, 16, 32, 64, and 128. In Figure 4, we give the results of species formation at generations 2, 16, and 128. The left column of Figure 4 gives the classification of the individuals into different species based on the minimal representation size. The distribution of individuals is shown on a contour plot of the search space. The plots of representation size versus the number of species are shown in the right column of the same figure. The number of species corresponding to the minimal representation size determines the degree of speciation in the solution. In this case, the number of species increased from 1 at generation 2, to 2 at generation 16, to 4 at generation 128. The graphs show that the algorithm is able to correctly classify any species formation as the GA runs. For example, the figure at the bottom left clearly shows the existence of four different species. The figure on the bottom right verifies the existence of four species by having a minimal representation size at a value of 4 for the number of species (populations).

Table 2 gives the solution points and their corresponding values obtained by the MRSC_GA run for the modified Himmelblau's function. The worst-case error is 0.0246%. Figure 5(a) shows the best individual of each species at the final generation. It is clear from this figure that MRSC_GA is able to locate all maxima for this function. In Figure 5(b), we plot the current best performance of each population at each generation. It is obvious from the graph that all four populations converge very close to the optimal value. Besides showing the performance of the GA, this graph provides a history of species formation in GAs. For example, the graph shows that after generation 32, two new species have evolved.

8.2 F2: Decaying Sinusoids

This function has five unequal peaks in the interval $0 \leq x \leq 1$. It is defined by

$$F2(x) = \sin(5.1\pi x + 0.5) * \exp\left(-4\log(2)\left(\frac{x - 0.0667}{0.8}\right)^2\right) \tag{11}$$

The maxima are represented by $(x,f(x))$ and have approximate values of (0.06683, 1.0000), (0.26181, 0.8472), (0.45679, 0.5153), (0.65178, 0.2251), and (0.84676, 0.0706).

Table 1 (column F2) shows the parameters with which the GA was run for this function. Table 3 gives the solution points and their corresponding values obtained by the MRSC_GA run for the decaying sinusoids. The results are very accurate. Figure 6(a) shows the best individual of each species at the last generation. It is clear from this figure that the MRSC_GA is able to locate all maxima for this function as well. In Figure 6(b), we plot the current best performance of each population at each generation. The graph shows that all four populations converge very close to the corresponding optimal values. It is interesting to note that at generation 2, a new species has evolved, and shortly after, at generation 16, it went into extinction. A similar species has been formed at generation 32. In Figure 6(b), we

