Pergamon

Pattern Recognition, Vol. 27, No. 2, pp. 321–329, 1994
Elsevier Science Ltd

0031-3203/94 $6.00 + .00

CLUSTERING WITH EVOLUTION STRATEGIES

G. PHANENDRA BABU and M. NARASIMHA MURTY
Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560012, India

(*Received* 22 *January* 1993; *in revised form* 1 *September* 1993; *received for publication* 23 *September* 1993)

Abstract—The applicability of evolution strategies (ESs), population based stochastic optimization techniques, to optimize clustering objective functions is explored. Clustering objective functions are categorized into centroid and non-centroid type of functions. Optimization of the centroid type of objective functions is accomplished by formulating them as functions of real-valued parameters using ESs. Both hard and fuzzy clustering objective functions are considered in this study. Applicability of ESs to discrete optimization problems is extended to optimize the non-centroid type of objective functions. As ESs are amenable to parallelization, a parallel model (master/slave model) is described in the context of the clustering problem. Results obtained for selected data sets substantiate the utility of ESs in clustering.

Hard clustering Fuzzy clustering Optimal partition Evolution strategies

1. INTRODUCTION

Clustering methods play a vital role in data analysis. They have been effectively applied in different areas including image processing, and exploratory data analysis.[1] Various types of clustering algorithms have been proposed to suit different requirements. Broadly, clustering algorithms can be classified into hierarchical and partitional algorithms. Hierarchical methods build a dendrogram structure of the given data, whereas partitional methods divide the data into a specified or estimated (ISODATA)[1] number of non-overlapping clusters. Typically clustering algorithms aim at optimizing a chosen mathematical objective function. In this paper, we confine our discussion to partitional methods. In general, partitional methods are iterative hill climbing techniques that optimize the stated objective function and usually converge to a locally optimal partition. Clustering objective functions are highly non-linear and multi-modal functions. As a consequence, it is very difficult to find optimal partition of the given data using hill climbing techniques. Readers are referred to references (1–3) for a detailed coverage on cluster analysis and its applications.

The number of possible partitions of a given data set with T data items and C clusters is given by the stirling approximation[2]

$$S = \frac{1}{C!} * \sum_{k=1}^{C} \left((-1)^{C-k} * \binom{C}{k} * k^T \right). \qquad (1)$$

It can be observed that for $T = 100$ and $C = 2$, we need to search $2^{99} - 1$ possible partitions in order to arrive at an optimal partition. So exhaustive enumeration is ruled out due to its exponential time complexity. Many attempts have been made to reduce the computational effort in finding the optimal partition. Some techniques such as integer programming,[4] dynamic programming[5] and branch and bound[6] methods have been applied to reduce the computational burden of exhaustive enumeration. Even though these methods are better than exhaustive enumeration, these are still computationally expensive for moderate and large values of T and C (> 2).

Stochastic optimization approaches such as simulated annealing and genetic algorithms have been used in the context of the optimal clustering problem.[6,7] Theoretically, it is possible to obtain optimal solutions using these methods, but in practice only near-optimal solutions will be obtained except in some special circumstances. References (6, 7) solve the afore-mentioned problem by formulating it as a discrete optimization problem, i.e. assigning each of the data items to one of the clusters. These approaches take large amounts of time to converge to globally optimal partition. An alternative way of solving the optimal partition problem is to find optimal initial seed values for which the selected algorithm converges to a globally optimal solution. Both the above stochastic methods are applied to select optimal initial seed values[8,9] and were reported to produce superior results. A centroid type of objective function, within group sum of squared (WGSS) error measure, was considered. However, the same approach can be extended to optimize the non-centroid type of objective functions also. Here, a centroid type of objective function refers to one which depends on the centroids of the clusters formed, whereas a non-centroid type of objective function does not involve cluster centers in its formulation.

Hard clustering deals with assigning each data item to exactly one of the clusters. Whereas fuzzy clustering extends this concept to associate each data item to each of the clusters with a measure of belongingness. We restrict our discussion to fuzzy C-means (FCM) clustering objective functions. Fuzzy C-means objective function, an extended form of the hard WGSS criterion, was first proposed by Dunn.[10] Later a generalized form of the FCM algorithm with a family of objective functions, $J_m(\cdot)$, $1 \leq m < \infty$, was proposed by Bezdek.[11]

Dunn's formulation is a special case ($m = 2$) of the above formulation. Convergence proof of the FCM clustering algorithm has been studied in references (12, 13) and it is stated that the FCM algorithm converges to a locally optimal solution or to a saddle point. Further, it is mentioned that methods to avoid saddle points are required in order to ensure that the FCM algorithm converges to a local optimum. In our study, we observe that the problem of local minima or saddle points is severe when the value of m, discussed in Section 3, is closer to 1.

