A New Approach on Particle Swarm Optimization for Multimodal Functions

Zahra Afsahi
Information & Communication Technology Management Systems & Quality V.P. MAPNA
Tehran, Iran
Electrical and Computer Engineering Department
Islamic Azad University
Qazvin, Iran
Afsahi_ai@yahoo.com

MohammadReza Meybodi
Computer Engineering and Information Technology Department
Amirkabir University
Tehran, Iran
mmeybodi@ce.aut.ac.ir

***Abstract*—This paper describes a technique that extends PSO to locate multiple optima on a multimodal functions. In this paper, we present a new algorithm based on clustering particles to identify niches. For that we employ the standard k-means clustering algorithm which can identify the number of clusters adaptively. In each niche we used artificial immune system algorithm to determine the true members of it. Experimental results show that the proposed algorithm can successfully locate all optimum solutions on a small set of test functions during all simulation runs.**

Keywords- Particle swarm optimization; Niche; Artificial Immune system and K-means

I. INTRODUCTION

The particle swarm optimization is first proposed by Kennedy and Eberhart [1]. It is a stochastic optimization technique inspired by the behaviour of a flock of birds. This evolutionary computation technique has been shown to effectively solve unimodal optimization problems. PSO are however not well equipped to locating multiple optimal solutions, because the design of this algorithm is usually targeted to the goal of finding a single optimal solution for a given problem. In multimodal functions, PSO pick just one of the optimal solutions or it could even be misled by the presence of more than a single optimum and fail to converge.

For optimizing these function, PSO have been modified introducing the concept of niching when niching applied to PSO, allows it to divide the space in different areas and search them in parallel. In this paper we introduce a niching technique to the PSO. Moreover, we will use the k-means clustering algorithm to identify niches in the population. After that, we propose artificial immune system to identify the members of each niche. In section 2, we present an overview of PSO algorithm. Section 3 gives a brief overview of existing niching techniques in the fields of PSO optimizers. Section 4 is dedicated to a detailed introduction of our new approach to niching PSO. In section 5, we show the results obtained by the algorithm from the experiments set up to compare its performance to those of other PSO niching algorithm.

978-1-4244-4754-1/09/$25.00 ©2009 IEEE

II. ORIGINAL PSO

The original PSO was inspired by the social behaviour of birds flocking or fish schooling. This algorithm consists of a swarm of particles flying through the search space [1]. Each individual i in the swarm contains parameters for position X_i, velocity V_i, and personal best position y_i, where $X_i \in R^n$, $V_i \in R^n$ and $y_i \in R^n$ while n is the dimension of the search space. The position of each particle represents a potential solution to the optimization. The personal best position associated with a particle i is the best position that particle has visited thus far, *i.e.* a position that yielded the highest fitness value for that particle. If f denotes the objective function, then the personal best of a particle at a time step t is updated as:

$$y_i(t+1)=\begin{cases} y_i(t) & \text{if} \quad f(X_i(t+1)) \geq f(y_i(t)) \\ X_i(t+1) & \text{if} \quad f(X_i(t+1) < f(y_i(t)) \end{cases} \tag{1}$$

Depending on the social network structure of the swarm *l*best and *g*best experience of particles, exchange among them. For the *g*best model, the best particle is determined from the entire swarm. If the position of the best particle is denoted by the vector, then:

$$\hat{y}(t) \in \{y_0, y_1, \ldots, y_s | f(\hat{y}_j(t+1))\} = \min\{f(y_0(t)), f(y_1(t), \ldots, f(y_s(t))\} \tag{2}$$

For the *l*best model, a swarm is divided into overlapping neighborhoods of particles. For each neighborhood N_j, a best particle is determined with position $\hat{y}_j$. The best particle is referred to as the neighborhood best particle, defined as:

$$\{\hat{y}_j(t+1) \in N_j | f(\hat{y}_j(t+1))\} = \min\{f(y_i)\} \forall y_i \in N_j \tag{3}$$