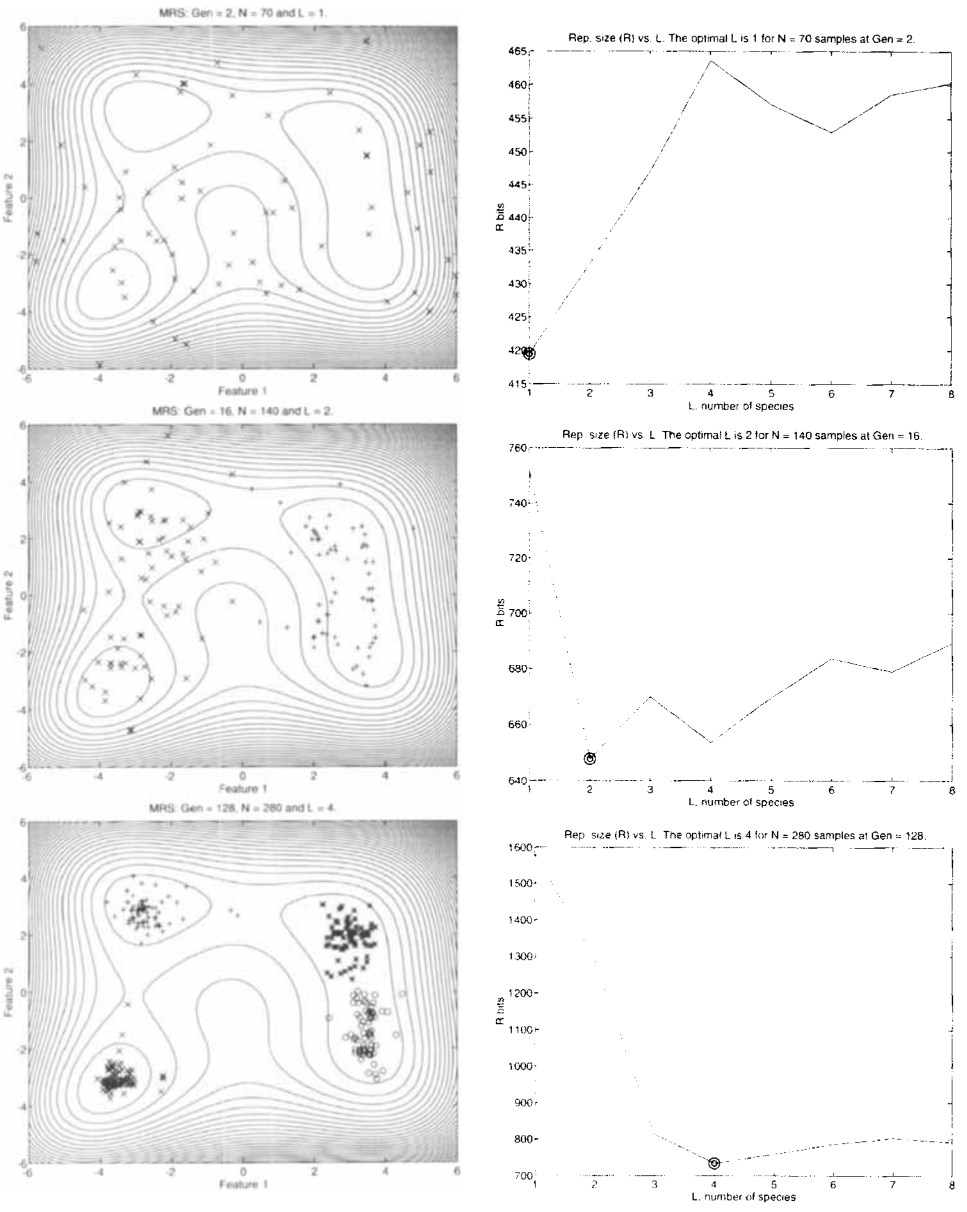

Figure 4. Plots of species formation (left column) and the corresponding representation (Rep.) size (right column) as the GA runs for the modified Himmelblau's function. The minimal representation size (MRS) selects the number of species (L) found in the solution. In this case, the number of species increases from 1 at generation (Gen) 2, to 2 at generation 16, and to 4 at generation 128.

also give the generations at which MRSC analysis was performed. The experiment could have been stopped after 50 generations with no loss of accuracy in the solutions. With the excessive number of generations, we hope to demonstrate that once the true function structure is revealed through speciation, the populations remain stable.

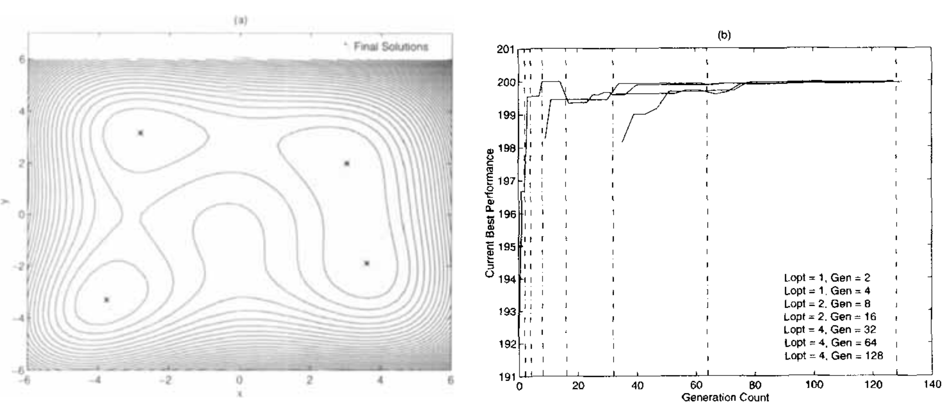

Figure 5. (a) Solution points for the modified Himmelblau's function generated by MRSC_GA. Final solutions are the best individual of each species at the last generation. (b) Current best performance of the populations for the modified Himmelblau's function. The plots also characterize the species formation and extinction as the GA runs.

Table 3. Solution points given by the MRSC_GA for decaying sinusoids.

	x	$f(x)$
1	0.06683	1.0000
2	0.26181	0.8472
3	0.45654	0.5153
4	0.65172	0.2251
5	0.84662	0.0706

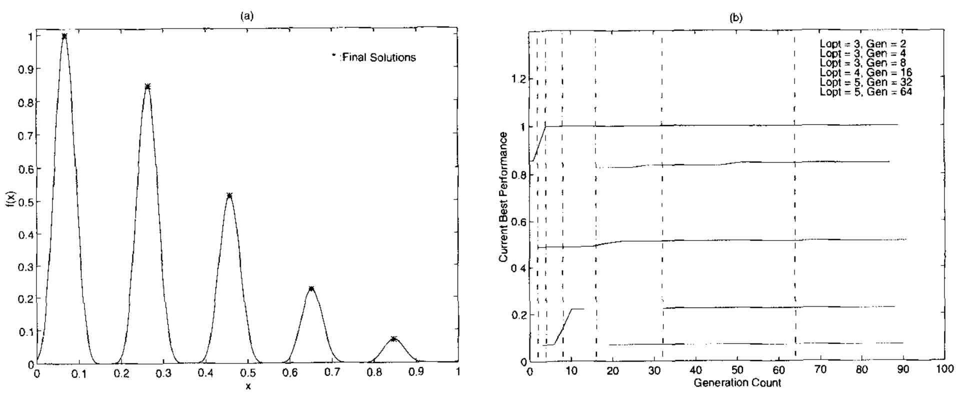

Figure 6. (a) Solution points for the decaying sinusoids generated by the MRSC_GA. Final solutions are the best individuals of each species at the last generation (Gen). (b) Current best performance of the populations for the decaying sinusoids. The plots also characterize the species formation and extinction as the GA runs.

