A New Clustering Approach based on Glowworm Swarm Optimization

Ibrahim Aljarah and Simone A. Ludwig
Department of Computer Science
North Dakota State University
Fargo, ND, USA
{ibrahim.aljarah,simone.ludwig}@ndsu.edu

***Abstract*—High-quality clustering techniques are required for the effective analysis of the growing data. Clustering is a common data mining technique used to analyze homogeneous data instance groups based on their specifications. The clustering based nature-inspired optimization algorithms have received much attention as they have the ability to find better solutions for clustering analysis problems. Glowworm Swarm Optimization (GSO) is a recent nature-inspired optimization algorithm that simulates the behavior of the lighting worms. GSO algorithm is useful for a simultaneous search of multiple solutions, having different or equal objective function values. In this paper, a clustering based GSO is proposed (CGSO), where the GSO is adjusted to solve the data clustering problem to locate multiple optimal centroids based on the multimodal search capability of the GSO. The CGSO process ensures that the similarity between the cluster members is maximized and the similarity among members from different clusters is minimized. Furthermore, three special fitness functions are proposed to evaluate the goodness of the GSO individuals in achieving high quality clusters. The proposed algorithm is tested by artificial and real-world data sets. The better performance of our proposed algorithm over four popular clustering algorithms is demonstrated on most data sets. The results reveal that CGSO can efficiently be used for data clustering.**

I. Introduction

Clustering [1] is a widely studied data mining and most important unsupervised learning technique used when analyzing data. Clustering algorithms can be used in many applications, for instance, pattern recognition [2], document categorization [3], and bioinformatics applications [4]. The core objective behind the clustering problem is to produce different groups from data instances without any information about the instance labels. The clustering algorithm collects the similar data instances having common attributes and splits them into different partitions/clusters based on a similarity metric.

Generally, clustering algorithms can be classified into three basic classes [5]: partitional clustering, density clustering, and hierarchical clustering. The partitional clustering (e.g., K-means) [6] constructs several disjoint clusters and then evaluates them by some measure such as minimizing the squared errors among the cluster representatives (centroids) and data instances. The density based clustering approaches (e.g., DBSCAN) [7] apply a density criterion to locate the dense regions that have more connectivity between the cluster members and then separates them by low density regions. On the other hand, hierarchical clustering [8] splits a big cluster into smaller ones or merges smaller clusters into their nearest cluster based on a similarity measure. In this paper, we are concerned with the partitional clustering. K-means clustering [6] is considered a common partitional clustering algorithm which is basically a minimization of the squared error objective function. K-means clustering suffers from some drawbacks such as the sensitivity of the initial centroids and the local optima convergence problem. In recent years, some researchers discussed clustering based on the idea of swarm intelligence [9] such as, ant colony optimization [10] and particle swarm optimization [11]. The use of swarm intelligence clustering algorithms is very efficient since these algorithms avoid the k-means drawbacks of the initial number of centroids as well as premature convergence.

Swarm intelligence [9] imitates the social natural communities such as birds flocks, ant colonies, and fish schools. The behavior of these communities is based on the receptor of the individual's interactions by communicating with each other to locate the food sources. Glowworm Swarm Optimization (GSO) [12] is one of the recent swarm intelligence algorithms, which belongs to the swarm intelligence field inspired by simulated experiments of the lighting worms' behavior. These glowworms are able to control their light emission and use it to glow for different purposes, such as attracting the prey, etc. GSO has been used in several applications such as the hazard sensing in ubiquitous environments [13] and mobile sensor networks and robotics [12]. The objective of the majority of the swarm intelligence algorithms is to locate the global solution for the given optimization problem. On the other hand, the GSO algorithm locates multiple solutions, having different or equal objective function values. The swarm in the GSO algorithm should have the ability to divide its members into disjoint groups to locate multiple solutions.