In [2] Kennedy and Mendes recommended the Van-Neumann architecture, in which a particle's neighbors are above, below and on each side on a two dimensional lattice, to be the most promising one. For each iteration of a gbest PSO algorithm, the j_{th} dimension of particle i_{th} 's velocity vector, and its position vector, X_i is updated as follows:

$$V_{i,j}(t+1) = w * V_{i,j}(t) + C_1 r_{1,j}(t)(y_{i,j}(t) - x_{i,j}(t)) + C_2 r_{2,j}(t)(\hat{y}_{i,j}(t) - x_{i,j}(t)) \quad (4)$$
$$X_i(t+1) = X_i(t) + V_i(t+1)$$

The PSO algorithm performs repeated applications of the update equations until a specified number of iterations have been exceeded, or until a user-defined stopping criterion has been reached.

III. NICHING TECHNIQUES IN PSO

Niching techniques maintain multiple solutions in multimodal domains, in contrast to swarm intelligence optimization technique such as PSO that have been designed to only locate single solutions. Applying niching techniques in PSO algorithm extends the inherent unimodal nature of this algorithm by growing multiple swarms from an initial particle population. The initial particle swarm is split into smaller swarms as niches are detected. Upon termination of the algorithm, each sub swarm represents one of the potential solutions to the problems.

Thus as pointed out by Engelbrecht et al. the standard PSO must be modified to allow the efficient location of multiple solutions [9]. Notwithstanding the differences between the approaches we will discuss for the particle swarm, with respect to the evolutionary ones, we will still refer to them as niching technique.

A. Objectie function stretching

The stretching technique adopts the landscape of an optimization problem's fitness function to remove local minimum [8, 10]. This method was applied as a sequential niching technique to the PSO algorithm, by Parsopoulos and Vahatis [12]. When a local solution is detected during the evolutionary learning process, the stretching operator is applied to remove the detected solution from the fitness landscape. Subsequent iteration of the PSO algorithm can then focus on locating solutions in other parts of the search space, assured that the detected local optima will not again lead to premature convergence. "Application of the stretching functions" means that the fitness calculation of removing particles is adapted. In this way successive iterations of the search space, lead to the identification of the other solutions [11]. The effectiveness of the stretching transformation is not uniform on every function. In fact, in some cases it can introduce false minima, which render this method unreliable. Van den Bergh pointed that, keeping the PSO from discovering all possible solutions [25].

B. Niche PSO

In 2002, the *n*best PSO introduced as a parallel niching. This method is aimed at locating multiple solutions to multimodal problems through the use of multiple, independent sub swarms [13]. A new approach, Niche PSO, was proposed which use sub swarms to locate multiple solutions to multimodal function optimization problems [14]. This algorithm starts by uniformly distributing particle the search space of an optimization problem. The in of particles is referred to as the main swarm. traverse the search space, they invariably mo positions that have attractive fitness. A potential identified by monitoring the change in a particle's a number of training iteration. When such a identified a new sub swarm is created by removi main swarm the particle that detected the potential creating a sub swarm from it. The algorithm is c have converged when sub swarms no longer imp solutions that represent.

C. Species-based PSO

In this algorithm, a procedure determines the s which identify the niches in the population [16 species seeds have been identified, all the other assigned to the niche formed by the closest se neighborhood structure is adapted to reflect the niches. In fact, each species seed would serve as all the other particles in its niche. This method suitable on dynamics environments [6]. Howe requires a radius parameter σ to determine the the niches.

D. Niching vector-based PSO

In this method, niches are identified by o sequential way [22]. This procedure starts from th and then repeating for the particles outside its ni the same authors developed the parallel vector-b more efficient version, based on the same principl followed a parallel approach. In this method, niches are identified and maintained in parall introduction of a special procedure which can niches when they become closer than a specified The vector-base approach has the appealing identifying niches by using operations on vecto inherent to the particle swarm algorithm. Thus good way to build a particles swarm for multim optimization.

E. Adaptive niching PSO

In this method the main parameters of nichi adaptively [4].