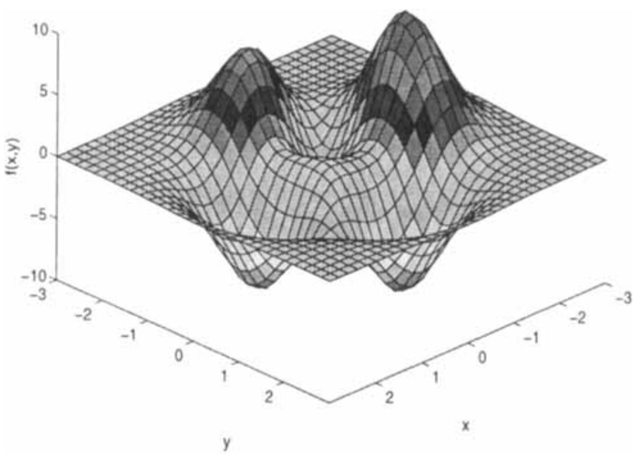

Figure 7. Valleys function.

Table 4. Solution points given by the MRSC_GA for the Valleys function.

	(x, y)	$f(x, y)$
1	(−0.566, −0.754)	−5.3574
2	(1.616, −0.109)	−7.8165
3	(−0.003, 1.587)	−8.0277

8.3 F3: Valleys Function

Valleys function is a function of two variables. It has been obtained by translating and rotating Gaussian distributions. It is defined by

$$F3(x,y) = -3(1-x)^2 * \exp(-x^2-(y+1)^2) + 10\left(\frac{x}{5} - x^5 - y^5\right) * \exp(-x^2-y^2) + 3 * \exp(-(x+1)^2 - y^2) \quad (12)$$

A three-dimensional surface plot of this function is shown in Figure 7. We represent three minima of interest by $(x, y, f(x, y))$ having approximate values of (−0.5507, −0.7607, −5.3628), (1.6016, −0.0059, −7.9015), and (0.0003, 1.5889, −8.0279).

The parameters with which the MRSC_GA was run for this function are given in Table 1 (column F3). Table 4 gives the solution points and their corresponding values obtained by the MRSC_GA run for this function. The results are satisfactorily accurate. Figure 8(a) shows the best individual of each species at the last generation. In this figure, regions of minima are represented by solid lines, and regions of maxima are shown by dashed lines. The minima of interest for this function are also discovered by the MRSC_GA algorithm. The best performance of each population at every generation is plotted in Figure 8(b). The graph shows that all three populations converge very closely to the corresponding optimal values. This graph also shows that at generation 2 and later at generation 4, new species have evolved. The GA run terminates with three species of interest, as shown by this graph.

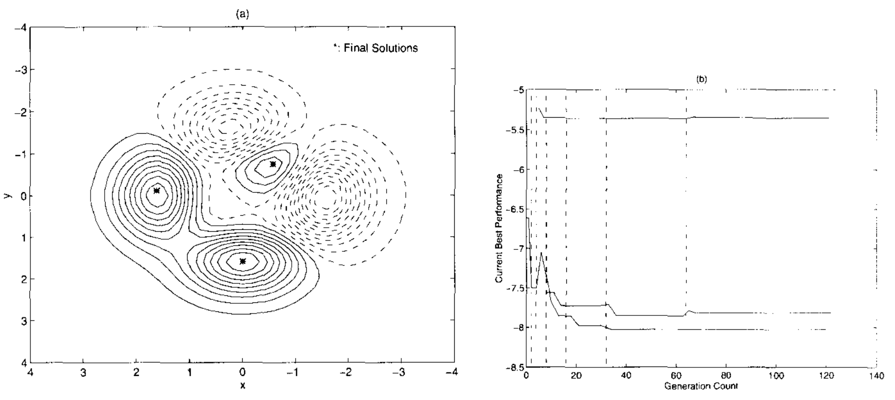

Figure 8. (a) Solution points for the Valleys function generated by the MRSC_GA. Final solutions are the best individual of each species at the last generation. Solid lines represent regions of minima, and dashed lines show regions of maxima. (b) Current best performance of the populations for the Valleys function. The plots also characterize the species formation and extinction as the GA runs.

9. Path Planning

The path-planning problem is known to be PSPACE-hard (Reif, 1979). Hence, researchers are actively seeking satisfactory, computationally efficient solutions. In general, the complexity of path planning increases exponentially with the dimension of the configuration space (Latombe, 1991).

In robotics, path planning is an active area of research. A review of the existing approaches for solving path-planning problems is provided in Latombe (1991). Recently, a few path planners have been developed using GAs (Davidor, 1991; Page, McDonnell, & Anderson, 1992; Shibata & Fukuda, 1993; Bessiere, Ahuactzin, Talbi, & Mazer, 1993; Ram, Arkin, Boone, & Pierce, 1994; Lin, Xao, & Michalewicz, 1995). Bessiere et al. (1993) describe a path planner for a robot moving in a dynamic environment. They use a combination of two local planning algorithms to build a global path planner. Lin et al. (1995) present an evolutionary planner/navigator for path planning and navigation. They utilize two GAs; one for off-line planning and the other for on-line planning.