As far as we know, no attempts have been made to find globally optimal values of fuzzy clustering objective functions. This entails optimizing an objective function with $T \times C$ continuous valued parameters subjected to some constraints. In this paper, we employ ESs to obtain the global optimal solution for fuzzy clustering objective functions.

This paper is organized as follows: Section 2 presents a brief review of evolution strategies. The problem formulation is presented in Section 3. Experimental study and results are presented in Section 4. A possible method for solving non-centroid type of objective functions using discrete optimization formulation is discussed in Section 5. A parallel design for the proposed method is described in Section 6.

2. EVOLUTION STRATEGIES

Evolution strategies have been introduced by Rechenberg[14] and are further developed by Schwefel.[15] They are modeled using biological evolution concepts. These methods adopt principles of the evolutionary process: selection, recombination and mutation. The first proposed ESs have been used to optimize mathematical functions with continuously changeable parameters and later have been extended to solve discrete optimization problems. As these methods do not assume any information about the functions and do not impose any restrictions such as continuity and differentiability on the functions to be optimized, these methods found their applications in solving many optimization problems including the traveling salesman problem (combinatorial optimization problem), PID (Proportional, Integral and Derivative) regulator with highly nonlinear system (control problem), and some discrete optimization problems.[16,17] The basic ESs are extended to support multi-criteria decision problems.

Evolution strategies are powerful function optimization tools, and can be applied to wide varieties of problems in various fields. The function for which ESs are applied should support strong causality, i.e. small changes in the parameters must result in small changes in the objective or quality function value. For discrete optimization problems, it is evident that slight perturbation to a solution yields a small change in the objective function value. If the problem does not possess strong causality, some special modifications are required to solve it by ES.

Evolution strategies belong to a class of population based approaches. Here population consists of a set of parents, U, and a set of offspring, Θ. Each parent or offspring represents a solution in the search space and is denoted by a string of parameters. Offspring in the population are produced by mutation and recombination operations over the parents in the same population. Parents in the next population are generated by selecting potential solutions in the previous population. The process of evolving the next population from the current population is called *generation*. This process of generating new population ends when optimal or near-optimal solution(s) are obtained or a limit on the number of generations is reached.

In the early model of evolution strategies, proposed by Rechenberg, only one parent ($|U| = \mu = 1$) and an offspring ($|\Theta| = \lambda = 1$) constituted a population. The mutation operator is used to generate offspring, whereas the selection operator is used to select a possible parent for the next population. The offspring is generated, through the mutation operation, from the parent by adding normally distributed random values with zero mean and with a specified variance. The best of the two solutions in the current population based on their fitness or merit (proportional to objective function value) is taken as the parent in the next population. In the $(1 + 1)$-ES, i.e. $\mu = 1$ and $\lambda = 1$, the selection operator considers both parent and offspring $(+)$. The $(1 + 1)$-ES is a sort of probabilistic hill climbing technique and the concept of population and recombination are not exploited. Later multimembered ESs have been proposed where parents ($\mu > 1$) participate in the generation of offspring giving rise to $(\mu + 1)$-ES. In $(\mu + 1)$-ES the recombination operator is introduced, i.e. two parents are selected with uniform mating probability and their features are combined and mutated to produce an offspring. Here the selection operator places the generated offspring into the population by replacing the least fit individual or solution string following the survival of the fittest guidelines.

The $(\mu + \lambda)$-ES and (μ, λ)-ES where $\mu > 1$ and $\lambda > 1$, were proposed to make use of parallel computers and to enable self adaptation of strategic parameters.[18] The $(\mu + \lambda)$-ES uses all individuals in the process of selecting parents for the next population, whereas (μ, λ)-ES uses only offspring Θ in this process. It is assumed that $\mu < \lambda$ and each parent contributes in the generation of (λ/μ) offspring in a population. A formal description of multimembered ESs is given below:

Π_i = ith population,
U, Θ = set of parents and offspring respectively,
$I = (x, \sigma)$ is an individual, where x is a solution vector of size n and σ is a mutation variance vector of size n,
$S(\cdot)$ = selection operator,
  $S(U \cup \Theta)$ for $(\mu + \lambda)$-ES type
  $S(\Theta)$ for (μ, λ)-ES type,
$R(\cdot)$ = recombination operator,
$M(\cdot)$ = mutation operator,
$U = \{I_1, I_2, I_3, \ldots, I_\mu\}$,

$\Theta = \{I_{\mu+1}, I_{\mu+2}, \ldots, I_{\mu+\lambda}\}$,
$\Pi_i = U_i \cup \Theta_i$,
$\Delta\sigma$ = global step size variance,
$f: x^n \to R$ is an objective function.

Parents in the initial population are selected randomly with uniform distribution and offspring in that population are produced using recombination and mutation operators. Parents in the next population are selected from the previous population depending on the strategy used ("+" or ","). Each operator is described below.

