A Chaotic Charged System Search Approach for Data Clustering

Y. Kumar and G. Sahoo
Department of Computer Science and Engineering, Birla Institute of Technology, Mesra, Ranchi, Jharkhand, India
E-mail: yugalkumar.14@gmail.com, gsaho@bitmesra.ac.in

Keywords: chaos theory, clustering, Coulomb law, charge particles, Gauss Law, Newton Law

Received: February 19, 2014

Data clustering is a key technique in the field of data mining, pattern recognition, bioinformatics and machine learning which concerns the organization and unexplored relationship between the huge amounts of data. It can analyse the data without knowing the size and distribution. Thus, this paper presents a new approach based on the charged system search (CSS) with chaotic map for partition clustering. The aim of this method is to achieve the global optimum solution by minimizing an objective function. The chaotic charge system search algorithm (CCSAA) utilizes the concept of CSS algorithm and chaos theory to obtain the desired results. The quality of the proposed algorithm is evaluated on seven datasets and then compared with other well-known algorithms in data mining domain. From the simulations results, it is observed that the proposed algorithm delivers more efficient and effective results than the other methods being compared.

Povzetek: Razvita je nova metoda gručenja s pomočjo kaotičnega učenja.

1 Introduction

The aim of clustering is to discover a subset of items in a given dataset which are more alike than others using similarity measures. The various authors have applied different criteria or similarity measures to identify the items in the clusters. But, the Euclidean distances is widely accepted similarity measure for clustering problems. Cluster analysis has proven its significance in many areas such as pattern recognition [47, 49], image processing [41], process monitoring [45], machine learning [1], quantitative structure activity relationship [8], document retrieval [17], bioinformatics [15], image segmentation [34] and many more. Due to wide area of clustering in diverse domains, a large number of algorithms have been developed by various researchers and applied successfully for clustering task. Generally, clustering algorithms can be classified into two groups- Hierarchical clustering algorithms and Partition based clustering algorithms [4, 5, 26, 30]. In hierarchical algorithms, a tree structure of data is formed by merging or splitting data based on some similarity criterion. In partition based algorithms, clustering is achieved by relocating data between clusters using some clustering criterion i.e. Euclidean distance. From the literature, it has been found that partition based algorithms are more efficient and popular than hierarchical algorithms [18]. The most popular and commonly used partition based algorithm is K-means algorithm [29]. It is easy, fast, and simple to implement. In addition to it, also have one more characteristic that is linear time complexity [10, 19]. In K-means algorithm, a dataset is divided into k number of predefined clusters and the clustering objective is to minimize the intra-cluster distance [18]. Nonetheless, this algorithm has some limitations such as the results of k-means algorithm is highly dependent on the initial cluster centers and also get stuck in local minima [20, 38]. Thus, in order to overcome the pitfalls of the K-means algorithm, several heuristic algorithms have been developed. K-harmonic mean algorithm was also proposed for clustering task instead of K-Means [48]. A simulated annealing (SA) based approach has been developed in [39]. A tabu search (TS) based method was introduced in [2, 42]. A genetic algorithm (GA) based methods was presented in [6, 16, 27, 31, 32]. A comparison has been performed to investigate the computational performance of K-Means, SA, TS and GA for data clustering [3]. Fathian et al., in 2007 have developed a clustering algorithm based on honey-bee mating optimization (HBMO) [9]. An ant colony optimization (ACO) based approach for clustering is proposed in [40]. Particle swarm optimization (PSO) is applied for clustering in [46]. Whereas Hatamlou et al. employed a big bang-big crunch algorithm for data clustering in [12]. Karaboga and Ozturk presented a novel clustering approach based on artificial bee colony (ABC) algorithm in [21]. A data clustering based on gravitational search algorithm was presented in [13, 14]. For the effective clustering, a teacher learning based optimization method is applied in [35, 36]. But every algorithm has some drawbacks, for example, K-Means algorithm sucks in local optima, convergence is highly dependent on initial positions in case of genetic algorithm, in ACO the solution vector has been affected as the number of iterations increased etc.

So,the aim of this research work is to investigate the capability of chaotic charged system search (CCSS) algorithm for data clustering. This algorithm is the combination of the chaotic map and charge system search algorithm. The chaotic maps are required for chaotic search. Several chaotic maps have been reported in the literature such as the logistic map, tent map and so on. These maps are used to produce a sequence of numbers

to substitute the random parameters r_i, $rand$, $rand_1$ and $rand_2$ of the CSS algorithm. Thus, in this paper a logistic map is considered for chaotic search with charge system search algorithm (CSS) and called it chaotic charge system search algorithm (CCSSA). On the other side, the CSS algorithm is the latest meta-heuristic optimization technique developed in [22]. This technique is built on three principles: Coulomb law, Gauss law and Newton second law of motion. Every meta-heuristic algorithm contains two unique features i.e. exploration and exploitation. The exploration is referred to generate promising searching space while the exploitation can be defined as determination of the most promising solution set. Thus, in CCSS, the exploration process is carried out using Coulomb and Gauss laws while Newton second law of motion is applied to perform exploitation process. The performance of the proposed algorithm has been tested on two artificial datasets and several real datasets from the UCI repository and compared with some existing algorithms in which the quality of the solution is improved using CCSS algorithm.

2 Background

2.1 Charge system search (CSS) algorithm

CSS is the latest meta-heuristic algorithm developed by the Kaveh et al. based on movement of charged particles in D-dimensional search space [22]. The position of the i^{th} charged particle in search space is described as $X_i = (X_{i,1}, X_{i,2}, X_{i,3}, \ldots\ldots\ldots X_{i,D})$. The velocity of the i^{th} charged particle is represented as $V_i = (V_{i,1}, V_{i,2}, V_{i,3}, \ldots\ldots V_{i,D})$. It is suggested that the position and velocities of the charged particle should be in the range of $[X_{min}, X_{max}]^D$ and $[V_{min}, V_{max}]^D$ respectively. Initially, it is assumed that the velocities of all charged particles are set to 0. The main steps of the CSS algorithm are as follows.

Begin

- Randomly initialize charge particle in D-dimensional search space using equation 1.

$$X_{i,j} = X_{min,i} + r_i * (X_{i,max} - X_{i,min}) \quad (1)$$

Here, $X_{i,j}$ represents the initial value of the i^{th} variable for the j^{th} CP; $X_{min,i}$ and $X_{i,max}$ are the minimum and the maximum values for the i^{th} variable; r_i is a random number in the interval [0,1] and the initial velocities of charged particles are zero.

$$V_k = 0, \qquad k = 1, 2, \ldots . K$$

While (The stopping criterion is not met)

- Evaluate fitness of each charged particle, compare the values of fitness function and sort them into increasing order.
- Store the positions of initial charged particles (C_k) into a variable, called CM.