The first step, the average distance between each its closest neighbor as:

$$r = \frac{\sum_{i}^{N} \min_{j \neq i} \|x_i - x_j\|}{N}$$

This parameter determines the formation of niche

IV. OUR APPROACH

In this paper we proposed the clustering method for PSO to identify multiple global and l a multimodal search space. The basic idea is inherent unimdal nature of the standard PSO growing multiple swarms from an initial partic The initial population (particle swarm) is split swarms which called niche, to preserve diversit niches in population we employ the standard k-means algorithm, which is probably the best known partition clustering algorithm. Passaro and Starita presented *KPSO* [3], which employed the standard k-means clustering algorithm and improved with a mechanism to adaptively identify the number of clusters, which is similar to ANPSO [4]. The main problem in these algorithms is, determining the true numbers of each niche. In our approach, this problem is solved by using *artificial immune system* (AIS). In this section we introduce our approach.

A. k-means

K-mean is a very simple algorithm [24] and Kennedy used this algorithm to cluster particles in a swarm in his research on stereotyping [7]. Particles cluster according to their pbest, the previous best position. After that the neighborhood topology modify so that each particle can communicate only with particles in the swarm cluster. Therefore, the main swarm turns in a collection of sub swarms which tend to explore different regions of search space. Since, we want to perform a local search in each sub swarm, each of them use a gbest topology, with all connected particles which are the members of that sub swarms. The initial swarm of particles is randomly generated. Then k-means algorithm is performed and repeated at regular intervals. Particles in different sub swarm (cluster) at early stage of the simulation can end up in the same local optimum, it means that those clusters are similar and merge by creating a new sub swarm. Social information of them share between the other clusters, or in contrast when some particles of a cluster fly towards a different optimum, that cluster can be split into two clusters. In our method clustering algorithm is applied only every C iteration. This idea maintains the clusters over time by blocking communication between particles in different clusters and allows particles to follow their natural dynamics for some steps. Moreover the computational overhead becomes smaller than the other niching PSO technique.

B. Estimating the number of clusters

One of the major shortcomings of k-means algorithm is the number of cluster k [17]. In our approach we optimize Bayesian Information criterion (BIC) [15] to estimate the best choice of k. we can calculate its BIC value with:

$$BIC(D) = \ell(D) - \frac{\rho}{2}.\log N \tag{6}$$

This is also known as the Schwarz criterion. Where D is a given clustering and $\ell(D)$ is the log-likelihood point, and ρ the number of parameters and is given by the sum of *k-1* probabilities $d.k$ centroid coordinates and the k variance estimate σ_j, thus we have:

$$\rho = (k-1) + d.k + k$$

$$\sigma_j^2 = \frac{1}{N_j - 1}\sum_{p\in C_j}\left\|p - m_j\right\|^2 \tag{7}$$

where σ_j is the variance of the cluster C_j, with m_j the center of the j_{th} cluster, N_j its size. The formula for the log-likelihood can be calculated considering that we are assuming components densities in the form of Spherical Gaussians.

$$\rho(x|m_j,\sigma_j) = \frac{1}{\sigma_j^d\sqrt{2\pi}}e^{-(1/2\sigma_j^2)\left\|x-m_j\right\|^2} \tag{8}$$

And with a few mathematical transformations, the log-likelihood of the clustering $\ell(D)$ can be written as:

$$\ell(D) = \sum_{i=1}^{N}\log\rho(x_i|D) = \sum_{j=1}^{k}\ell(D_j) - N.\log N \tag{9}$$

where, $\ell(D_j)$ is the log-likelihood for each cluster D_j:

$$\ell(D_j) = -\frac{N_j}{2}.\log 2\pi - \frac{N_j.d}{2}.\log\sigma_j^2 - \frac{N_j - 1}{2} + N_j.\log N_j \tag{10}$$

At each clustering application, k-means algorithm is thus repeated with varying value for k that usually in the range from 2 to $N/2$, then the clustering with highest BIC is chosen. The other additional parameter is r_k, the number of times for a single clustering application. Because of the depended of k-mean algorithm on the initial assignment the seeds, repeat this algorithm a few times to obtain good results. Thus we set $r_k = 10$.