This paper makes use of GSO optimization to solve the clustering problem, which takes into account the advantages of the GSO multimodal search ability to locate optimal centroids. In addition, the proposed algorithm can discover the numbers of clusters without needing to provide the number in advance. Furthermore, three different fitness functions are introduced to add flexibility and robustness to the proposed algorithm. In

addition, the proposed algorithm is tested on real and artificial data sets with different shapes to demonstrate the clustering quality.

The remainder of this paper is organized as follows: Section II presents the related work in the area of clustering analysis based on nature-inspired optimization algorithms. In Section III, the classical glowworm swarm optimization approach is introduced. In Section IV, our proposed clustering algorithm is introduced. The experimental evaluation and results are shown in Section V, and Section VI presents our conclusions.

II. Related Work

Many clustering techniques are available in the literature [2, 7, 8] such as K-means [6], DBSCAN [7], Furthest First [14], and Learning Vector Quantization (LVQ) algorithm [15] for unsupervised clustering. Due to space constraints, we focus only on closely related work of clustering based nature-inspired optimization algorithms.

The clustering based nature-inspired optimization algorithms have received much attention to find better solutions for clustering analysis problems. The clustering problem in these algorithms is mapped to an optimization problem to locate the optimal solution based on different similarity metrics. Several clustering based nature-inspired optimization algorithms have been proposed to meet the challenges of clustering analysis problems.

In [16], the authors proposed a solution to the clustering analysis problem where the genetic algorithm capability was used. Their results showed that the genetic based clustering algorithm provides a good performance that is better than the K-means algorithm for different data sets. In [17], the Ant Colony Optimization (ACO) was used to perform the clustering analysis. The authors mainly formulated the problem by simulating the ant movement to group the data instances according to their similarity which is expressed by the available pheromone trails to guide ants to the optimal solutions. From their performance comparison results with other stochastic algorithms, their results in terms of the quality were better than the other techniques.

Clustering algorithm based Particle Swarm Optimization (PSO) was introduced by Omran et al. in [11] to solve the image clustering problem. The results of their algorithm showed that PSO is applicable to solve the clustering problems. Another work applied PSO in clustering analysis, proposed in [18], where the problem discussed was document clustering. The authors compared their results with some state-of-the-art techniques, and concluded that the PSO algorithm is applicable to locate compact clusters.

Most of the existing nature-inspired optimization clustering algorithms locate the global solution for the given optimization problem, whereas our proposed algorithm locates multiple solutions, having different or equal objective function values. In addition, for most algorithms, the number of clusters as a parameter is required in advance to guide the clustering process. However, in several practical applications, the determination of the number of the clusters before exploring the data set is impossible. Some other nature-inspired algorithms have suffered from the slow convergence and, the clusters quality is low in particular when the data set is noisy. Furthermore, the authors faced some problems to produce well separated clusters. Our proposed algorithm in this paper uses the modified GSO algorithm to solve the clustering analysis problem to tackle the slow convergence problem and the problem of determining the number of clusters in advance.

III. Classical Glowworm Swarm Optimization Algorithm

GSO is one of the most recent swarm intelligence method introduced by Krishnan and Ghose in 2005 [12]. GSO was first used for optimizing multimodal functions with equal or unequal objective function values. In GSO, glowworm swarm S, which consists of m glowworms, is distributed in the objective function search space. Each glowworm g_j ($j = 1...m$) is assigned a random position p_j inside the given function search space. Glowworm g_j carry its own luciferin level L_j, and has the vision range called local-decision range rd_j. The luciferin level depends on the objective function value and glowworm position. The glowworm with a better position is brighter than others, and therefore, has a higher luciferin level value and is very close to one of the optimal solutions. All glowworms seek the neighborhood set within their local decision range, and then move towards the brighter one within the neighborhood set. Finally, most of the glowworms gather to create compact groups in the function search space at multiple optimal locations of the objective function. Initially, all the glowworms carry an equal luciferin level (L_0). The rd and radial sensor range r_s are initialized with the same value (r_0). After that, the iterative process consists of several luciferin updates and glowworm movements are executed to find the optimal solutions. Throughout the luciferin level update, the objective function is evaluated at the current glowworm position (p_j) and then the luciferin level for all glowworms are adjusted based on the new objective function values. The luciferin level L_j is updated using the following equation:

$$L_j(t) = (1 - \rho)L_j(t - 1) + \gamma F(p_j(t)) \tag{1}$$

where $L_j(t-1)$ is the previous luciferin level for glowworm j; ρ is the luciferin decay constant ($\rho \in (0, 1)$); γ is the luciferin enhancement fraction, and $F(p_j(t))$ represents the objective function value for glowworm j at current glowworm position (p_j); t is the current iteration. After that, each glowworm j explores its own neighborhood region to extract the neighbors that have the highest luciferin level by applying the following rule:

$$z \in N_j(t) \;\; \textit{iff} \;\; Distance_{jz} < rd_j(t) \;\; and \;\; L_z(t) > L_j(t) \tag{2}$$

where z is one of the closer glowworms to glowworm j, $N_j(t)$ is the neighborhood set, $Distance_{jz}$ is the Euclidean distance between glowworm j and glowworm z, $rd_j(t)$ is the local decision range for glowworm j, and $L_z(t)$ and $L_j(t)$ are the luciferin levels for glowworm z and j, respectively.

After that, to select the best neighbor from the neighborhood set, the probabilities for all neighbors are calculated using the following equation:

$$Prob_{jz} = \frac{L_z(t) - L_j(t)}{\sum_{k \in N_j(t)} L_k(t) - L_j(t)} \tag{3}$$

where z is one of the neighborhood set $N_j(t)$ of glowworm j. After that, each glowworm selects the movement direction using the roulette wheel method whereby the glowworm with the higher probability has a higher chance to be selected from the neighborhood set. Then, the glowworm position (p_j) is adjusted based on the selected neighbor position (p_z) using the following equation:

$$p_j(t) = p_j(t-1) + s \ \frac{p_z(t) - p_j(t)}{Distance_{jz}} \tag{4}$$

$p_j(t-1)$ is glowworm j's previous position, s is a step size constant, and $Distance_{jz}$ is the Euclidean Distance between glowworms j and z. At the end of the GSO iteration, the local decision range rd_j is adjusted by the following equation:

$$\begin{aligned} rd_j(t) = min\{rs, max[0, rd_j(t-1) \\ +\beta(nt - |N_j(t-1)|)]\} \end{aligned} \tag{5}$$

$rd_j(t-1)$ is the previous rd_j, r_s is the radial sensor range constant, β is a model constant, nt is a constant parameter used to restrict the neighborhood set size, and $|N_j(t)|$ is the actual neighborhood set size. In our proposed algorithm, we relaxed the local decision range update step and fixed the value of the rd_j to be the same value as the r_s constant. However, the parameters nt and β are also relaxed.

IV. Proposed Algorithm

Our approach is a partitioning-based clustering which is motivated by the notion that instances are gathered around the centroids. K-means is one of the partitioning-based clustering techniques, where the centroids are extracted based on the weighted average of the data instances. The weighted average extraction method could be efficient if the data set is divided into organized shaped clusters. However, it is not efficient if the data set contains arbitrary shaped clusters. In our proposed algorithm, we are formulating the clustering problem as a multimodal optimization problem to extract the centroids based on glowworms' movement.

The proposed algorithm partitions the given data set into sets of clusters, such that each glowworm in the swarm tries to cover larger numbers of data set instances. Furthermore, each glowworm moves toward the glowworms that cover a larger amount of data instances and has smaller distances between the data instances in the local region for that glowworm which is controlled by r_s. In the next subsections, we provide a formal description of the clustering problem as well as the core components used in our proposed algorithm. Then, we discuss the proposed clustering algorithm.