For n = 1 to number of charged particles

- Compute the value of moving probability for each charged particle using the following equation.

$$p_{ik} = \begin{cases} 1 & \frac{fit(i) - fit(best)}{fit(j) - fit(i)} \\ 0 & otherwise \end{cases} \quad (2)$$

If, $\frac{fit(i)-fit(best)}{fit(j)-fit(i)} > rand\ V\ fit(j) > fit(i)$,

Here, fit (i) represents the fitness of i^{th} instance of dataset while fit (best) represents the best fitness value and fit (j) represents fitness value of j^{th} data instance of the dataset and rand is a random number in between [0, 1].

- Determine the value of actual electric force (F) using the equation 3.

$$F_j = q_j \sum_{i,i \neq j} \left(\frac{q_i}{a^3} * i_1 + \frac{q_i}{r_{ij}^2} * i_2 \right) * p_{ij} * (X_i - X_j),$$

$$\begin{cases} j = 1, 2, 3, \ldots . K \\ i_1 = 1, i_2 = 0 \ \leftrightarrow \ r_{ij} < a \\ i_1 = 0, i_2 = 1 \ \leftrightarrow \ r_{ij} \geq a \end{cases} \quad (3)$$

Here, q_i and qj represents the fitness of i^{th} and j^{th} CP, $r_{i,j}$ represents the separation distance between i^{th} and j^{th} CPs, i_1 and i_2 are the two variables whose values are either 0 or 1, 'a' represents the radius of CPs and it is assumed that each CPs has uniform volume charge density but changes in every iteration. The value of q_i, $r_{i,j}$ and 'a' are evaluated using the equation 12, 13 and 14.

End for

For d = 1 to number of dimensions of charged particles

- Update the positions and velocities of charged particles using equation (4) and (5) respectively.

$$X_{j,new} = rand_1 * Z_a * \frac{F_j}{m_j} * \Delta t^2 + rand_2 * Z_v * V_{j\,old} * \Delta t + X_{j\,old} \quad (4)$$

$$V_{j\,new} = \frac{X_{j\,new} - X_{j\,old}}{\Delta t} \quad (5)$$

where, $rand_1$ and $rand_2$ are the two random functions whose values lie in between [0, 1], Z_a and Z_v are the control parameters which control the influence of actual electric force and previous velocities, m_j is the mass of j^{th} CPs which is equal to the q_k and Δt represents the time step which is set to 1.

- If charged particles cross its boundary limit then correct its position using HS based method.

End for

- Determine the fitness function of new generated charged particles and compare it to the charged particles stored into CM.
- Exclude the worst charged particles from CM.

End while

- Obtain the desired results

End

In short span of time, the CSS algorithm is applied in various research domain which shows its potency [22, 23, 24, 25, 44, 28].

2.2 Chaos theory

Recently, chaos has fascinated researchers and academicians from all over the world from different fields of sciences like chaos control, synchronization, pattern recognition, optimization theory and many others. Chaotic optimization algorithm (COA) can be defined overall as the one which uses chaotic variables in place of random variables [37]. From the extended literature survey, it came to revelation that chaos has unpredictable irregular behavior in the long term.

Whereas, nature inspired algorithms viz. ABC [21], CSS [22], GSA [13], SA [39], GA [16] and so on, as their name indicates are motivated by the biological living systems, swarms characteristics and natural phenomena's where order (regular and self-organized) replaces the disorder (irregular and unorganized) [43]. But, the chaos and nature inspired optimization algorithms shares two basic characteristics like self-organization and evolution, so, amalgamation of these two may enhance the performance of the optimization algorithm by replacing the random number with the chaotic variable. In 1976, May brought the concept of the logistic map, which is popular one among the chaotic maps [33]. It appears in nonlinear dynamics of biological population evidencing its chaotic behavior. The logistic map is a well-known polynomial chaotic map which can be described as:

$$cm_{a+1} = \alpha \times cm_a(1 - cm_a) \quad (6)$$

Where cm_a is the a^{th} chaotic number and "a" denotes the current iteration number. Clearly, $cm_a \in (0, 1)$, it is simple to indicate that if $0 < \alpha \leq 4$ then the interval (0, 1) is mapped into itself i.e. $cm_0 \in (0, 1)$, then $cm_a \in (0, 1)$. Mathematically, it is already established that when α=4, then all the values generated by the chaotic map will lie in the range of 0 -1 except that $cm_0 \notin$ (0, 0.25, 0.50, 0.75, 1). In this paper, α=4 has been used to perform the experiment.

3 Proposed CCSS algorithm for clustering

This section describes the working of CCSS algorithm for clustering task. A chaotic (logistic) map is applied to tune the CSS parameters in three different ways to enhance the performance of the CSS algorithm as well as global search especially for clustering problems. These modifications are described as:

CCSSA-1: The quality of the clustering algorithms depends on the initial cluster centers. In CSS algorithm, the initial positions of charged particles (CPs) are determined using the equation 1 which contains a random parameter (r_i). The Parameter (r_i) of equation 1 is updated using the chaotic map (cm) during iterations and the modified equation is rewritten as:

$$C_k = X_{min,i} + cm_a * (X_{i,max} - X_{i,min}),$$
$$where\ i = 1, 2 \ldots n\ and\ k = 1,2 \ldots K \quad (7)$$

In the standard CSS algorithm, r_i presents a random number in between 0 and 1. In CCSSA-1, the r_i is replaced by the chaotic number (cm_i) whose values also lies in between 0 and 1,C_k represents the number of cluster centers, $X_{min,\ i}$ and $X_{max,\ i}$ represent the minimum and maximum value of the i^{th} attribute in the dataset respectively; K is the total number of cluster centers in the given dataset.

CCSSA-2: The selected chaotic map is introduced in equation 2 which is used to compute the value of the moving probability of particles towards the charged particle. The modified equation can be given as follows.

$$p_{ik} = \begin{cases} 1 & \frac{fit(i) - fit(best)}{fit(k) - fit(i)} \\ 0 & otherwise \end{cases} \quad (8)$$

$$\text{If, } \frac{fit(i) - fit(best)}{fit(k) - fit(i)} > cm_a\ V\ fit(k) > fit(i),$$

In the standard CSS algorithm, $rand_i$ is the random number between 0 and 1. In CCSSA-2, it is replaced by chaotic number cm_i which ranges in between 0 and 1.

CCSSA-3: The parameters $rand_1$ and $rand_2$ of the position update equation 4 is altered using selected chaotic map and the modified equation can be given as.

$$C_{k,new} = cm_{a,1} * Z_a * \frac{F_k}{m_k} * \Delta t^2 + cm_{a,2} * Z_v * V_{k\ old} * \Delta t + C_{k\ old} \quad (9)$$

In the standard CSS algorithm, $rand_1$ and $rand_2$ present the random number in between 0 and 1. The chaotic map $cm_{a,1}$ and $cm_{a,2}$ are used instead of $rand_1$ and $rand_2$.

3.1 Algorithm details

In this section, CCSS algorithm is explained to solve the clustering problem. The aim of this algorithm is to find the optimal cluster points to assign N numbers of items to K cluster centers in R^n. Euclidean distance is taken as the objective function for clustering problem and items are assigned to a cluster center with minimum Euclidean distance.