Recombination operator $R(\cdot)$: $R(I', I'') \to I$. Two individuals I' and I'' are selected with equal mating probability from the parent set U. Each individual comprises of a solution vector and a variance vector

$$I' = (x', \sigma')$$
$$I'' = (x'', \sigma'')$$
$$I = (x, \sigma).$$

There are many types of recombination operators and five of them are mentioned in reference (15). We consider two out of those five which seem to work well for the clustering problem.

Discrete recombination: R_1

$$x_i = x_i' \text{ or } x_i''$$
$$\sigma_i = \sigma_i' \text{ or } \sigma_i'', \forall i.$$

Intermediate recombination: R_2

$$x_i = \tfrac{1}{2}(x_i' + x_i'')$$
$$\sigma_i = \tfrac{1}{2}(\sigma_i' + \sigma_i''), \forall i.$$

We use these two operators in our experimental study.

Mutation operator $M(I) \to I'$. The mutation operator takes an individual and the following steps are executed to produce a mutated individual. It can be observed that mutation information is also a part of the individual

$$I = (x, \sigma)$$
$$I' = (x', \sigma')$$
$$\sigma_i' = \sigma_i \exp(N_0(0, \Delta\sigma))$$
$$x_i' = x_i + N_0(0, \sigma_i'), \forall i.$$

where $N_0(0, \sigma)$ generates normally distributed random variates with mean zero and variance, σ.

Selection operator $S(\Omega) \to U$. The set Ω is $U \cup \Theta$ for $(\mu + \lambda)$-ES and is Θ for (μ, λ)-ES. The selection operator selects the best μ individuals from the set Ω to produce parents in the next generation.

It is suggested that by equating λ/μ ratio to 5 or 6 gives the maximum convergence rate.[15] The algorithm for the multi-membered ES is given below.

Algorithm
Input:
U = Initial set of parents,
μ = Number of parents, λ = Number of offspring,
max_gen = Maximum number of generations;
Output: Solution String S;
begin
  $s = 0$;
  no_generation = Max_gen,
  calculate fitness values (f) of solutions in U,
  while (*no_generation* > 0)
  begin
    Θ = NIL
    while ($|\Theta| < \lambda$)
    begin
      x = Rand$(1, \mu)$; y = Rand$(1, \mu)$;
      select I_x and I_y from U;
      $I' = M(R(I_x, I_y))$;
      $\Theta = \Theta \cup \{I'\}$;
      Compute $f(I')$;
    end
    if ($f(s)$ > fitness value of best individual) replace S by that string,
    if $f(s)$ is near-optimal or optimal) output S and exit,
    else *no_generation* = *no_generation* − 1,
    $U = S(U \cup \Theta)$; /* if "+" type selection operation is used. */
  end
end.

There is no concrete proof that ESs converge to a global minimum. In the next section, we discuss the formulation of the centroid type of clustering objective functions for both hard and fuzzy clustering paradigms.

3. CLUSTERING WITH ESs

In the above section, we described how evolution strategies can be used to optimize a function with real-valued parameters. In this paper, we restrict our discussion to the centroid type of clustering objective functions which can be posed as real-valued parameter optimization problems. In Section 5, we present how the discrete optimization formulation of the clustering objective function can be solved using ESs.

The centroid type of clustering objective functions, i.e. objective functions that depend on the centroids of the clusters formed, can be formulated as real-valued parameter optimization problems so that a search can be performed to locate optimal cluster centers in order to produce an optimal partition of the given data. In our study, we consider WGSS objective function for hard clustering and FCM objective functions for fuzzy clustering. The evaluation function f, discussed in the above section, has to be defined in each case.

Hard clustering

Let

$V = \{v_1, v_2, \ldots, v_T\}$ be a set of T data vectors each of d dimensions,

$O = \{c_1, c_2, \ldots, c_C\}$ be a set of centroids each of d dimensions.

Optimization function is defined as:

$$\text{minimize } J(W, O) = \sum_{i=1}^{C} \sum_{j=1}^{T} u_{ij} D^2(v_j, c_i) \tag{2}$$

$$\text{subject to } \sum_{i=1}^{C} u_{ij} = 1; u_{ij} \in \{0, 1\}$$

$$L(l) \le C_i(l) \le R(l), \quad 1 \le l \le d, 1 \le i \le C,$$

where $W = \{u_{ij}\}$ is a $T \times C$ association matrix,

$$D(v, c) = \sqrt{\left(\sum_{l=1}^{d} (v(l) - c(l))^2 \right)}$$

$$L(l) = \min_j (x_j(l)), R(l) = \max_j (x_j(l)), 1 \le j \le T.$$

Now the problem is reduced to optimize $J(\cdot)$ with the above-mentioned constraints. Let W^* and O^* be the configurations of W and O at the global optimal point, respectively. There are two ways of finding the global optimal value of $J(\cdot)$.