The existing global approaches to path planning assume that a complete representation of the configuration space has been computed before searching for a path under a search map. The global approaches are, in general, complete in that if a path exists, it will be found. However, generating the complete configuration space is computationally very expensive. In general, the complexity grows exponentially with the number of degrees of freedom of a robot (Latombe, 1991). Local methods such as potential field approaches are more efficient techniques, but they could get trapped in a local minimum. Evolutionary computation techniques are proven to be effective in solving hard problems, such as the traveling salesman (Braun, 1990) and flow shop scheduling (Cleveland & Smith, 1989). Randomized search can be very efficient and effective in escaping local minima. GA-based approaches look for a solution in parallel where individual candidates interact through genetic operators to generate possibly better solutions. The individual candidates are evaluated with respect to the workspace, so that computation of the complete configuration space is not required. The GA-based approaches can also easily be implemented on a massively parallel machine to achieve superlinear speedup with the number of processors (Andre & Koza, 1996). The

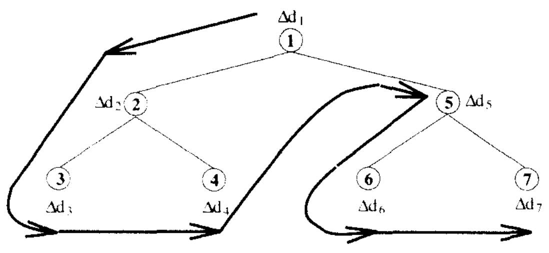

Figure 9. Genetic representation of a path.

evolutionary approaches to path planning are not confined to searching for a solution in a search map, which may affect the quality of the search.

The traditional planners cannot accommodate a variety of optimization criteria or allow changes in the optimization criteria without changing the characteristics of the planner or the search map. The GA-based approaches, on the other hand, can handle a variety of optimization goals and are very tolerant to the form of the evaluation function. Functions to be optimized need not be differentiable or continuous. The GA-based path-planning approaches are flexible to changes in the environment and are robust to uncertainties. The MRSC_GA algorithm is also capable of generating multiple solutions to the path-planning problem. The generation of alternative solutions is especially important for planning paths in dynamic environments.

10. MRSC_GA-Based Path Planning

The GA described in Section 7 is used to generate paths for robotics applications. An iterative multiresolution path representation is used as a basis for the GA coding. The use of a multiresolution path representation can reduce the expected search length for the path-planning problem. If a successful path is found early in the search hierarchy (at a low level of resolution), then further expansion of that portion of the path search is not necessary. This advantage is mapped into the encoded search space and adjusts the string length accordingly.

A path is represented by an ordered set of distances Δd_i. A Δd_i represents the signed distance from the perpendicular bisector of two two-dimensional knot points. Hence, each Δd_i represents an intermediate knot point. In general, $D - 1$ Δd_i's will be necessary to generate a point in D-dimensional space.

In this work, we represent a path using a complete binary tree, where the nodes of the binary tree correspond to Δd_i's, as shown in Figure 9. In D-dimensions, each node will have $D - 1$ Δd_i. Given the start and goal positions, the path represented by an ordered set of Δd_i's is generated by performing a *preorder* traversal of the complete binary tree, as shown in Figure 9. Figure 10 shows the construction of the path using Δd_i's given by the preorder traversal of the tree. Given the start S and goal G, each Δd_i corresponds to an intermediate knot point, as shown in Figure 11, where each distance is replaced by its corresponding knot point. The start position is appended as the left child of the leftmost node, and the goal position is appended as the right child of the rightmost node of the original complete binary tree. The *in-order* traversal of this binary tree provides the sequence of knot points representing the path, as described by the Δd_i's.

Given two knot points (P_a and P_b) and $D - 1$ Δd_i's in D-dimensional space, knot point P_t is constructed by making use of the Gram-Schmidt orthogonalization process. Figure 12 sets up the problem in two dimensions. After computing a unit direction vector (first basis

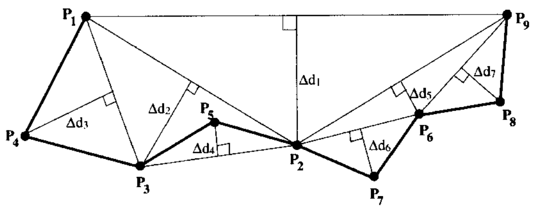

Figure 10. Construction of a path.

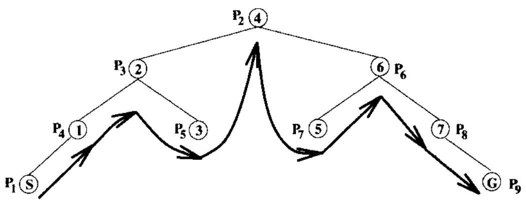

Figure 11. Knot point sequence.

vector) d_{cb}, $D-1$ D-dimensional unit vectors are generated using the modified Gram-Schmidt orthogonalization process. The computed set of basis vectors gives us an orthonormal transformation matrix **Trans**. The knot point P_t is then given by

$$P_t = P_c + \mathbf{Trans} * \mathbf{Dd},$$

where $\mathbf{Dd} = [0.0\ \Delta d_1\ \cdots\ \Delta d_i\ \cdots\ \Delta d_{D-1}]^T$.