The algorithm starts with defining the initial positions and velocities of K number of charged particles (CPs). The initial positions of CPs are defined in random manner. Here, CPs are assumed to be a charged sphere of radius 'a' and its initial velocity is set to zero. Thus, the algorithm starts with randomly defined initial cluster centers and ended with optimal cluster centers. Consider Table 1 which illustrates a dataset used to explain the working of CCSS algorithm for clustering with N=10, n=4 and the number of cluster centers (K) is 3. To obtain the optimal cluster centers, the CPs uses resultant electric force (attracting force vector), mass and moving probability of particles and cluster centers. After the first iteration, the velocities of CPs are determined and locations of CPs are also moved. The objective function is calculated again using the new positions of CPs and also compared with the old CPs positions that are stored in the memory pool, called as charged memory (CM). The CM is now updated with the new positions of CPs and excludes the worst CPs from CM. As the algorithm grows, the positions of CPs are updated along with the

content in CM is also updated. This process progresses until the maximum iteration or no better position of CPs will be generated.

As described earlier, the algorithm starts with the identification of the initial positions and velocities of CPs in random fashion. Thus, the equation 7 is used to initialize the initial positions of randomly defined CPs (cluster centers) and the initial positions of CPs are given in Table 2. The initial velocities of these CPs are set to zero. The CPs are described as charged spheres that contain some mass. Thus, the mass of each CP is calculated using the following equation.

$$m_k = \frac{fit(k) - fit(worst)}{fit(best) - fit(worst)} \quad (10)$$

Where, fit (k) represents the fitness of k^{th} instance of dataset while fit (best) represents the best fitness value and fit (worst) represents worst fitness value of dataset. The mass of initial positioned CPs are 1.2576, 1.714 and 0.94176.

Table 1: Illustrate a dataset to explain the CCSS algorithm for clustering with N=10, n=4 and K=3.

N	*n*			
	1	*2*	3	4
1	5.1	3.5	1.4	0.2
2	4.9	3	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5	3.6	1.4	0.2
6	5.4	3.9	1.7	0.4
7	4.6	3.4	1.4	0.3
8	5	3.4	1.5	0.2
9	4.4	2.9	1.4	0.2
10	4.9	3.1	1.5	0.1

Table 2: shows the position of initial CPs using equation.

K	*i*			
	1	*2*	3	4
1	4.459	3.281	1.5889	0.12854
2	5.0672	3.1964	1.5394	0.14556
3	4.8364	2.9127	1.3916	0.1791

7.

Euclidean distance is used as objective function to find the closeness of particles to CPs and assigned the particles to CPs with minimum objective value. Table 3 provides the value of objective function of initial positioned CPs for our example dataset. Euclidean distance can be given as.

$$d_{i,k} = \sum_{k=1}^{K} \sum_{j=1}^{N} \sum_{i=1}^{n} \sqrt{\left\| X_{j,i} - C_{k,i} \right\|^2} \quad (11)$$

Table 3: Normalized value of objective function.

N	*K*		
	1	*2*	*3*
1	0.70682	0.34008	0.64416
2	0.56056	0.29821	0.11036
3	0.39144	0.44174	0.33163
4	0.25624	0.48175	0.32117
5	0.65969	0.43567	0.70688
6	1.1639	0.83447	1.1985
7	0.31484	0.55047	0.55501
8	0.56552	0.22469	0.52577
9	0.43529	0.74525	0.43714
10	0.48574	0.20218	0.23904

The information contained in the given string is used to arrange the items into different clusters which are given below.

2	3	3	1	2	2	1	2	1	2

From the above string, it is observed that the first, fifth, sixth, eighth and tenth particles belong to the cluster 2^{nd}; second and third and tenth particles belong to cluster 3^{rd} ; fourth, seventh and ninth particles belong to cluster 1^{st} . Hence at this step, the dataset is divided into three different clusters and store the values of positions of CPs in a new variable called charge memory (CM) which can be used to memorize the positions of CPs. Later on, these CPs positions will be used for comparisons with newly generated CPs positions and the best positions are included in the CM and excluded the worst positions from CM. Here, the size of CM is equal to the N/4. The main work of CM is to keep track of the number of good positions of CPs which are obtained during the execution of CSS algorithm and after the execution of algorithm; the optimal number of CPs positions (i.e. K number of CPs) are determined using minimized objective function values. The above discussion relates to the initialization of the proposed algorithm for clustering problem.

As the study of various meta-heuristic algorithms, it is found that every meta-heuristic algorithm contains two approaches i.e. exploration and exploitation in which one approach initiated local search while the other approach carried out the global search. The local search tends to the exploration of random search space such that the most promising solution space can be obtained while the global search refers to the exploitation of good solution vectors from the promising solution space. Hence in case of CCSS algorithm, the local search i.e. exploration is initiated using Coulomb and Gauss laws while the global search i.e. exploitation is performed by Newton second law of motion. The local search of CCSS algorithm starts by measuring the electric force ($E_{i,k}$) generated by CP. The electric force ($E_{i,k}$) generates at a point is either inside the CPs or outside CPs. So, this direction of force is described as moving probability ($P_{i,k}$) of CPs. While

the Coulomb and Gauss laws are applied to measure the total electric force generated on a CP , called it actual electric force (F_k). The moving probability $P_{i,k}$ for each CPs can be determined using the equation 8. The value of moving probability ($P_{i,k}$) is either 0 or 1 and it gives the information about the movement of CPs. Table 4 shows the moving probability ($P_{i,k}$) values for each particles to each cluster centers.

The Coulomb and Gauss laws are employed to determine the value of actual electric force (F_k) generated on CPs. The Coulomb law is used to calculate the force outside the CP and Gauss law is used to calculate the force inside the CP. The equation 3 is used to determine the actual electric force (F_k).

$$q_i = \frac{fit(i) - fit(worst)}{fit(best) - fit(worst)}, i = 1, 2, \dots N \quad (12)$$

$$r_{ik} = \frac{\|X_i - X_k\|}{\|(X_i + X_k)/2 - X_{best}\| + \in} \quad (13)$$

$$a = 0.10 * max\{x_{i,max} - x_{i,min} | \, i = 1, 2 \dots n\} \quad (14)$$

Table 4: Moving probability P_{ik} of each CPs with each items of dataset.

N	*K*		
	1	*2*	*3*
1	0	0	0
2	0	1	0
3	1	1	0
4	1	1	0
5	0	0	0
6	0	0	0
7	0	1	0
8	0	0	0
9	1	0	1
10	0	1	0

Table 5: values of magnitude of charge (q_i) of each CPs and separation distance ($r_{i,k}$).