Procedure 1. Find W^* such that $J(W^*, O)$ produces the globally optimal value.

Procedure 2. Find O^* such that $J(W, O^*)$ produces the globally optimal value.

Procedure 1 is a discrete optimization problem and is independent of whether the objective function depends on cluster centers or not. For the non-centroid type of objective functions O will not appear in the formulation and these functions require a discrete optimization problem to be solved. For the centroid type of objective functions that depend on O also, we can obtain the optimal function value by solving Procedure 2. The objective or evaluation function, defined in Section 2, takes a real-valued vector as input and returns a real value that acts as fitness value.

Let x be a solution and be represented by a vector of real numbers $x_1 x_2 \cdots x_n$ and $n = C \times d$ where d is the number of dimensions and C the number of clusters. The hard clustering evaluation function $f_h(x)$ is computed as follows:

$f_h(x)$:

1. $c_i(l) = x((i-1)*d + l), 1 \le i \le C, 1 \le l \le d,$
2. assign data items $v_1, v_2, \ldots, v_T$ to the nearest centers, and obtain W,
3. recompute centers $c_1, c_2, \ldots, c_C$ from the W and V,
4. compute error $J()$ from equation (2),
5. return $J()$ value.

The above objective function is used to solve Procedure 2. Experimental results pertaining to hard clustering are discussed in the next section.

Fuzzy clustering

The FCM clustering objective functions are considered in this study. The formulation of the FCM objective function proposed by Bezdek is presented below. The FCM algorithm starts either with initial cluster centers O or with initial fuzzy assignment matrix W. Equations (4) and (5) are executed in a sequence until the desired stable configuration with respect to a termination condition is reached. The fuzzy C-means clustering objective function is described below:

optimization function is defined as:

$$\text{minimize } J_f(W, O) = \sum_{i=1}^{C} \sum_{j=1}^{T} (u_{ij})^m D^2(v_j, c_i) \tag{3}$$

$$\text{subject to } \sum_{i=1}^{C} u_{ij} = 1; u_{ij} \in [0, 1];$$

$$L(l) \le C_i(l) \le R(l), 1 \le l \le d, 1 \le i \le C,$$

where

$W = \{u_{ij}\}$ is a $C \times T$ association matrix,

$$D(v, c) = \sqrt{\left(\sum_{l=1}^{d} (v(l) - c(l))^2 \right)},$$

$$L(l) = \min_j (x_j(l)), \quad R(l) = \max_j (x_j(l)), \quad 1 \le j \le T,$$

$$c_j = \frac{\sum_{i=1}^{T} (u_{ij})^m v_i}{\sum_{i=1}^{T} (u_{ij}^m)} \tag{4}$$

$$u_{ij} = \frac{1}{\sum_{r=1}^{C} \left(\frac{D(v_i, c_j)}{D(v_i, c_r)} \right)^{1/(m-1)}} \tag{5}$$

termination condition:

$\|W_t - W_{t+1}\| \leq \varepsilon$, where ε is a small value (say 0.001).

Solving for W^* or O^* involves real-valued parameter optimization. $W(\in R^{CT})$ is a real-valued matrix. If either of W^* or O^* is found, the other follows from equations (4) and (5).

In order to find W^*, we need to operate on $C \times T$ parameters, whereas finding O^* requires optimizing a function with $C \times d$ parameters. It is evident that in most of the problems, we encounter $d < n$, so solving for O^* is computationally more efficient than solving for W^*. Thus the problem reduces to finding optimal cluster centers that produce optimal objective function value. Reference (12) gives a mathematical procedure to verify whether the objective function converged to a local minimum or to a saddle point.

Evaluation function: the ES is used to optimize $J()$. Here the solution string $x = x_1 \cdots x_n$ represents a solution in the continuous search space. The function $f_c(x)$ returns FCM function value for a specified value of m. Steps for computing $f_t(x)$ for a given solution x are given below:

$f_t(x)$:

1. $c_i(l) = x((i-1)*d + l), 1 \leq i \leq C, 1 \leq l \leq d$,
2. compute association measures u_{ij} from equation (5),
3. recompute centers $c_1, c_2, \ldots, c_C$ from the W and V using equation (4),
4. compute error $J()$ from equation (3),
5. return $J()$ value.

The above evaluation function is used to evaluate the merit of each individual and $J()$ is optimized using ESs to obtain the optimal or the near-optimal function value. Experimental results for selected data sets are presented in the next section.