C. Artificial immune system

After performing k-means algorithm and defining niches in population, we have to remove some particles from the overcrowded clusters. We use the concept of artificial immune systems. History and progression of research in the field of AIS, shows that works in this area has 3 major roots, and consequently distinct philosophies: idiotypic network theory, negative selection and danger theory [18, 19, 20, 21].

In this paper we use the theory of negative selection as a cutting procedure to avoid the formation of overcrowded niches. The negative selection algorithm (NSA) is the most widely used techniques in AISs [5]. The NSA is based on the principles of self-nonself discrimination. The algorithm was inspired by the thymic negative selection process that intrinsic to natural immune systems, consisting of screening and deleting self-reactive T-cells, *i.e.* these T-cells that recognize self-cells.

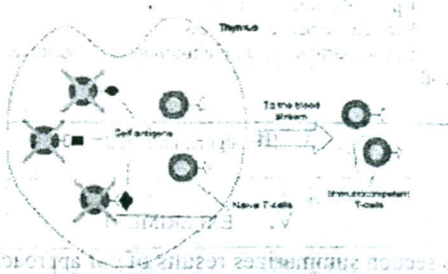

Figure1. Simplified view of the thymic negative selection

Resumption of this paper and in our approach negative selection responsible for eliminating all particles (T-cells

Labels) whose fitness is smaller than the average fitness of the niche which they belong it (receptors recognize and bind with self antigens represented in thymus). This concept is illustrated in Fig. 1.

In this problem the antigens are position vectors $X(t)$. For indicating self and non-self antigen in this problem we need to calculate the average fitness of all the particles in each niche. These procedures continue by the following as in (8):

$$\text{If } f(x_{i,j}) \le f_{avrage} \Rightarrow x_i \quad \text{non-self} \qquad (8)$$

$$\text{If } f(x_{i,j}) > f_{avrage} \Rightarrow x_i \quad \text{self}$$

Where, $x_{i,j}$ is the i_{th} particle in j_{th} niche. In a niche defines as an antigen remove from that niche and will add to a new niche, which known as $N_{explore}$. At the end, all the members of $N_{explore}$ reinitialize randomly. Using this procedure has two benefit, 1)avoiding the formation of overcrowded niches, which would end up in a waste of computational power, as too many particles would explore the search space around a single optimum, 2)reinitializing randomly all the members of $N_{explore}$ cause to explore new areas. After performing AIS the members of each niche identify. Now we have k niches whose particles are fully connected, realizing a *g*best topology in each niche. The particles in $N_{explore}$ organize in a von Neumann lattice neighborhood, as it is shown in Fig. 2.

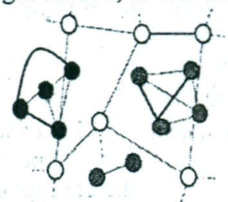

Figure 2. Van-Neumann architecture for the particles in $N_{explore}$

Fig. 3 reports the pseudo code for main algorithm and AIS.

```
Procedure Main
  Initialize particles with random positions and velocities.
    Set particles pbests to their current positions.
    Calculate particles fitness.
    for step t=0 → T-1 do
        if t mod c=0 then
           Cluster Particles with K-means algorithm
           Execute the procedure Artificial Immune System
        end
        Update particles velocities.
        Update Particles positions.
        Recalculate particles fitness.
        Update particles and neighborhood best positions.
    end
end.
```

Figure 3. Main algorithm pseudo code

V. EXPERIMENT

This section summarizes results of our approach to finding optimum of multimodal functions. We compare our algorithm with SPSO, ANPSO and KPSO algorithm. The study was conducted on the same set of benchmark functions which are reported in table1. In order to show how our algo effectively identify niches surrounding the opti function, we show in Fig. 4 several significant iterati algorithm on the Branin ROC function. Fig. 4 (a) how at the beginning of the run the particles of swarm are randomly distributed on the search space and (c) they naturally start to split in different niche shown in figure 4 (e), because of running AIS and some particles from overcrowded niches, at iteratio each niche we have just 2 or 3 particles. Hence, ju shown in table2 our method is the fast one amon algorithm. It is interesting to note that, when the fist algorithm, k-means, performs it defines 5 niches actually 3 of them correspond to the global optima, other is a spurious one. At the end at iteration algorithm will eventually converge to exactly 3 niche clusters become empty.