N	q_i	*Separation Distance ($r_{i,k}$)*		
		K		
		1	*2*	*3*
1	1.8833	1.991	0.61468	2.4197
2	1.1833	0.2648	1.3881	1.7242
3	0.93333	0.80968	1.0094	1.352
4	1.2333	0.45338	1.4435	1.1904
5	1.8833	2.0875	0.59338	2.2831
6	4	3.1051	1.9508	3.3476
7	1.6167	1.3312	0.55961	1.3142
8	1.9333	1.8999	0.42761	2.3886
9	0.58333	4.1443	4.1107	3.9615
10	1.2	0.80915	0.78601	1.9357

Tables 5,6 and 7 provide the values of q_i, $r_{i,k}$, i_1, i_2, "a" and (X_i-C_k) variables which are used to calculate the actual electric force (F_k), applied on the CPs and the value of 'a' is 0.4459, 0.9526 2 and 1.4363 which is calculated using equation 14. Thus, the values of the actual electric force (F_k) generated on initial CPs (cluster centers) is 0.32832, 1.1605 and 0.20691.The values of electric force (F_k) are used with Newton second law of motion to determine the new position of CPs and velocities of CPs.

The Newton second law of motion is employed to get the new positions and velocities of CPs. This is referred to as exploitation of solution vectors from the random space search.

Table 6: values of I_1 and I_2.

N	*K (I1)*			*K (I2)*		
	1	*2*	*3*	*1*	*2*	*3*
1	1	0	1	0	1	0
2	0	0	1	1	1	0
3	0	0	0	1	1	1
4	0	1	0	1	0	1
5	1	0	1	0	1	0
6	1	1	1	0	0	0
7	0	0	0	1	1	1
8	1	0	1	0	1	0
9	1	1	1	0	0	0
10	0	0	1	1	1	0

Table 7: values of X_i-C_k for each cluster center K.

N	*K*		
	1	*2*	*3*
1	0.74245	0.25143	0.88023
2	0.042448	0.44857	0.18023
3	0.057552	0.54857	0.080228
4	0.057552	0.54857	0.080228
5	0.74245	0.25143	0.88023
6	1.9424	1.4514	2.0802
7	0.24245	0.24857	0.38023
8	0.64245	0.15143	0.78023
9	0.55755	1.0486	0.41977
10	0.14245	0.34857	0.28023

Thus, the new positions of CPs and velocities are obtained from equation 9 and 5 respectively. Z_a and Z_v act as the control parameters which are used to control the exploration and exploitation process of the algorithm. These parameters also affect the values of previous velocities and actual resultant force generated on a CP. These values may be either increased or decreased. Thus, Z_a is the control parameter belongs to the actual electric force F_k and controls the exploitation process of CSS algorithm. The large value of Z_a increases the

convergence speed of the algorithm while small value increases the computational time of the algorithm. Z_v is the control parameter for the exploration process and acts with the velocities of CPs. Here, it is noted that Z_a is the increased function parameter while Z_v is the decreased function parameter. Table 8 provides the new positions of CPs which are evaluated using CCSS algorithm. The new positions of CPs are mentioned in Table 8 and the values of control parameters Z_a and Z_v are determined using the equation 14. The new velocities ($V_{k,new}$) of each CPs is 0.21599, 0.46049 and 0.10449.

$$Z_a = (1 - iteration/iteration\ max),$$
$$Z_v = (1 + iteration/iteration\ max) \quad (14)$$

Table 8: New position of CPs.

K	i			
	1	2	3	4
1	4.513	3.335	1.6429	0.18253
2	5.1823	3.3115	1.6545	0.26068
3	4.8625	2.9388	1.4177	0.20523

Hence from the above discussion, the process of the algorithm can be categorized into three sections: Initialization, Search and Termination condition. The initialization section deals with the CCSS algorithm parameters; positions and velocities of the initial CPs; determine the value of objective function and rank them; store the positions of CPs into CM. In the search section, the new positions and velocities of CPs are determined using moving probability ($P_{i,k}$) and actual electric force (F_k). The value of objective function is evaluated using newly generated CPs, compared with previous CPs; rank them and store the best CPs in CM. The termination condition of the algorithm is either maximum number of iterations or repeated positions of CPs.

3.2 Pseudo code of CCSS algorithm for clustering

Step 1: Load the dataset and specify the number of initial cluster centers (K).

Step 2: * initialize the initial positions and velocities of charged particles (CPs) * /
For each charged particles k=1 to K
For each j=1 to m
Determine the value of initial position of CPs (C_k) using equation 7;
Calculate the value of mass for each C_k using equation 10;
End for
End for
V_k=0; * velocities of each CP*/
Iteration =0;

Step 3: Evaluate the value of objective function using eq. 11 and assign the items to the clusters with minimum objective function values.

Step 4: Store the positions of initial charged particles (C_k)
into a variable, called CM.

Step 5: while (the termination conditions are not met), do
\ * Calculate the value of moving probability $P_{i,k}$ for each charged particle C_k * /
For each charged particles k=1 to K
For each i=1 to n
Determine the value of fitness of each instance (q_{ik}) with each CP (C_k) using eq. 8;
End for
If (fit (q_{ik}) > fit (k))
$P_{ik} \rightarrow 1$;
Else
$P_{ik} \rightarrow 0$;
End if
End for

Step 6: \ * Determine the value of actual electric force (F_k)*/
Determine the value of mass for each instance q_i using eq.12;
Calculate the value of radius 'a' using eq. 14;
For each charged particles k=1 to K
Calculate the value of separation distance ($r_{i,k}$) using eq. 13;
If ($r_{i,k} < a$)
$i_1 \rightarrow 1$
Else
$i_2 \rightarrow 0$
End if
If ($r_{i,k} \geq a$)
$i_2 \rightarrow 1$
Else
$i_1 \rightarrow 0$
End if
End for
For each charged particles k=1 to K
Determine the value of actual electric force (F_k) using eq. 3;
End for

Step 7: Calculate the new positions and velocities of CPs using eq. 9 and 5 respectively;

Step 8: Compute the value of objective function using new positions of CPs;

Step 9: Compare the value of objective function of newly generated CPs to CPs reside in CM;

Step 10: Memorize the best solution achieved so far
Iteration= Iteration +1;
End while

Step 11: Output the best solution obtained.

4 Experimental results

This section describes the results of the CCSS algorithm for data clustering problem. To assess the performance of CCSS algorithm, it is applied on ten datasets. These datasets are ART1, ART2, iris, wine, CMC, glass, breast cancer wisconsin, Liver disease (LD), thyroid and vowel in which iris, wine, CMC, glass. Liver disease (LD), thyroid, vowel and breast cancer wisconsin datasets are real that are downloaded from the UCI repository while rest of two datasets are artificial i.e., ART1 and ART2. The characteristics of these datasets are discussed in Table 9. The proposed algorithm is implemented in

Matlab 2010a environment on a core i5 processor with 4 GB using window operating system. For every dataset, the algorithm is run 100 times individually to check the effectiveness of the proposed algorithm using randomly generated cluster centers. The parameter settings for CCSS algorithm are mentioned in Table 10. The sum of intra cluster distance and f-measure parameters are used to evaluate the quality of solutions for clustering algorithm. The sum of intra cluster distances can be defined as distances between the instances placed in a cluster to the corresponding cluster center. The results are measured in terms of best case, average cases, worst case solutions and standard deviation. The quality of clustering is directly related to the minimum of the sum of distances. The accuracy of clustering is measured using f-measure. To ensure the effectiveness and adaptability of CCSS algorithm in clustering domain, the investigational results of CCSS algorithm are compared with K-Means, GA, PSO, ACO and CSS algorithms which are given in Table 11.