4. EXPERIMENTAL STUDY AND RESULTS

In our experimental study, we tested many data sets with different numbers of clusters. In almost all cases, the proposed model could find an optimal or a near-optimal partition of the data. The convergence rate is dependent on the number of parameters, global step size variance, and initial step variances (σ). Very small step variances slow down the convergence rate, whereas very large step variances may totally miss the optimal solution making the search a random one. We observe that the initial convergence rate is very high and as the generations progress convergence rate decreases rapidly. We stop after some generations and the clustering algorithm, corresponding to the selected objective function is run by considering the best solution available as the initial seed vector. This guarantees near-optimal local minimum. Here, results for some selected data sets are presented. The British Towns Data (BTD)[19] with 50 data items is tested with different numbers of clusters for the hard clustering problem. Some data sets have been chosen from reference (12) for testing the fuzzy clustering objective function along with BTD.

We set $\mu = 10$ and $\lambda = 60$ and global variance $\Delta\sigma = 0.01$ throughout our studies. Both R_1 and R_2 recombination operators are equally good and there is no substantial difference observed. We used R_1 for hard clustering and R_2 for fuzzy clustering. Experiments were conducted on a CD4360 mini-frame machine. For hard clustering, we tested the BTD for $C = 6$, 8 and 10 and the convergence results are shown in Figs 1–3, respectively. The dimensionality of the BTD is 4, so in the first case ES optimizes 24 parameters, in the second case ES optimizes 32 parameters and in the

Table 1

		C	$J()$
BTD		6	141.46
hard		8	113.505
clustering		10	93.941
Fuzzy	Ex. 1	4	113.039
clustering	Ex. 2	2	215.438
	Ex. 3	3	3.166

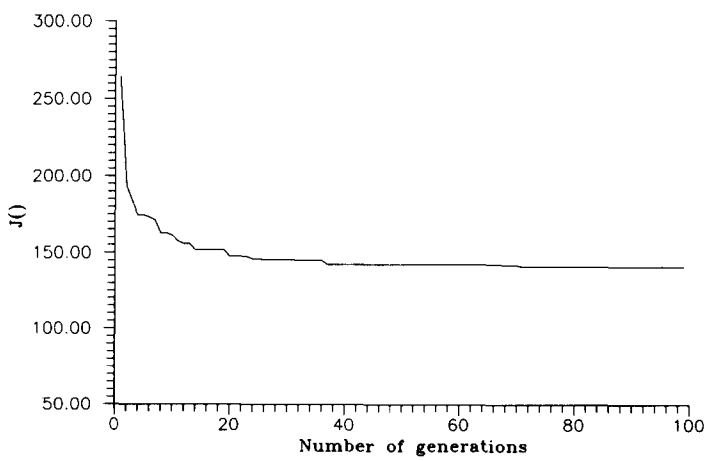

Fig. 1. British Towns Data, $T = 50$, $C = 6$.

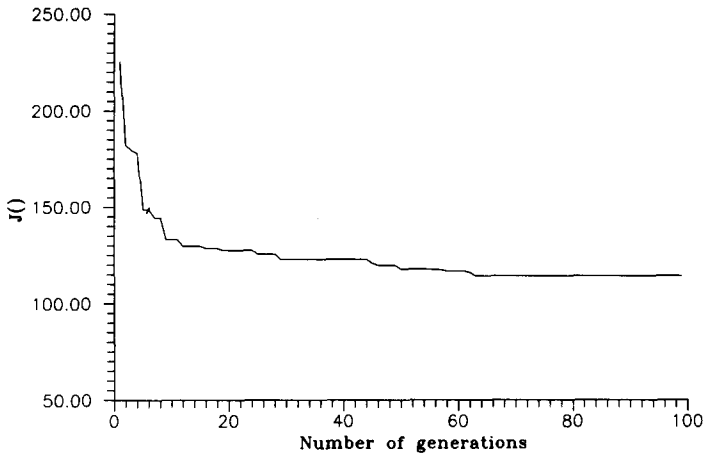

Fig. 2. British Towns Data, $T = 50$, $C = 8$.

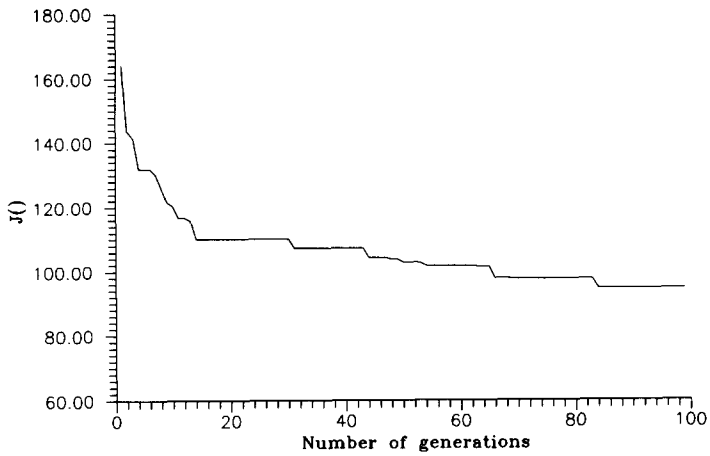

Fig. 3. British Towns Data, $T = 50$, $C = 10$.