A. Preliminaries

The clustering algorithm is applied on data set, D, consisting of n instances with d-dimensions, each instance is represented by x_i, $i = 1...n$. Given D, a clustering algorithm tries to extract a set of clusters $C = \{C_1, C_2, ..., C_k\}$, each is represented with a point called centroid, such as $c = \{c_1, c_2, ..., c_k\}$, where k is the number of centroids in the c centroid set. Furthermore, the clustering algorithm tries to maximize the similarity of the instances in the same cluster, and to minimize the similarity of instances from different clusters. In addition, each cluster should have at least one instance assigned to it, and the different clusters should be disjoint such that $\bigcap_{i..k} C_i = \{\}$ and $\bigcup_{i..k} C_i = D$, which ensures that there is no empty cluster. The Sum Squared Errors (SSE) fraction is calculated using the following equation:

$$SSE = \sum_{j=1}^{k} \sum_{i=1}^{|C_j|} (Distance(x_i, c_j))^2 \tag{6}$$

Another fraction is used in our proposed algorithm called Inter-Distance, which is calculated by the following equation:

$$InterDist = \sum_{i=1}^{k} \sum_{j=i}^{k} (Distance(c_i, c_j))^2 \tag{7}$$

In this paper, we use the Euclidean distance to calculate the $Distance$.

The swarm S used in the GSO optimization process consists of m glowworms, where each glowworm is represented by a vector, g_j, $j = 1...m$. Each g_j has 5 components: luciferin level (L_j), fitness function value (F_j), d-dimensional position vector (p_j), coverage set (cr_j) which is the set of the data instances that are covered by g_j, and intra-distance ($intraD_j$) between the cr_j set members and g_j position. The g_j should cover at least one data instance in its local range. The local range is a constant and is equal to the radial sensor range r_s, which is the same for all glowworms in swarm S. Furthermore, keeping the local range constant throughout the clustering process ensures that a glowworm keeps moving towards the optimal glowworms in all cases, even if it does not have neighbors or if it has many neighbors around.

B. Clustering based GSO Algorithm - CGSO

In recent years, GSO has been proven to be effective to solve optimization problems [19]. In data clustering, it can be formulated as an optimization problem that finds the optimal centroids of the clusters rather than to find optimal data partitions. The strength of optimization motivates us to apply GSO for finding the optimal solutions for the clustering problems. GSO is distinguished from other optimization techniques (that locate one local or global optimal solution), by finding multiple optimal solutions. The found solutions either have equal values for the dedicated objective function or not.

In our proposed clustering algorithm CGSO, the GSO objective is adjusted to locate multiple optimal centroids such that each centroid represents a sub-solution and the combination

of these sub-solutions formulate the global solution for the clustering problem. The proposed CGSO consists of four main phases: initialization phase, luciferin level update, glowworm movement, and candidate centroids set construction.

In the initialization phase, first an initial glowworm swarm S of size m is created. For each glowworm g_j, a random position vector (p_i) is generated using uniform randomization within the given search space within the minimum and the maximum values that are calculated from the data set D. Then, the luciferin level (L_j) is initialized using the initial luciferin level L_0. The fitness function value F_j is initialized to zero. The local range r_s is set to an initial constant range r_0. Secondly, after initializing the swarm, the set of data instances cr_j which are covered by g_j, is extracted from data set D, and the $intraD_j$ is calculated using the following equation:

$$intraD_j = \sum_{i=1}^{|cr_j|} Distance(cr_{ji}, g_j) \tag{8}$$

where cr_{ji} is the data instance i which is covered by g_j; $|cr_j|$ is the number of data instances which is covered by g_j. Then, in the last step of the initialization phase, the swarm-level fractions SSE and $InterDist$ are calculated.

To initialize SSE, we extract the glowworms list that covered the highest number of data instances (the glowworms have the maximum $|cr_j|$ sizes). These glowworms should be disjointed from each other, where each glowworm is located in a different region (the distance between any pair of these glowworms should be greater than r_s). The extracted glowworm list is considered the initial set of the candidate centroid c. After that, the candidate centroid set c is used to calculate the initial SSE using Equation 6. The same initial set c is also used to calculate the $InterDist$ which is calculated by Equation 7. After the initialization phase, an iterative process is performed to find optimal glowworms that represent the clustering problem centroids. The result of each iteration is an updated swarm with updated candidate centroids set c. In the luciferin level update phase, firstly, the fitness function F is evaluated to assign new F_j values for each glowworm using the glowworm position and other information.