4.1 Datasets

4.1.1 ART1

It is two dimensional artificial dataset, generated in matlab to authenticate the proposed algorithm. This dataset includes 300 instances with the two attributes and three classes. Classes in dataset are disseminated using μ and λ where μ is the mean vector and λ is the variance matrix. The data has generated using μ1 = [3, 1], μ2 = [0, 3], μ3 = [1.5, 2.5] and λ1 = [0.3, 0.5], λ2 = [0.7, 0.4], λ3 = [0.4, 0.6]. The Figure 1(a) depicts the distribution of data into ART1 and figure 1(b) shows the clustering of same data using CSS method.

4.1.2 ART2

It is three dimensional artificial data which includes 300 instances with three attributes and three classes. The data has generated using μ1 = [10, 25, 12], μ2 = [11, 20, 15], μ3 = [14, 15, 18] and λ1 = [3.4, -0.5, -1.5], λ2 = [-0.5, 3.2, 0.8], λ3 = [-1.5, 0.1, 1.8]. The Figure 2(a) represents the distribution of data in ART2 dataset and Figure 2(b) shows the clustering of same data using CCSS method.

4.1.3 Iris Dataset

Iris dataset contains three variety of the iris flower which is setosa, versicolour and virginica. The dataset contains 150 instances with three classes and four attributes in which each class contains of 50 instances. The attributes of iris dataset are sepal length, sepal width, petal length, and petal width.

4.1.4 Wine Dataset

It contains the chemical analysis of wine in the same region of Italy using three different cultivators. This dataset contains 178 instances with thirteen attributes and three classes. The attributes of dataset are alcohol, malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines and proline.

4.1.5 Glass

This dataset consists of six different types of glass information. The dataset contains 214 instances and 7 classes. It contains nine attributes which are refractive index, sodium, magnesium, aluminum, silicon, potassium, calcium, barium, and iron.

4.1.6 Breast Cancer Wisconsin

This dataset characterizes the behavior of cell nuclei present in the image of breast mass. It contains 683 instances with 2 classes i.e. malignant and benign and 9 attributes. The attributes are clump thickness, cell size uniformity, cell shape uniformity, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli, and mitoses. Malignant class consists of 444 instances while benign consists of 239 instances.

4.1.7 Contraceptive Method Choice

It is a subset of National Indonesia Contraceptive Prevalence Survey data that had been performed in 1987. This dataset contains information about married women who were either pregnant (but did not know about pregnancy) or not pregnant. It contains 1473 instances and three classes i.e., no use, long term method and short term method. Each class contains 629, 334 and 510 instances respectively. It has nine attributes which are Age, Wife's education, Husband's education, Number of children ever born, Wife's religion, Wife's now working, Husband's occupation, Standard-of-living index and Media exposure.

4.1.8 Thyroid

This dataset contains the information about the thyroid diseases and classifies the patient into three classes-normal, hypothyroidism and hyperthyroidism. The dataset consists of 215 instances with five features. The features are the medical tests which have been used to categorize the patients. The features are Tresin, Thyroxin, Triiodothyronine, Thyroidstimulating and TSH value.

4.1.9 Liver Disorder

This dataset is collected by BUPA medical research company. It consists of 345 instances with six features and two classes. The features of the LD dataset are mcv,alkphos, sgpt, sgot, gammagt and drinks.

4.1.10 Vowel

This dataset consists of 871data instances of Indian Telugu vowel sounds with three features which correspond to the first, second, and third vowel frequencies and six classes.

4.2 Performance measures

4.2.1 Sum of intra cluster distances

It is sum of distances between the data instances present in one cluster to its corresponding cluster center. Minimum the sum of intra cluster distance indicates the better the quality of the solution. The results are measured in terms of best, average and worst solutions.

Table 9: Characteristics of datasets.

Dataset	*Classes*	*Features*	*Total instances*	*Instance in each classes*
ART 1	3	2	300	(100, 100, 100)
ART 2	3	3	300	(100, 100, 100)
Iris	3	4	150	(50, 50, 50)
Glass	6	9	214	(70,17, 76, 13, 9, 29)
LD	2	6	345	(145, 200)
Thyroid	3	3	215	(150, 30, 35)
Cancer	2	9	683	(444, 239)
CMC	3	9	1473	(629,334, 510)
Vowel	6	3	871	(72, 89, 172, 151, 207, 180)
Wine	3	13	178	(59, 71, 48)

4.2.2 Standard Deviation

This parameter gives the information about the dispersion of data present in the cluster from its cluster center. The minimum value of standard deviation indicates that the data are close to its center while a large value indicates the data are far from its center points.

4.2.3 F-Measure

F-measure is calculated by the recall and precision of an information retrieval system [7, 11]. It is weighted harmonic mean of recall and precision. To determine the value of F –measure, every cluster describes a result of the query while every class describes as a set of credentials for the query. Thus, if each cluster j consists a set of n_j data instances as a result of a query and each class i consists of a set of n_i data instances require for a query then n_{ij} gives the number of instances of class i within cluster j. The recall and precision, for each cluster j and class i is defined as

$$Recall\ (r(i,j)) = \frac{n_{i,j}}{n_i} \text{ and}$$

$$Precision\ (p(i,j)) = \frac{n_{i,j}}{n_j} \qquad (13)$$

The value of F-measure (F (i, j)) is computed as

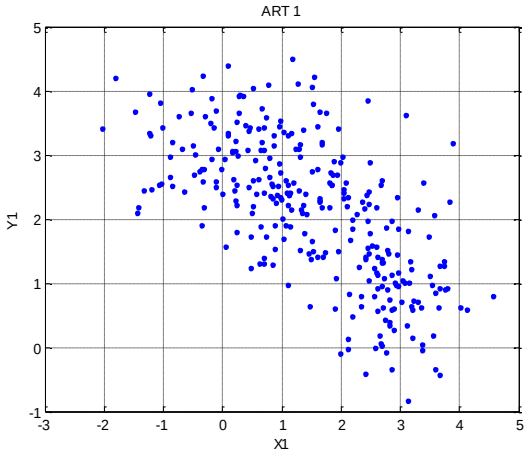

Figure 1(a): Distribution of data in ART1 dataset.

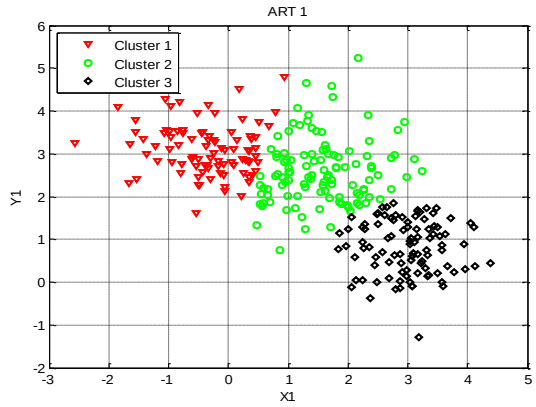

Figure 1(b): Clustered the ART1 dataset using CCSS.