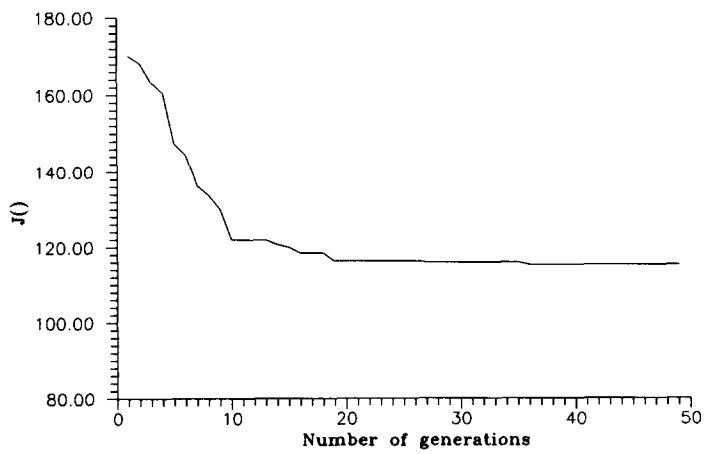

Fig. 4. British Towns Data, $C = 4$, $m = 2.0$, $T = 50$.

last case it optimizes 40 parameters. As the number of parameters increases, the convergence rate decreases.[15] Best solutions in each simulation for $C = 6$, 8 and 10 are taken and the K-means algorithm is run over each solution corresponding to each type and the results are shown in Table 1. Instead of waiting for the ES to converge to an exact optimal solution, it is better to stop after a number of generations when the convergence rate becomes very slow and run the corresponding clustering algorithm in order to reach the appropriate (near-optimal) local minimum.

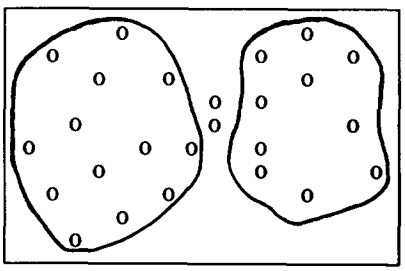

Fig. 5. Touching Clusters Data.

We tested some data sets presented in reference (12) that have a tendency to converge to saddle point(s). Evolution strategy can easily avoid saddle points as it is a population based probabilistic search algorithm. Results in fuzzy clustering with ES are encouraging and we present convergence results for chosen data sets.

Example 1. The BTD, with parameters: $T = 50$, $m = 2.0$ and $C = 4$, is tested and the convergence results are shown in Fig. 4. After going through a sufficient number of generations, i.e. when the convergence rate becomes small, the FCM algorithm is run by taking the best available solution as the initial seed (center) vector.

Example 2. This data is taken from Example A in reference (12) and consists of two touching clusters with two inliers as shown in Fig. 5. The data is provided in the Appendix. This data was used to find inliers using the FCM algorithm. We set $m = 2.0$ and $C = 2$. The convergence result is shown in Fig. 6.

Example 3. This is a symmetric data, Example E in reference (12). It is shown that this data converges to

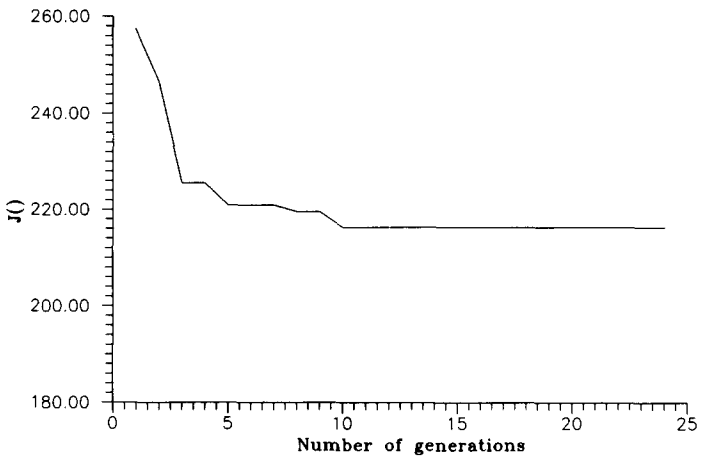

Fig. 6. Touching Clusters Data, $C = 2$, $m = 2.0$, $T = 25$.