Three different fitness functions are proposed to evaluate the goodness of the glowworm. For all proposed fitness functions, each glowworm tries to maximize the coverage percentage from the data instances $|cr_j|$ by keeping the intra-distances $intraD_j$ among the covered data instances and the glowworm as minimum. Furthermore, we used normalized fractions for the $|cr_j|$ and $intraD_j$ by dividing the total number of data instances n and $\max_j(intraD_j)$, respectively, to avoid the biased state between the two fractions. The fitness functions are different from each other depending on the swarm-level fractions (SSE, and $InterDist$) that are used. The first fitness function is given by the following equation:

$$F1(g_j) = \frac{\frac{1}{n}|cr_j|}{SSE \times \frac{intraD_j}{\max_j(intraD_j)}} \tag{9}$$

In $F1(g_j)$, and beside the purpose of maximizing $|cr_j|$ and minimizing $intraD_j$, we incorporate SSE swarm-level fraction to be minimized between the candidate centroids set as a whole. The second fitness function is given by the following equation:

$$F2(g_j) = \frac{InterDist \times \frac{1}{n}|cr_j|}{\frac{intraD_j}{\max_j(intraD_j)}} \tag{10}$$

In $F2(g_j)$, we incorporate $InterDist$ swarm-level fraction in the process, and this fraction should be maximized among the candidate centroids set c. The third fitness function is given by the following equation:

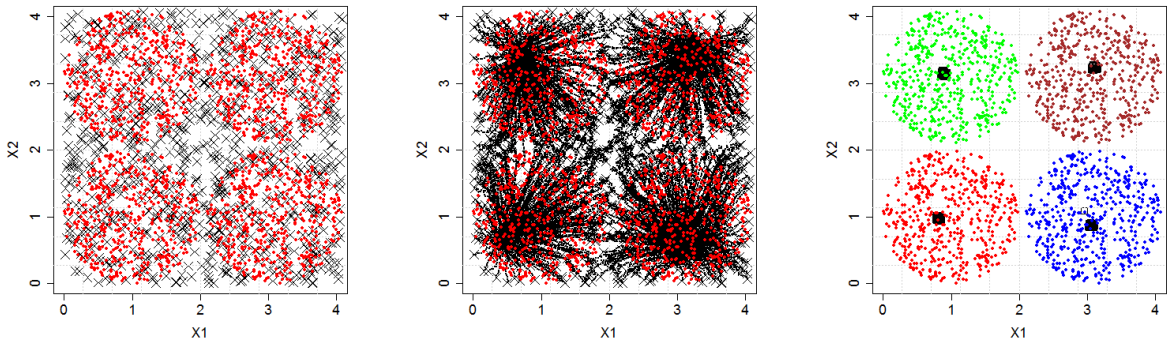
$$F3(g_j) = \frac{InterDist \times \frac{1}{n}|cr_j|}{SSE \times \frac{intraD_j}{\max_j(intraD_j)}} \tag{11}$$

In $F3(g_j)$, a combination between maximization of the $InterDist$ and minimization of the SSE fractions is added to the $F3(g_j)$ at the same time. After the fitness function evaluation for glowworm g_j, the luciferin level L_j is updated using Equation 1. Then, each glowworm g_j locates the neighborhood group using Equation 2, and the neighbor probability values are calculated based on Equation 3 to find the best neighbor using the roulette wheel selection method. Then, the glowworm is moved towards the best neighbor by updating its p_j vector by Equation 4 using the best neighbor position. After that, $|cr_j|$, and $intraD_j$ are updated based on the new glowworm g_j positions.