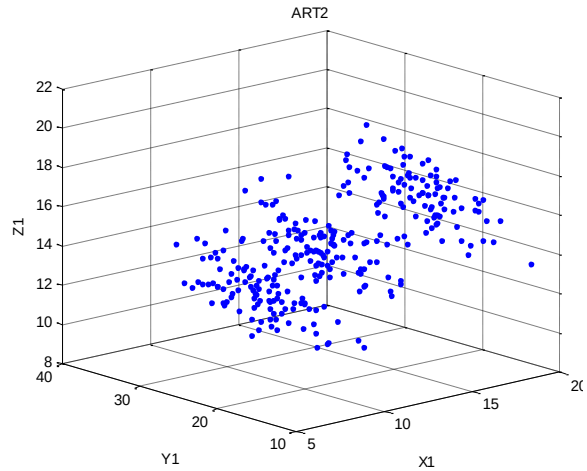

Figure 2 (a): Distribution of data in ART2 dataset.

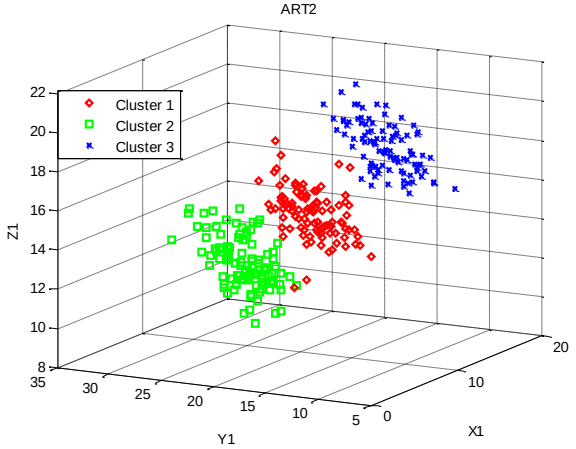

Figure 2(b): Clustered the ART2 data using CCSS.

$$F(i,j) = \frac{2 * (Recall * Precision)}{(Recall + Precision)} \quad (14)$$

Finally, the value of F-measure for a given clustering algorithm which consist of n number of data instances is given as

$$F(i,j) = \sum_{i=1}^{n} \frac{n_i}{n} . \max_i(F(i,j)) \quad (15)$$

Table 10: Parameters setting for CCSS algorithm

Parameters	Value
No. of CPs	No. of Clusters
c	0.1
∈	0.001

Table 11: Comparisons of different clustering algorithm with CCSS algorithm.

Dataset	*Parameters*	*K-means*	*GA*	*PSO*	*ACO*	*CSS*	*CCSS*
ART 1	Best	157.12	154.46	154.06	154.37	153.91	152.69
	Average	161.12	158.87	158.24	158.52	158.29	157.32
	Worst	166.08	164.08	161.83	162.52	161.32	159.86
	Std.	0.34	0.281	0	0	0	0
	F-Measure	99.14	99.78	100	100	100	100
ART2	Best	743	741.71	740.29	739.81	738.96	737.91
	Average	749.83	747.67	745.78	746.01	745.61	745.36
	Worst	754.28	753.93	749.52	749.97	749. 66	748.78
	Std.	0.516	0.356	0.237	0.206	0.209	0.182
	F-Measure	98.94	99.17	99.26	99.19	99.43	99.54
Iris	Best	97.33	113.98	96.89	97.1	96.47	96.38
	Average	106.05	125.19	97.23	97.17	96.63	96.52
	Worst	120.45	139.77	97.89	97.8	96.78	96.69
	Std.	14.631	14.563	0.347	0.367	0.14	0.11
	F-Measure	0.782	0.778	0.782	0.779	0.787	0.791
Wine	Best	16555.68	16530.53	16345.96	16530.53	16282.12	16183.94
	Average	18061	16530.53	16417.47	16530.53	16289.42	16218.42
	Worst	18563.12	16530.53	16562.31	16530.53	16317.67	16268.73
	Std.	793.213	0	85.497	0	10.31	43.16
	F-Measure	0.521	0.515	0.518	0.519	0.529	0.535
LD	Best	11397.83	532.48	209.15	224.76	207.09	205.98
	Average	11673.12	543.69	224.47	235.16	228.27	223.87
	Worst	12043.12	563.26	239.11	256.44	242.14	238.84
	Std.	667.56	41.78	29.38	17.46	18.54	13.29
	F-Measure	0.467	0.482	0.493	0.487	0.491	0.494
Cancer	Best	2999.19	2999.32	2973.5	2970.49	2946.48	2937.56
	Average	3251.21	3249.46	3050.04	3046.06	2961.16	2954.41
	Worst	3521.59	3427.43	3318.88	3242.01	3006.14	2978.34
	Std.	251.14	229.734	110.801	90.5	12.23	19.23
	F-Measure	0.829	0.819	0.819	0.821	0.847	0.856
CMC	Best	5842.2	5705.63	5700.98	5701.92	5672.46	5663.86
	Average	5893.6	5756.59	5820.96	5819.13	5687.82	5692.23
	Worst	5934.43	5812.64	5923.24	5912.43	5723.63	5727.18
	Std.	47.16	50.369	46.959	45.634	21.43	24.76

	F-Measure	0.334	0.324	0.331	0.328	0.359	0.367
Thyroid	Best	13956.83	10176.29	10108.56	10085.82	9997.25	9981.48
	Average	14 133.14	10218.82	10149.7	10108.13	10078.23	10054.32
	Worst	146424.21	10254.39	10172.86	10134.82	10116.52	10102.74
	Std.	246.06	32.64	27.13	21.34	49.02	47.28
	F-Measure	0.731	0.763	0.778	0.783	0.789	0.792
Glass	Best	215.74	278.37	270.57	269.72	203.58	241.08
	Average	235.5	282.32	275.71	273.46	223.44	256.45
	Worst	255.38	286.77	283.52	280.08	241.27	265.29
	Std.	12.47	4.138	4.55	3.584	13.29	8.38
	F-Measure	0.431	0.333	0.359	0.364	0.446	0.453
Vowel	Best	149422.26	149513.73	148976.01	149395.6	149335.61	149326.74
	Average	159242.89	159153.49	151999.82	159458.14	159128.19	159012.39
	Worst	161236.81	165991.65	158121.18	165939.82	164537.08	164286.97
	Std.	916	3105.544	2881.346	3485.381	3128.023	3316.58
	F-Measure	0.652	0.647	0.648	0.649	0.649	0.656

Table 12: Cluster center generated using CCSS method for ART1 and ART2 dataset.

Dataset	Center1	Center2
ART1	0.7036	3.1243
	1.8296	2.4327
	2.9504	0.9212

Dataset	Center1	Center2	Center3
ART2	10.8915	20.0376	15.7924
	14.9639	14.9693	17.8362
	9.2168	24.9836	12.9167

Table 13: Cluster center of Iris, Wine and CMC dataset using CCSS algorithm.