In our current implementation, the complete binary tree is implemented as an array of Δd_i's. The path represented by the complete binary tree of Figure 9 looks like the following:

$$[\Delta d_3\quad \Delta d_4\quad \Delta d_2\quad \Delta d_1\quad \Delta d_5\quad \Delta d_6\quad \Delta d_7]$$

The MRSC_GA algorithm, which is described in Section 7, is used with this representation to find a (sub)optimal path. The evaluation criterion used to assign fitnesses to individuals is based on the number of collisions made by the knot points and the shortest path. The cost of a path is given by

$$\text{Cost(path)} = K_1 * \sum_{i=1}^{N-1} D(P_i, P_{i+1}) + K_2 * \sum_{i=2}^{N-1} F(P_i)$$

where $D(P_i, P_{i+1})$ gives the Euclidean distance between points P_i and P_{i+1}, and $F(P_i)$ returns a 1 if knot point P_i collides with an obstacle or a 0 otherwise; K_1 and K_2 are constants; and K_2 is large compared to K_1.

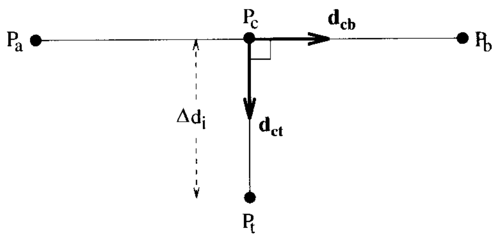

Figure 12. Knot point computation.

Table 5. GA parameters used for each planning environment.

	PLE_1	PLE_2	PLE_3
No. of Δd_i's	15	15	15
Initial No Pops	1	1	3
Max No Pops	1	1	3
Population Size	50	50	30
Crossover Rate	0.6	0.6	0.6
Mutation Rate	0.01	0.01	0.01
Max Func Evals	3000	3000	6000

The use of the knot point collisions as an evaluation measure is consistent with the applicability of the algorithm to higher dimensions. In higher dimensional path-planning problems, one will only need to perform *C-Space* mappings of the knot points for collision checking and avoidance. High-resolution path representation can be used to avoid infeasible path generation due to discrete representation of a path. In two-dimensional path planning, the feasibility of path segments can be used to ensure feasibility, but we have avoided using path segments for applicability of the evaluation criterion to higher dimensional spaces. The evaluation function based on knot point collisions provides a means of testing the feasibility of the representation and search algorithm and encourages the extension of this approach to more complex evaluation functions.

11. Path-Planning Simulation Results

In this section, we describe initial experiments conducted for planning paths in two-dimensional space. The results reported are obtained for the following planning environments:

- PLE_1: Planning a path for a point-robot.
- PLE_2: Piano-movers problem with no rotations.
- PLE_3: Planning multipaths for a point-robot.

Implementation of the MRSC_GA-based planning algorithm is not limited to two-dimensional path planning. It can also be used for planning paths in higher dimensional spaces. These issues are discussed in Section 12.

Table 5 gives the set of parameters used to run the experiments. In the evaluation function, K_1 was set to a value of 1, and K_2 was set to a value of 100. Figure 13 shows the results obtained for the planning environment called PLE_1. The paths generated for these cases are reasonable. In Figure 14, we show the results for the piano-movers problem with no rotations. The path generated is satisfactory for this planning environment. The results of multipath generation using the MRSC_GA-based planning algorithm are provided in Figure 15. Despite the simplicity of the experiments performed, the results obtained are very encouraging. The paths shown can easily be processed further to improve smoothness, time, and distance characteristics, but the original paths are shown here to better demonstrate

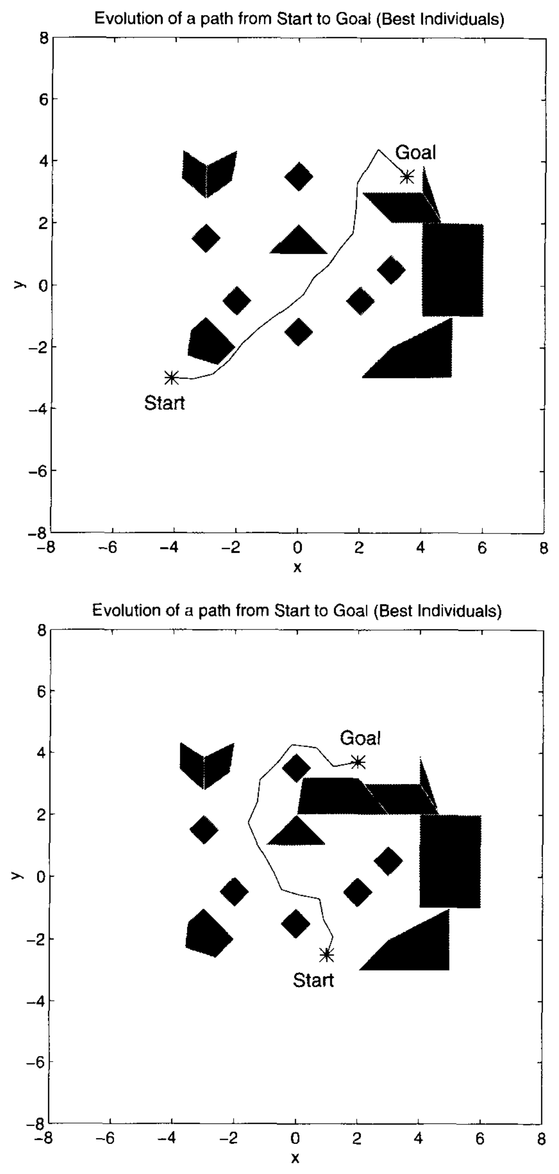

Figure 13. Paths generated using the MRSC_GA-based planning algorithm for PLE_1.

the inherent characteristics of the search algorithm. Further extensions and improvements to the existing planning algorithm are discussed next in Section 12.