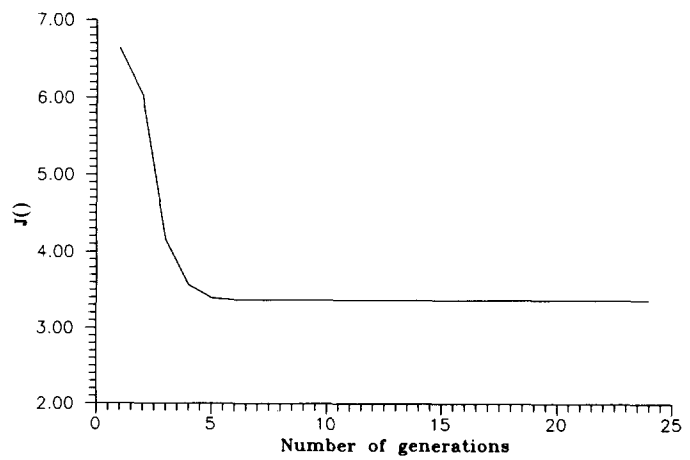

Fig. 7. Symmetry Data, $T = 20$, $C = 3$, $m = 3.0$.

a saddle point for $m = 3.0$ and $C = 3$. This data consists of 20 data items in a 2D plane and is obtained by adding an equidistant vector to four points $(-3.0, 0.0)$, $(-1.0, 0.0)$, $(1.0, 0.0)$ and $(3.0, 0.0)$. The data is provided in the Appendix. The ES was run for 25 generations and the convergence result is shown in Fig. 7.

5. NON-CENTROID TYPE OF CLUSTERING OBJECTIVE FUNCTIONS

In this paper, we explained primarily how ESs can be used effectively to find the optimal partition for the centroid type of clustering objective functions. However, there are many non-centroid type objective functions which require to solve discrete optimization formulation (Procedure 1). In this section we discuss a method for solving discrete optimization formulation with ESs.

Each individual I is of the form $I = (x)$ and does not have any mutation information. The entire mutation is controlled by global mutation step size $\Delta\sigma$ as in the earlier case. Solution string x is an ordered sequence of T discrete parameters, each one represents the cluster label of a data item. Each parameter can assume a discrete value from the set $\{1, 2, \ldots, C\}$.

There is no change in the recombination and selection operators except intermediate recombination is not valid in this context. The mutation operator takes x as input and modifies some of the parameters producing a mutated solution x'. The number of mutations, i.e. number of parameter changes is controlled by a global parameter.

The evaluation function $f()$ takes a solution string as input and forms clusters based on the data assignment. Computation of objective function value is straightforward and the objective function value is used as fitness or figure of merit of the corresponding solution string. Evolution strategies can be used to find optimal assignment matrix W^*.

6. PARALLEL MODEL

Most of the execution time of the ES algorithm is spent in the evaluation of objective function value for a given solution. This multimembered ES can easily be implemented on the available parallel hardware that gives linear speedup. For more details readers are referred to references (15, 18, 20). In this section, we present a parallel model where more than one processor is available. This design is a simple one, shown in Fig. 8, and more involved designs can be thought of. The number of processors is proportional to the number of parents μ.

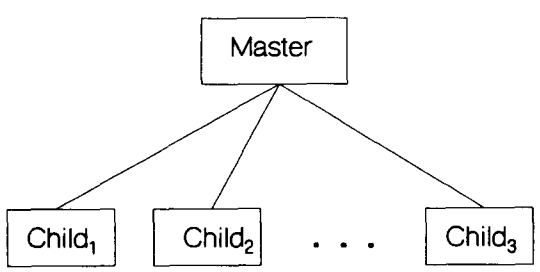

Fig. 8.

The central processor maintains a set of parent solutions and its only job is selection operation and pass parent solutions to child processors. A random initial set of parent solutions is fed to the central processor. The central processor selects the mates and passes those mates to the child nodes in a sequence. The child processors perform mutation and recombination operations and evaluate the objective function values. Newly formed solution(s) along with their fitness values are sent back to the central node. Then the central node, after receiving all child node responses, performs selection operation to select a new set of parents for the next generation. This process is continued until a near-optimal or optimal solution is found or a limit on the number of generations is reached. This design gives a speedup of $A - \gamma$, where γ is the communication overhead, and A the number of child processors.

7. CONCLUSIONS

In this paper, we have explored the use of ESs for solving the optimal clustering problem. Centroid type of clustering objective functions are posed as real-valued parameter optimization problems and are solved using ESs to find optimal value. Both hard and fuzzy clustering objective functions have been considered in this study. The FCM clustering algorithms have a tendency to converge to saddle points and this problem is overcome by exploiting the stochastic nature of ESs. Evolution strategies for solving discrete optimization formulation have been presented. A parallel model to obtain linear speedup has been discussed.