Dataset	Center1	Center2	Center3
Iris	5.01476	5.92624	6.75143
	3.36827	2.73618	3.06485
	1.46534	4.43953	5.63867
	0.24169	1.41694	2.12363
CMC	24.96782	33.57356	43.75427
	3.04023	3.15648	2.86549
	3.87451	3.56795	3.47264
	1.76986	3.65193	4.60256
	0.96258	0.78947	0.79624
	0.79371	0.69826	0.77486
	2.30346	2.10543	1.82644
	2.97296	3.29489	3.43257
	0.03826	0.05987	0.09278

Dataset	Center1	Center2	Center3
Wine	13.94261	12.62849	12.85965
	1.86138	2.34257	2.573642
	2.43564	2.33089	2.38526
	18.06218	20.95385	19.86478
	106.47693	98.69536	92.87932
	2.87896	2.14357	2.08431
	3.24652	1.78643	1.47298
	0.27138	0.43738	0.42846
	2.01853	1.44086	1.41791
	5.63729	4.34576	5.79516
	1.07814	0.95393	0.86541
	3.02437	2.47285	2.26429
	1138.0563	465.3951	688.0637

Table 14: Cluster center of Glass dataset using CCSS algorithm.

Dataset	Center1	Center2	Center3	Center4	Center5	Center6
	1.55643	1.53278	1.52736	1.51864	1.51961	1.52875
	12.98963	13.16383	13.09752	14.9486	14.10576	15.86713
	3.46978	0.25842	3.53789	0.06579	2.43568	3.73632
	1.03678	1.41572	1.34384	2.23528	2.63459	2.98763
Glass	72.01356	72.9382	72.59836	73.6853	71.26418	74.45752
	0.27906	0.31646	0.57812	0.05127	2.59642	5.1922
	9.46573	12.01694	8.3824	8.7243	6.02568	14.63761
	0.03548	0.05182	0.003929	1.03263	1.3356	2.6271
	0.05593	0.05673	0.05904	0.01861	0.01268	0.43456

Table 15: Cluster center of Cancer dataset using CCSS algorithm.

Dataset	Cancer								
Center1	2.89264	1.12476	1.21938	1.16893	1.99784	1.12856	2.06953	1.15432	1.07583
Center2	7.12836	6.64249	6.62812	5.61532	5.27268	8.23549	6.12637	6.10794	2.37681

Table 16: Cluster center of Thyroid dataset using CCSS algorithm.

Dataset	Thyroid				
Center1	0.9386	-1.1452	-0.4298	0.9786	4.1683
Center2	1.6832	-1.6678	-1.0207	2.3346	0.8634
Center3	-0.1723	0.1591	0.2596	-0.3010	-0.2206

Table 17: Cluster center of Vowel dataset using CCSS algorithm.

Dataset	Center 1	Center 2	Center 3	Center 4	Center 5	Center 6
	506.47598	407.6284	624.21692	356.82783	376.54276	437.90257
Vowel	1839.1372	1016.2586	1308.9836	2291.01353	2153.6867	990.76184
	2555.97546	2314.362	2332.9563	2976.8596	2676.4656	2661.53974

Table 18: Cluster center of LD dataset using CCSS algorithm.

Dataset	Center 1	Center 2	Center 3	Center 4	Center 5	Center 6
LD	87.92621	69.86842	25.85635	22.04346	27.10362	2.89347
	91.23956	75.06531	59.18264	38.92736	129.86542	5.96174

From Table 11, it can be seen that the results obtained from the CCSS algorithm are better as compared to the other algorithms. The best values achieved by the algorithm for iris, wine, cancer, CMC, glass, LD, thyroid and vowel datasets are 96.38, 16183.94, 2937.56, 5663.86, 241.08, 205.98, 9981.48 and 149326.74. The CCSS algorithm gives better results with most of the datasets. From the simulation results, it is also observed that CCSS algorithm achieve minimum value to the best distance parameter for the LD dataset and worst distance parameter for vowel dataset among all methods being compared. The standard deviation parameter shows how much the data are far from the cluster centers. The value of standard deviation parameter for CCSS algorithm is also smaller than other methods. Moreover, the CCSS algorithm provides better f-measure values than others which show higher accuracy of the said algorithm. To prove the viability of the results given in Table 11, the best centers obtained by the CCSS algorithm are given in Tables 12–18.

5 Conclusion

In this paper, a chaotic charged system search algorithm is applied to solve the clustering problem. In the

proposed algorithm, Newton second law of motion is used to get the optimal cluster centers but it is the actual electric force (Fk) and chaotic map which plays a vital role to obtain the optimal cluster centers. Hence, the working of the proposed algorithm is divided into two steps. First step involves the tuning of the CSS parameters using chaotic map. In the second step, the optimal cluster centers using Newton second law of motion are obtained. The CCSS algorithm can be applied for data clustering when the number of cluster centers (K) is already known. The performance of the CCSS algorithm is tested on the several datasets and compared with other algorithms; in which proposed algorithm provides better results and the quality of solutions obtained by the proposed algorithm is found to be superior in comparison to the other algorithms.

References

[1] Alpaydin, E. (2004) *Introduction to machine learning*. MIT press.

[2] Al-Sultan, K.S. (1995). A Tabu search approach to the clustering problem. *Pattern Recognition*, vol. 28, pp. 1443–1451.

[3] Al-Sultana, Khaled S., and Khan M. M. (1996). Computational experience on four algorithms for the hard clustering problem. *Pattern Recognition Letters*, vol. 17, no. 3, pp 295-308.

[4] Barbakh, W., Wu, Y., Fyfe, C. (2009). *Review of Clustering Algorithms*. Springer, Berlin/ Heidelberg, pp. 7–28.

[5] Camastra, F., Vinciarelli, A. (2008). Clustering Methods. *Machine Learning for Audio, Image and Video Analysis. Springer*, London, pp. 117–148.

[6] Cowgill, M.C., Harvey, R.J., Watson, L.T. (1999). A genetic algorithm approach to cluster analysis. *Comput. Math. Appl*., vol. 37, pp. 99–108.

[7] Dalli, A. (2003). Adaptation of the F-measure to cluster based lexicon quality evaluation, Association for Computational Linguistics, *In Proceedings of the EACL*, pp. 51-56.

[8] Dunn III, W. J., Greenberg, M. J., and Soledad S. C. (1976). Use of cluster analysis in the development of structure-activity relations for antitumor triazenes. *Journal of medicinal chemistry*, vol. 19, no. 11, pp. 1299-1301.

[9] Fathian, M., Amiri, B., Maroosi, A. (2007). Application of honey-bee mating optimization algorithm on clustering. *Appl. Math. Comput*., vol. 190, pp. 1502–1513.

[10] Forgy, E.W. (1965) Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, vol. 21, no. 2, pp. 768 - 769.

[11] Handl, J., Knowles, J., & Dorigo, M. (2003). On the performance of ant-based clustering. Design and Application of Hybrid Intelligent System. *Frontiers in Artificial Intelligence and Applications*, vol. 104, pp 204–213.

[12] Hatamlou, A., Abdullah, S., Hatamlou, M. (2011a). Data clustering using big bang-big crunch algorithm. *In the proceeding of innovative computing technology*, Springer Berlin Heidelberg publisher, Tehran, Iran pp.383–388.