12. Path Planner Extensions and Discussions

In this section, we briefly describe some of the future extensions that will be made to the planning algorithm. In the current MRSC_GA implementation for path planning, mutation and crossover operate on binary strings of individuals. Individuals that are represented by Δd_i's are converted into strings of binary numbers for genetic operations. This not only

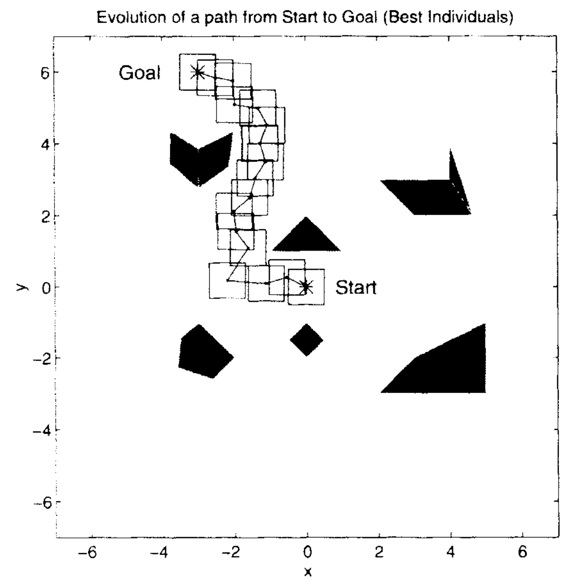

Figure 14. Path generated using the MRSC_GA-based planning algorithm for PLE_2.

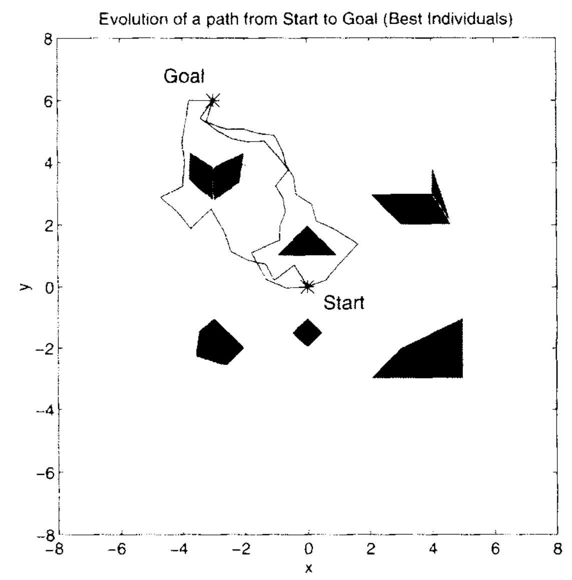

Figure 15. Multipaths generated using the MRSC_GA. *Cluster Interval* = 1, *Max Cluster Count* = 5.

requires additional time for conversions but also does not exploit any problem domain-specific knowledge that may improve the convergence to a near-optimal solution. Figure 16a shows a more natural mutation operator in the parameter space. Mutation of Δd_i's at lower levels of the tree changes the path more drastically. So, each level of the tree is given a different probability of undergoing mutation. Mutation probabilities can be adjusted dynamically to control the rate of mutation as the algorithm converges, for example, reducing the probabilities at lower levels of the tree (i.e., p_1) and increasing those at higher levels of the

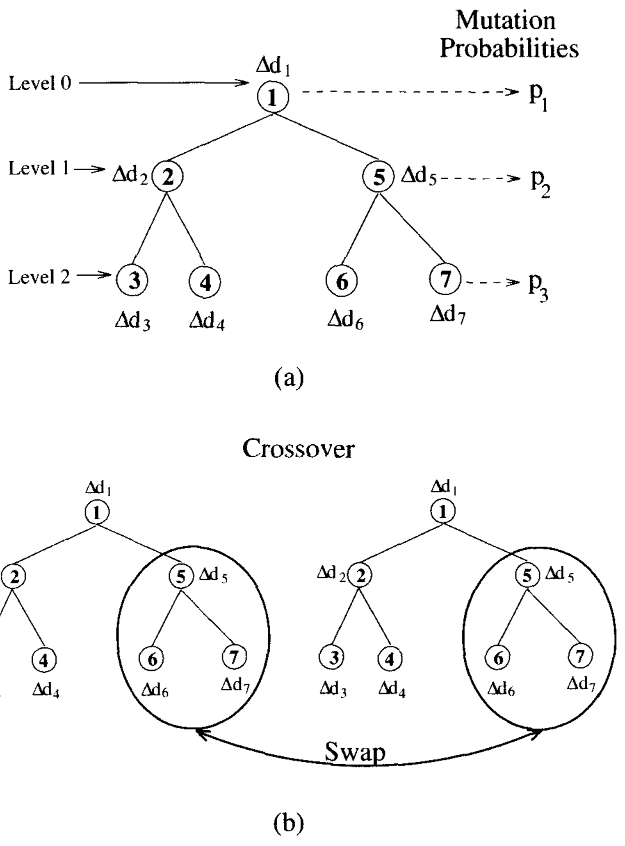

Figure 16. (a) Variable mutation probabilities. (b) Crossover operator swaps subtrees.