REFERENCES

1. A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Prentice-Hall, Englewood Cliffs, New Jersey (1989).
2. M. R. Anderberg, *Cluster Analysis for Applications*. Academic Press, London (1973).
3. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Wiley, New York (1973).
4. R. E. Jenson, A dynamic programming algorithm for cluster analysis, *Operations Res.* **12**, 1034–1057 (1969).
5. W. L. G. Koontz, P. M. Narendra and K. Fukunaga, Branch and bound clustering algorithm, *IEEE Trans. Comput.* **C-23**, 908–914 (1975).
6. R. W. Klein and R. C. Dubes, Experiments in projection and clustering by simulated annealing, *Pattern Recognition* **22**, 213–220 (1989).
7. V. V. Raghavan and K. Birchand, A clustering strategy based on a formalism of the reproduction process in natural system, *SIGIR Form* **14**, 10–22 (1979).
8. G. P. Babu and M. N. Murty, A near-optimal initial seed value selection in K-Means algorithm using a genetic algorithm, *Pattern Recognition Lett.* **14**, 763–769 (1993).
9. G. P. Babu and M. N. Murty, Simulated annealing for selecting initial seeds in the K-Means algorithm, accepted in *Indian J. Pure Appl. Math.*
10. J. C. Dunn, A fuzzy relation of the ISODATA process

and its use in detecting compact well-separated clusters, *J. Cybernet.* 32–57 (1974).
11. J. C. Bezbek, *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, New York (1983).
12. T. Kim, J. C. Bezdek and R. J. Hathaway, Optimal tests for the fixed points of the fuzzy C-means algorithm, *Pattern Recognition* **21**, 651–663 (1988).
13. J. C. Bezdek, A convergence theorem for fuzzy ISODATA clustering algorithm, *IEEE Trans. Pattern Analysis Mach. Intell.* **PAMI-2**(1), 1–8 (1980).
14. I. Rechenberg, *Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der biologischen Evolution.* Frommann-Holzboog, Stuttgart (1973).
15. H.-P. Schwefel, *Numerical Optimization of Computer Models.* Wiley, New York (1981).
16. M. Herdy, Application of the 'Evolutionsstrategie' to discrete optimization problems, *Parallel Problem Solving from Nature, Ist Workshop, PPSNI Proceedings*, Dortmond, H.-P. Schwefel and Manner, eds, pp. 188–192. Springer, Berlin (1990).
17. I. Rechenberg, Evolution strategy: nature's way of optimization, *Optimization: Methods and Applications, Possibilities and Limitations*, H. W. Bergmann, ed., Lecture Notes in Engineering, pp. 106–126. Springer, Berlin (1989).
18. F. Hoffmeister and T. Back, Genetic algorithms and evolution strategies: similarities and differences, Technical Report SYS-1/92, Department of Computer Science, University of Dortmund, Germany (1992).
19. Y. T. Chien, *Interactive Pattern Recognition.* Marcel Dekker, New York (1978).
20. R. Lohmann, Application of evolution strategy in parallel populations, *Parallel Problem Solving from Nature, Ist Workshop, PPSNI Proceedings*, Dortmond, H.-P. Schwefel and Manner, eds, pp. 198–208. Springer, Berlin (1990).

APPENDIX

(1) Data set for Example 2: this set consists of 25 data vectors in the 2D plane and are provided in Table A1.

(2) Data set for Example 3: this set consists of 20 data vectors in the 2D plane and are provided in Table A2.

Table A1

Touching clusters data (x_1, x_2)	(x_1, x_2)
(2, 11)	(3, 7)
(3, 13)	(4, 15)
(4, 10)	(5, 8)
(5, 12)	(6, 14)
(6, 6)	(7, 11)
(8, 8)	(8, 13)
(9, 11)	(10, 10)
(12, 7)	(12, 9)
(12, 12)	(14, 6)
(14, 8)	(13, 11)
(14, 13)	(16, 7)
(16, 10)	(17, 12)
(10, 9)	

Table A2

Symmetric data (x_1, x_2)	(x_1, x_2)
(−3.1, 0.1)	(−2.9, −0.1)
(−2.9, 0.1)	(3.1, −0.1)
(−3.0, 0.0)	(−1.1, 0.1)
(−1.1, −0.1)	(0.9, 0.1)
(−0.9, −0.1)	(−1.0, 0.0)
(−0.9, 0.1)	(0.9, −0.1)
(1.1, 0.1)	(1.1, −0.1)
(1.0, 0.0)	(2.9, 0.1)
(2.9, −0.1)	(3.1, 0.1)
(−3.1, −0.1)	(3.0, 0.0)

About the Author—G. PHANENDRA BABU is a research scholar in the Department of Computer Science and Automation at Indian Institute of Science, Bangalore, India. His areas of research interest include optimal clustering, evolutionary algorithms, neural networks, and knowledge based systems.

About the Author—M. NARASIMHA MURTY is an Associate Professor in the Department of Computer Science and Automation at Indian Institute of Science, Bangalore, India. His areas of research interest include pattern recognition, genetic algorithms, neural networks, and knowledge based systems.