In table2 we report the results obtained c benchmark functions with KPSO, SPSO and AP execution was repeated 30 times. Value of C, the r steps between two clustering applications, is parameter that we have to define. With the higher v performance of the algorithm did not vary significant

VI. RESULT

In this paper we introduced a new approach niching technique in PSO algorithm, which allow it optima in multimodal functions. Then we described our approach used k-means algorithm for defining n also AIS algorithm as a cutting procedure. The algorithm maintains essentially the structure of stand Moreover, in our approach niches can define a Result showed comparable and higher performan entire test we conducted.

REFERENCES

[1] J. Kennedy and R. C. Eberhart, "Particle swarm optim Proceedings of IEEE International Conference Networks(ICNN '95), vol. 4, pp. 1942-1948, IEEE Ser Perth, Western Australia, November-December 1995.

[2] J. Kennedy and R. Mendes, "Population structure and pa performance," in Proceedings of IEEE Congress on Computation (CEC '02), vol. 2, pp. 1671–1676, Honol USA, May 2002.

[3] A. Passaro and A.Starita," Particle swarm optimization for functions: a clustering approach", Journal of the Artificial E Applications, vol. 2008, no.482032, February 2008S. W "Niching method for genetic algorithms", PHD thesis, U Illinois at Urbana-Champaign, Champaign, Ill, USA, 1995.

[4] S. Bird and X. Li, "Adaptively choosing niching parameter in Proceedings of the 8th Annual Conference on Evolutionary Computation (GECCO '06), vol. 1, pp. 3-10, Seattle, Wash, USA, July 2006.

[5] Hunt, J. E. & Cooke, D. E. (1996), "Learning Using Immune System", Journal of Network and Computer Appl pp. 189-212.

[6] D. Parrott and X. Li, "A particle swarm model for track peaks in a dynamic environment using speciation,"in Pr IEEE Congress on Evolutionary Computation (CEC '04), 103, IEEE Service Center, Portland, Ore, USA, June 2004.

[7] J. Kennedy, "Stereotyping: improving particle swarm perf cluster analysis", in Proceedings of IEEE Congress on

Computation (CEC'00), vol. 2, pp. 1507-1512, La Jolla, Calif, USA, July 2000.

[8] S. W. Mahfoud, "Niching method for genetic algorithms", PHD thesis, University of Illinois at Urbana-Champaign, Champaign, Ill, USA, 1995.

[9] A. P. Engelbrecht, B. S. Masiye, and G. Pampara, "Niching ability of basic particle swarm optimization algorithms", in Proceedings of the IEEE Swarm Intelligence Symposium (SIS '05), pp. 397-400, Pasadena, Calif, USA, June 2005.

[10] K. E. Parsopoulos, V. P. Plagianakos, G. D. Magoulas, and M. N. Vrahatis, "Stretching technique for obtaining global minimizes through particle swarm optimization", in Proceedings of the Particle Swarm Optimization Workshop, pp. 22-29, Indianapolis, Ind, USA, April 2001.

[11] K. E. Parsopoulos, V. P. Plagianakos, G. D. Magoulas, and M.N. Vrahatis, "Improving particle swarmoptimizer by function "stretching"," in Nonconvex Optimization and Applications, vol. 54, pp. 445-457, chapter 3, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[12] K. E. Parsopoulos and M. N. Vrahatis, "Modification of the particle swarm optimizer for locating all the global minima", In Artificial Neural Network and Genetic Algorithms, Computer Science, pp. 324-327, Springer, Wien, Austria, 2001.