The candidate centroid set c is reconstructed based on the highest fitness values (F_j), and not like the way they are extracted during the initialization phase, which is based on the highest number of data instances (the glowworms have the maximum $|cr_j|$). The same rule is used during the internalization phase to construct candidate centroid set c, where all c members should be disjoint from each other such as each glowworm should locate in different regions and the distance between the glowworm pairs should be greater than range r_s. After that, the candidate centroid set c is used to calculate the new value for SSE which is calculated by Equation 6. In addition, the same candidate centroids set c is also used to calculate $InterDist$, which is calculated by Equation 7. Then, the fitness function is reevaluated using the new information. The iterative process is continued until the size of the candidate centroid set c becomes less than a specific threshold (minimum number of centroids is given), or the maximum number of iterations is achieved. The candidate centroid set c decreases throughout the iterative process, and after the clustering process is completed, the candidate centroid set is used to evaluate the clustering results.

C. Illustrative Example

Figure 1 shows an illustrative example of the CGSO clustering algorithm process to visualize the clustering results. An artificial data set with 2 dimensional instances is generated with 4 balanced clusters such as each cluster formulates a circle. Figure 1(a) shows the initial swarm state distributed in



(a) Initial State (b) Glowworms Movements (c) Glowworms Final Locations (centroids)

Fig. 1. Clustering process for the artificial data set with swarm size=1,000, maximum number of iterations=200, and r_s=1.2: the glowworms start from an initial random location and move to one of the centroids. 1(a) The initial random glowworm locations (small black crosses) with data set instances (red points). 1(b) The movements of the glowworms throughout the clustering process. 1(c) The final locations of glowworms (small squares) after the clustering process with 4 centroids, each cluster in the data set has a different color based on the minimum distances to the centroid.

the search space using the random uniform distribution and the scattered artificial data set in the same search space. The second part, Figure 1(b) shows glowworm movements towards the near optimal centroids. At the end, all glowworms are gathered at the 4 optimal centroids as shown in Figure 1(c).

V. Experiments and Results

This section presents a performance analysis to investigate the efficiency of the CGSO algorithm in the data clustering. We present the results obtained using the CGSO algorithm on well-known data sets to conduct a reliable comparison. Furthermore, the comparisons between the three introduced fitness functions are presented to show the algorithm's robustness. In addition, we present the comparison of the CGSO with other four well-known clustering algorithms: K-Means clustering [6], average linkage agglomerative Hierarchical Clustering (HC) [8], Furthest First (FF) [14], and Learning Vector Quantization (LVQ) [15], which have been used in the literature and we analyze their performance. Finally, the time complexity and algorithm convergence are discussed.

A. Environment

We ran the experiments on the PC containing 6GB of RAM, 4 Intel cores (2.67GHz each). For our experiments, we used Java runtime 1.6 to implement the proposed algorithm and WEKA [20] open source for comparisons. We present the results obtained using the CGSO on 7 typical data sets which are used in the literature. The first 5 data sets are obtained from the UCI database repository[1]. Furthermore, we used 2 artificial data sets from ELKI[2], and use them to visualize the clustering results. All data sets are described in Table I.

TABLE I
Summary of the data sets

Data set	#Records	#Features	#Clusters	Type
Iris	150	4	2	*Real*
Ecoli	327	7	5	*Real*
Glass	214	9	6	*Real*
Balance	625	4	3	*Real*
Seed	210	7	3	*Real*
Mouse	490	2	3	*Artificial*
VaryDensity	150	2	3	*Artificial*

B. Evaluation Measures

In our experiments, we use two different measures for the evaluation of the cluster quality: entropy and purity [21]. These are the standard measures of the clustering quality. Entropy measures how the various semantic classes are distributed within each cluster, and is calculated by the following equation:

$$Entropy = \sum_{j=1}^{k} \frac{\mid C_j \mid}{n} E(C_j) \tag{12}$$

where C_j contains all data instances assigned to cluster j by the clustering algorithm, n is the number of data instances in the data set, k is the number of clusters that is generated from the clustering process, and $E(C_j)$ is the individual entropy of cluster C_j which is defined by the following equation:

$$E(C_j) = -\frac{1}{log\ q} \sum_{i=1}^{q} \frac{\mid C_j \cap L_i \mid}{\mid C_j \mid} log\ (\frac{\mid C_j \cap L_i \mid}{\mid C_j \mid}) \tag{13}$$