[13] Hatamlou, A., Abdullah, S., Nezamabadi-pour, H. (2011b). Application of Gravitational Search Algorithm on Data Clustering. In the proceeding of the 6th international conference (RKST 2011), Banff, Canada, Springer Berlin Heidelberg publisher, pp. 337–346.

[14] Hatamlou, A., Abdullah, S., Nezamabadi-pour, H. (2012). A combined approach for clustering based on K-means and gravitational search algorithms. *Swarm and Evolutionary Computation*, vol. 6, pp. 47-52.

[15] He, Yi, Pan, W and Jizhen L. (2006). Cluster analysis using multivariate normal mixture models to detect differential gene expression with microarray data. *Computational statistics & data analysis*, vol. 51, no. 2, pp. 641-658.

[16] Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. U Michigan Press.

[17] Guobiao, Hu, Shuigeng Z., Jihong G., and Xiaohua Hu. (2008). Towards effective document clustering: A constrained< i> K</i>-means based approach. *Information Processing & Management*, vol. 44, no. 4, pp. 1397-1409.

[18] Jain, A.K., Murty, M.N., Flynn, P.J. (1999). Data clustering: A Review. *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323.

[19] Jain, A.K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letter*, vol. 31, pp. 651–666.

[20] Kao, Y.-T., Zahara, E., Kao, I.W. (2008). A hybridized approach to data clustering. *Expert Systems Appl*., vol. 34, pp. 1754–1762.

[21] Karaboga, D., and Ozturk C. (2011). A novel clustering approach: Artificial Bee Colony (ABC) algorithm. *Applied Soft Computing*, vol. 11, no. 1, pp. 652-657.

[22] Kaveh, A., and Talatahari, S. (2010 a). A novel heuristic optimization method: charged system search. *Acta Mechanica*, vol. 213, no. 3-4, pp. 267-289.

[23] Kaveh, A., and Talatahari, S. (2010b). Optimal design of skeletal structures via the charged system search algorithm. *Structural and Multidisciplinary Optimization*, vol. 41, no. 6, pp. 893-911.

[24] Kaveh, A., and Laknejadi, K. (2011a). A novel hybrid charge system search and particle swarm optimization method for multi-objective optimization. *Expert Systems with Applications*, vol. 38, no. 12, pp. 15475-15488.

[25] Kaveh, A., and Talatahari S. (2011b). An enhanced charged system search for configuration optimization using the concept of fields of forces. *Structural and Multidisciplinary Optimization*, vol. 43, no. 3, pp. 339-351.

[26] Kogan, J., Nicholas, C., Teboulle, M., Berkhin, P. (2006). A Survey of Clustering Data Mining

Techniques- *Grouping Multidimensional Data*. Springer Berlin Heidelberg, pp. 25–71.

[27] Krishna, K., Murty, MN. (1999). Genetic K-means algorithm. *IEEE Transactions on Systems, Man, Cybernet. Part B: Cybernet*. Vol. 29, pp. 433–439.

[28] Kumar, Y, and Sahoo G. (2014). A charged system search approach for data clustering. *Progress in Artificial Intelligence*, vol. 2, no. 2-3, pp. 153-166.

[29] MacQueen, James. (1967). Some methods for classification and analysis of multivariate observations. *In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 281-297, p. 14.

[30] Maimon, O., Rokach, L. (2010). *A survey of Clustering Algorithms, Data Mining and Knowledge Discovery Handbook*. Springer, pp. 269–298.

[31] Maulik, U., Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. *Pattern Recognition*, vol. 33, pp. 1455–1465.

[32] Murthy, C.A., Chowdhury, N. (1996). In search of optimal clusters using genetic algorithms. *Pattern Recognition Letter*, vol. 17, pp. 825–832.

[33] May RM. (1976). Simple mathematical models with very complicated dynamics. *Nature*, vol. 261, pp. 459–474.

[34] Pappas, Thrasyvoulos N. (1992). An adaptive clustering algorithm for image segmentation. *IEEE Transactions on Signal Processing*, vol. 40, no. 4, pp. 901-914.

[35] Satapathy, S. C., & Naik, A., (2011) Data clustering based on teaching-learning-based optimization. *In the proceeding of the second international conference (SMECOO),* Springer Berlin Heidelberg publisher, *Vishakhapatnam, Andhra Pradesh, India* pp. 148-156, 2011.

[36] Sahoo, A. J., & Kumar, Y. (2014). Modified Teacher Learning Based Optimization Method for Data Clustering. *In Advances in Signal Processing and Intelligent Recognition Systems*, Springer Berlin Heidelberg publisher, Kerala, India, pp. 429-437.

[37] Schuster, H. G., & Just, W. (2005) *Deterministic chaos: An introduction.* Wenham: Wiley-VCH Verlag.

[38] Selim, S.Z., Ismail, M.A. (1984). K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 81–87.

[39] Selim, S.Z., Alsultan, K. (1991). A simulated annealing algorithm for the clustering problem. *Pattern Recognition*, vol. 24, pp. 1003–1008.

[40] Shelokar, P. S., Jayaraman, V. K., and Kulkarni, B. D., (2004). An ant colony approach for clustering. *Analytica Chimica Acta*, vol. 509, no. 2, pp. 187-195.

[41] Sonka, Milan, Vaclav Hlavac, and Roger Boyle. (1998). *Image processing, analysis, and machine vision*. Champion and Hall.

[42] Sung, C.S., Jin, H.W. (2000). A tabu-search-based heuristic for clustering. *Pattern Recognition*, vol 33, pp. 849–858.

[43] Talatahari S., Farahmand A.B., Sheikholeslami R., and Gandomi AH. (2012) Imperialist competitive algorithm combined with chaos for global optimization. *Commun Nonlinear Sci Numer Simul.*, vol 17, pp. 1312–1319.

[44] Talatahari, S., Kaveh, A. and Sheikholeslami, R. (2011). An efficient charged system search using chaos for global optimization problems. *Int J Optim Civil Eng*., vol. 1, no. 2, pp. 305-25.

[45] Teppola, P., Mujunen, S-P and Minkkinen, P. (1999). Adaptive Fuzzy C-Means clustering in process monitoring. *Chemometrics and intelligent laboratory systems*, vol. 45, no. 1, pp. 23-38.

[46] Van der Merwe, D. W., and Engelbrecht, A. P. (2003). Data clustering using particle swarm optimization." *In Congress on Evolutionary Computation CEC-03*, vol. 1, pp. 215-220.

[47] Webb, A. (2002). *Statistical pattern recognition*. New Jersey: John Wiley & Sons, pp. 361–406.

[48] Zhang, B., Meichun H., and Dayal, U., (1999). K-harmonic means-a data clustering algorithm, Hewlett-Packard Labs *Technical Report* HPL.

[49] Zhou, H., and Yonghuai L. (2008). Accurate integration of multi-view range images using k-means clustering. *Pattern Recognition*, Vol. 41, No. 1. Pp. 152-175.