[13] R. Brits, A. P. Engelbercht, and F. van den Bergh, "Solving systems of unconstrained equations using particle swarm optimization", in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '02), vol. 3, pp. 100-105, Hammamet, Tunisia, October 2002.

[14] R. Brits, A. P. Engelbercht, and F. van den Bergh, "A niching particle swarm optimization", in Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAI '02), vol. 2, pp. 692-696, Singapore, November 2002.

[15] G. Schwarz, "Estimating the dimension of a model," Annals of Statistics, vol. 6, no. 2, pp. 461–464, 1978.

[16] X. Li, "Adaptively choosing neighborhood bests using species in a particle swarm for multimodal function optimization", in Proceedings of the Conference on Genetic and Evolutionary Computation(GECCO '04), K. Deb, R. Poli, W. Banzhaf, et al., Eds., vol. 3102 of Lecture Notes in Computer Science, pp. 105-116, Springer, Seattle, Wash, USA, June 2004.

[17] D. Pelleg and A. Moore, "X-means: extending k-means with efficient estimation of the number of clusters", in Proceedings of the 17th International Conference on Machine Learning (ICML '00), pp. 727-734, Morgan Kaufmann, Stanford, Calif, USA, June-July 2000.

[18] Dasgupta, D., (1997), "Artificial Neural Networks and Artificial Immune Systems: Similarities and Differences", Proc. of the IEEE SMC, 1, pp. 873-878.

[19] McCoy, D. F. & Devaralan, V. (1997), "Artificial Immune Systems and Aerial Image Segmentation", Proceedings of the SMC'97, pp. 867-872.

[20] Dasgupta, D., (1999a), Artificial Immune Systems and Their Applications, Ed., Springer-Verlag.

[21] Dasgupta, D., (1999b), "Immunity-Based Intrusion Detection System: A General Framework", In Proc. of the 22nd NISSC.

[22] I. L. Schoeman and A. P. Engelbrecht, "Using vector operations to identify niches for particle swarm optimization," in Proceedings of the IEEE International Conference on Cybernetics and Intelligent Systems (CCIS '04), vol. 1, pp. 361-366, Singapore, December 2004.

[23] I. L. Schoeman and A. P. Engelbrecht, "A parallel vector-based particle swarm optimizer," in Proceedings of the 7th International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA '05), Coimbra, Portugal, March 2005.

[24] R. Xu and D. Wunsch II, "Survey of clustering algorithms", IEEE Transactions on Neural Networks, vol. 3, no. 16, pp. 645-678, 2005.

[25] F. V. Bergh, "An analysis of particle swarms optimizers", PhD thesis, Department of computer science, university of Pretoria, Pretoria, South Africa, 2002.

Table 1. list of test functions

Name	F	Global Optimums	Search Range
Branin ROC	$f(x,y)=(y-\frac{5.1x^2}{4\pi^2}+\frac{5x}{\pi}-6)^2+10(1-\frac{1}{8\pi})\cos(x)+10$	3	$-5\le x\le 10$ $0\le y\le 15$
Six-Hump camel back	$f(x,y)=-4\left[\left(4-2.1x^2+\frac{x^4}{3}\right)x^2+xy+(-4+4y^2)y^2\right]$	2	$-1.9\le x\le 1.9$ $-1.1\le y\le 1.1$
Himmelblau	$f(x,y)=200-(x^2+y-11)^2-(x+y^2-7)^2$	4	$-6\le x, y\le 6$

Table 2. Number of evaluations required to find all global optima

Function	Numbers of particles	SPSO	ANPSO	k-PSO	k-PSO with AIS
f_1	30	3169±692	3323 ± 5220	2084±440	**1854±471**
f_2	30	2872±827	2798±857	1124±216	**986±584**
f_3	30	4096±2018	16308±13157	2259±538	**1789 ± 603**