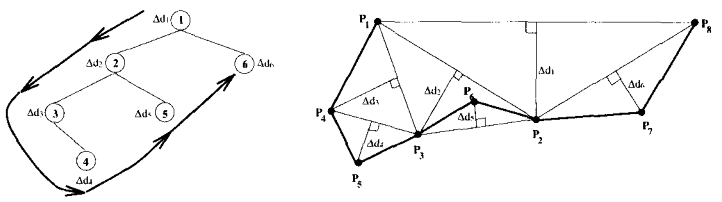

Figure 17. Variable number of knot points and nonsymmetric path representation.

tree (i.e., p_3). Swapping subtrees of the complete binary trees may provide the recombination operator in parameter space, as shown in Figure 16b.

A variable number of knot points can be used to represent different paths. This results in complete binary trees of different heights. The symmetric representation of paths using Δd_i's may also be broken by using binary tree representations, as shown in Figure 17. In these cases, additional genetic operators, such as additions and deletions of Δd_i's, can be used. Although these changes may result in more efficient representations, the results that may be achieved with these representations need to be studied. In high-dimensional spaces, it will be important to utilize the resolution of a path to perform feasibility testing so as to avoid *C-Space* mappings of the environment except at the knot points. This can either be achieved by using a complete binary tree representation with high-resolution knot points or by an unbalanced binary tree representation with an intelligent algorithm that controls the resolution, depending on the locality.

By using potential field functions, the evaluation criterion can also be improved in terms of clearances and avoiding previous collision points. Hybrid methods may lead into more powerful path-planning algorithms. Clustering algorithms exploiting the path-planning domain can be developed to improve multipath generation.

13. Conclusions

This paper has described a new approach to multimodal function optimization that utilizes the MRSC analysis to generate species even when the peaks of the function are not evenly spread throughout the search space. Although the cluster analysis increases the computational time required for convergence, the clustering is done only several times during the genetic search. So, the additional complexity introduced into the algorithm does not affect the performance adversely. The computation time can be improved considerably by making use of parallel implementations of the algorithm. The use of multiple populations is consistent with this line of thought.

During the species formation process, we make use of a KMEANS clustering algorithm. We use this algorithm because of its simplicity. In general, any cluster analysis algorithm, such as the maximum likelihood method (Fukanaga, 1972), may be used.

The simulation results demonstrated that the MRSC_GA we have presented performs well on the functions used in the literature and in the Valleys function we have introduced in this paper. The algorithm was also tested on a number of other functions, including several deceptive trap functions. The proposed algorithm is a powerful one in that it does not require a priori knowledge about the function landscape. For example, the number of local optima and how they are spread throughout the search space are not required by the MRSC_GA.

This paper has also described a new algorithm for solving path-planning problems for robotic applications. The planning algorithm utilizes the MRSC_GA for generating multiple solutions to the path-planning problem. The path-planning simulation results demonstrated that the presented MRSC_GA-based planning algorithm performs well on simple path-planning problems. The improvements discussed in Section 12 will lead into a more powerful path-planning algorithm.

Acknowledgments
This work was conducted at the Electronics Agile Manufacturing Research Institute (EAMRI) of Rensselaer Polytechnic Institute. The EAMRI is partially funded by Grant DMI-9320955 from the National Science Foundation (NSF) and the Defense Advanced Research Projects Agency (DARPA).

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, 19*, 716–723.

Andre, D., & Koza, J. R. (1996). A parallel implementation of genetic programming that achieves super-linear performance. In H. R. Arabnia (Ed.), *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications* (pp. 1163–1174).

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In J. J. Grefenstette (Ed.), *Proceedings of the Second International Conference on Genetic Algorithms and Their Applications* (pp. 14–21). Hillsdale, NJ: Lawrence Erlbaum Associates.

Beasley, D., Bull, D. R., & Martin, R. R. (1993). A sequential niche technique for multimodal function optimization. *Evolutionary Computation*, 1(2), 101–125.

Bessiere, P., Ahuactzin, J.-M., Talbi, E.-G., & Mazer, E. (1993). The "Ariadne's clew" algorithm: Global planning with local methods. In *Proceedings of the 1993 IEEE-IROS Conference on Intelligent Robots and Systems* (pp. 1373–1380). Piscataway, NJ: IEEE Press.

Braun, H. C. (1990). On solving travelling salesman problems by genetic algorithms. In H. R. Manne & D. H. Schwefel (Eds.), *Proceedings of Parallel Problem Solving from Nature (PPSN1)* (pp. 129–133).

Cavicchio, D. J. (1970). *Adaptive search using simulated evolution*. PhD thesis. Ann Arbor, MI: University of Michigan.

Chaitin, G. J. (1975). A theory of program size formally identical to information theory. *Journal of the Association for Computing Machinery, 3(July)*, 329–340.

Chen, Y., Nakao, Z., & Fang, X. (1996). Parallelization of a genetic algorithm for image restoration and its performance analysis. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation* (pp. 463–468). Piscataway, NJ: IEEE Press.

Cleveland, G. A., & Smith, S. F. (1989). Using genetic algorithms to schedule flow shop releases. In J. D. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms and Their Applications* (pp. 160–169). San Mateo, CA: Morgan Kaufmann.