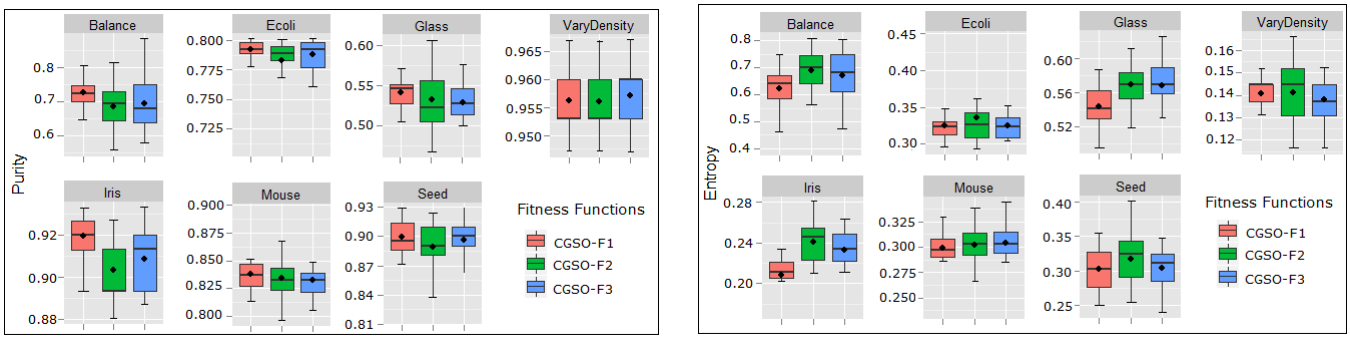
where L_i denotes the true assignments of the data instances in cluster i; q is the number of actual clusters in the data set. Similarly to the previous equation, the purity of the clustering is defined as:

$$Purity = \frac{1}{n} \sum_{j=1}^{k} \max_{i} (\mid L_i \cap C_j \mid) \tag{14}$$

Smaller entropy values and larger purity values indicate better clustering solutions. The clustering quality is perfect if clusters only contain data instances from one true cluster; in that case the purity and entropy equal 1 and 0, respectively.

We used the GSO settings that are recommended in [19]. We used $\rho = 0.4$; $\gamma = 0.6$; the initial luciferin level $L_0 = 5.0$; the

[1] http://archive.ics.uci.edu/ml/index.html
[2] http://elki.dbs.ifi.lmu.de/wiki/DataSets



(a) Purity (b) Entropy

Fig. 2. Box plots of the purity and entropy results obtained by comparing three different fitness functions (F1, F2, and F3) with different data sets. The small solid circles represent the average of 25 runs, and the bar inside the rectangle shows the median; minimum and maximum values are represented by whiskers below and above the box.

step size $s = 0.03$. The swarm size used in our experiments is equal to 1000 glowworms and the maximum number of iterations is set to 200. Furthermore, since radial sensor range (local range) r_s depends on the data set, preliminary experiments were conducted by varying the r_s values in order to choose the best r_s value for each individual data set. The best r_s values that were empirically determined for the Iris, Ecoli, Glass, Balance, Seed, Mouse, and VaryDensity data sets are 1.35, 0.38, 0.38, 0.48, 0.052, and 0.06, respectively.

C. Results

This section presents the comparison among the three proposed fitness functions to evaluate the impact of these functions in our proposed CGSO algorithm. In addition, comparisons with other well-known clustering methods are proposed. In order to abbreviate our proposed algorithm variants which are based on different fitness functions, a specific format is used to distinguish them, such that CGSO-F1, CGSO-F2, CGSO-F3 are our proposed algorithm using F1, F2, and F3, respectively.

To evaluate the impact of the fitness functions in our proposed CGSO algorithm, we compared the three variants (CGSO-F1, CGSO-F2, and CGSO-F3) to show the algorithm flexibility and robustness. The purity and entropy results distribution for applying CGSO on the given data sets are shown as the box plots in Figure 2. It can be seen from Figure 2(a) that the highest average purity (25 independent runs) is produced using F1 for all data sets (however, it is not statistically significant). Furthermore, it can be noted from Figures 2(b), F1 achieved the minimum average entropy for the Iris, Glass, Balance, Seed data sets (however, again not statistically significant). F2 obtained the minimum average entropy for the Ecoli and VaryDensity data sets.