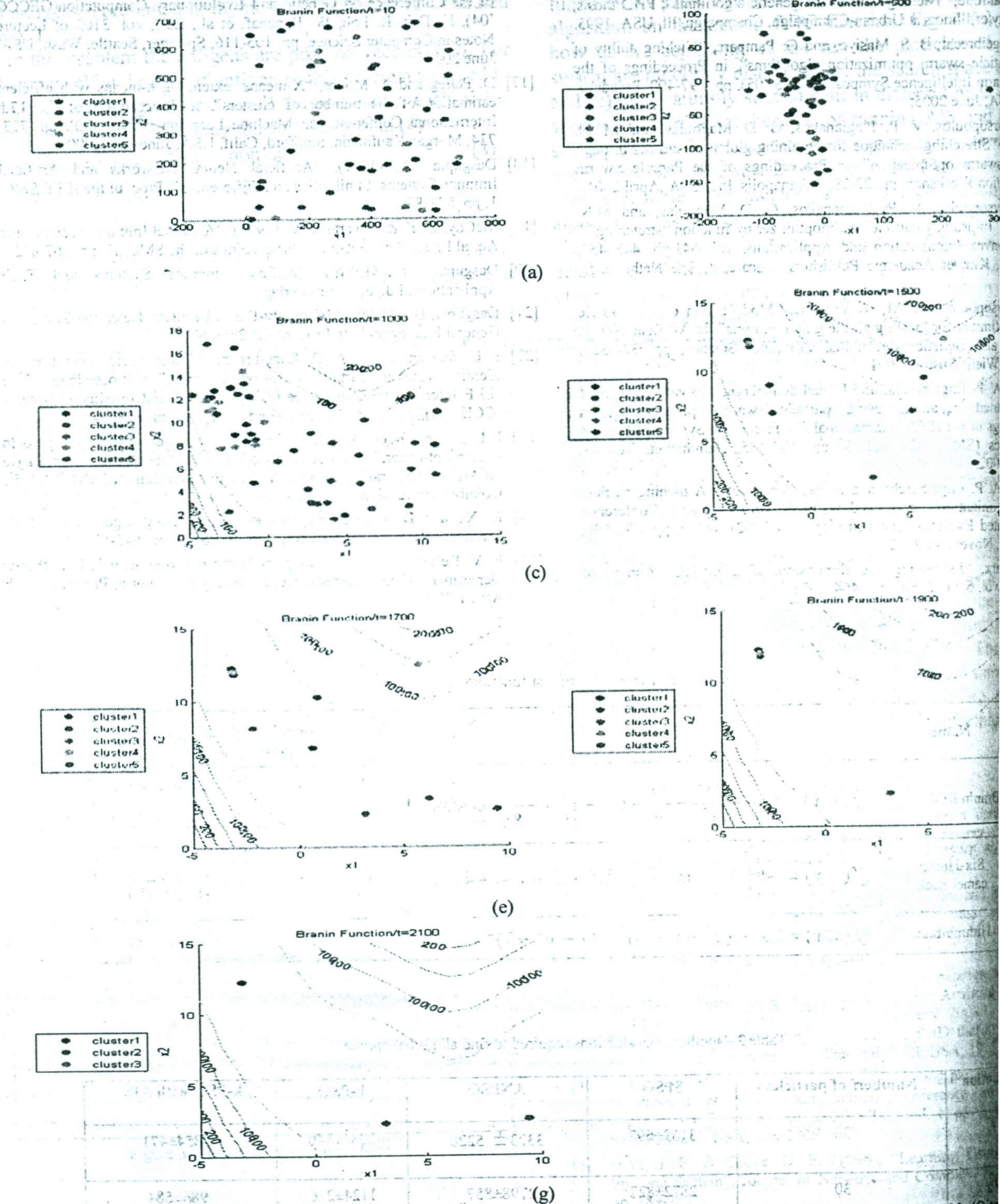

Figure 4. Significant iterations of our approach: (a) iteration 10, (b) iteration 500, (c) iteration 1000, (d) iteration 1500, (e)iteration 1700, (f)it (g)iteration 2100