Cohoon, J., Martin, W., & Richards, D. (1991). A multi-population genetic algorithm for solving the k-partition problem on hyper-cubes. In R. Belew & L. Booker (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms and Their Applications* (pp. 244–248). San Mateo, CA: Morgan Kaufmann.

Davidor, Y. (1991). *Genetic algorithms and robotics*. River Edge, NJ: World Scientific.

De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems*. PhD thesis. Ann Arbor, MI: University of Michigan.

Deb, K. (1989). *Genetic algorithms in multimodal function optimization* (TCGA Rep. No. 89002). Birmingham, AL: University of Alabama.

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

Fan, D. (1993). Gadelo: A multi-population genetic algorithm based on dynamic exploration of local optima. In S. Forrest (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms and Their Applications*. San Mateo, CA: Morgan Kaufmann.

Fukanaga, K. (1972). *Introduction to statistical pattern recognition*. New York: Academic Press.

Goldberg, D. E., & Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In J. J. Grefenstette (Ed.), *Proceedings of the Second International Conference on Genetic Algorithms and Their Applications* (pp. 41–49). Hillsdale, NJ: Lawrence Erlbaum Associates.

Hartigan, J. A. (1975). *Clustering algorithms*. New York: John Wiley.

Hocaoğlu, C., & Sanderson, A. C. (1995). Evolutionary speciation using minimal representation size clustering. In J. R. McDonnell, R. G. Reynolds, and D. B. Fogel (Eds.), *Evolutionary Programming IV: Proceedings of the Fourth Annual Conference on Evolutionary Programming*. Cambridge, MA: MIT Press.

Hocaoğlu, C., & Sanderson, A. C. (1996). Planning multi-paths using speciation in genetic algorithms. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation* (pp. 378–383). *Nagoya, Japan*. Piscataway, NJ: IEEE Press.

Iba, H., de Garis, H., & Sato, T. (1994). Genetic programming using a minimum description length.

In K. E. Kinnear, Jr. (Ed.), *Advances in genetic programming* (pp. 265–284). Cambridge, MA: MIT Press.

Kolmogorov, A. N. (1968). On the logical basis of information theory and probability theory. *IEEE Transactions and Information Theory, 14*, 662–664.

Latombe, J.-C. (1991). *Robot motion planning*. Boston: Kluwer Academic Publishers.

Levine, D. (1994). *A parallel genetic algorithm for the set partitioning problem*. PhD thesis. Chicago: Illinois Institute of Technology. Department of Computer Science.

Lin, H.-S., Xiao, J., & Michalewicz, Z. (1995). Evolutionary navigator for a mobile robot. In *Proceedings of the 1994 International Conference on Robotics and Automation* (pp. 2199–2204). San Diego.

Mahfoud, S. W. (1992). Crowding and preselection revisited. In R. Männer & B. Manderick (Eds.), *Proceedings of the Second Conference on Parallel Problem Solving from Nature* (pp. 27–36). Amsterdam, The Netherlands.

Mühlenbein, H., Schomisch, M., & Born, J. (1991). The parallel genetic algorithm as function optimizer. In R. Belew & L. Booker (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms and Their Applications* (pp. 271–278). San Mateo, CA: Morgan Kaufmann.

Page, W. C., McDonnell, J. R., & Anderson, B. (1992). An evolutionary programming approach to multi-dimensional path planning. In D. B. Fogel & W. Atmar (Eds.), *Proceedings of the First Annual Conference on Evolutionary Programming* (pp. 63–70). San Diego, CA: Evolutionary Programming Society.

Ram, A., Arkin, R., Boone, G., & Pearce, M. (1994). Using genetic algorithms to learn reactive control parameters for autonomous robotic navigation. *Adaptive Behavior, 2(3)*, 277–304.

Reif, J. H. (1979). Complexity of the mover's problem and generalizations. In *Proceedings of the IEEE Symposium on Foundations of Computer Science* (pp. 421–427). Piscataway, NJ: IEEE Press.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica, 14*, 465–471.

Sanderson, A. C., & Foster, N. J. (1989). Attributed image matching using a minimum representation size criterion. In *Proceedings of the 1989 IEEE Conference on Robotics and Automation* (pp. 360–365). *Scottsdale, Arizona*. Piscataway, NJ: IEEE Press.

Segen, J., & Sanderson, A. C. (1981). *Model inference and pattern discovery by minimal representation method* (CMU-RI-TR-82-2). Pittsburgh, PA: Carnegie-Mellon University, The Robotics Institute.

Shibata, T., & Fukuda, T. (1993). Robot motion planning by genetic algorithm with fuzzy critic. In *Proceedings of the Eighth IEEE International Symposium on Intelligent Control, Chicago*. Piscataway, NJ: IEEE Press.

Solmonoff, R. J. (1964). A formal theory of inductive inference. *Information and Control*, 7, 1–22; 224–254.

Wallace, C. S., & Boulton, D. M. (1968). An information measure for classification. *Computer Journal, 11(2)*, 185–194.

Whitley, D. (1993). An executable model of a simple genetic algorithm. In D. Whitley (Ed.), *Foundations of genetic algorithms 2* (pp. 45–62). San Mateo, CA: Morgan Kaufmann.

Yin, X., & Germay, N. (1993). A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms* (pp. 450–457). Berlin: Springer-Verlag.