A comparison of the clustering quality in terms of purity and entropy with other clustering methods are shown in Tables II and III, respectively. For our proposed algorithm, the average and the standard deviation of the purity and entropy results for 25 independent runs for each of the three fitness functions as well as the best results (within brackets) are presented in Tables II and III. The highest purity and smallest entropy values in each case are shown in bold. It can been seen from the Table II, CGSO-F1 outperforms all other clustering techniques for most data sets with an average purity of 0.919, 0.792, 0.541, 0.726, 0.900, and 0.956 for Iris, Ecoli, Glass, Balance, Seed, VaryDensity, respectively. The HC obtained the best purity for the Mouse data set (0.91), however, its result was not much different compared to the result achieved by CGSO.

For the entropy results in Table III, where a smaller entropy implies a better result, CGSO-F1 shows competitive performance and outperforms other clustering techniques for most data sets with an average entropy of 0.209, 0.543, 0.622, and 0.302 for Iris, Glass, Balance, and Seed, respectively. The HC obtained the best entropy for the Mouse data set (0.165). The K-Means obtained the best entropy for the Ecoli data set (0.307). Furthermore, CGSO-F3 obtained the best entropy for the VaryDensity data set. Figure 3 shows the visualization of clustering quality results (best run is selected from the highest function results) of the Mouse data set. Figure 3(b) shows that the clustering quality results of CGSO-F3 (obtains the best results among the three functions), and Figure 3(c) shows the clustering quality results of K-means. It can been seen that CGSO-F3 is able to assign the data instances to the correct clusters, with highest purity of 0.896, while K-means' purity result is 0.827.

D. Complexity and convergence analysis

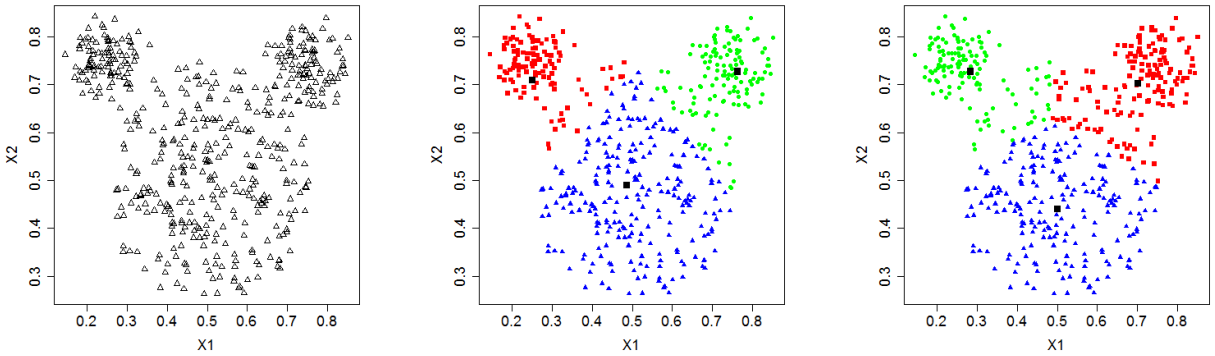
The overall time complexity of our proposed algorithm depends mainly on the amount of time it requires to find the neighborhood set for each glowworm and the amount of time it requires to retrieve the coverage set (cr_j) from the data set that is covered by individual glowworm g_i as well as the time to calculate $IntraDist_j$ between the glowworm g_i and its coverage set (cr_j). Furthermore, the overall time also depends on the dimensionality of the data set used, as well as the swarm size and the maximum number of iterations. The three proposed fitness functions F1, F2, F3 share the two fractions $|cr_j|$ and $IntraDist_j$ that are distinguished from each other in terms of use of SSE and $InterDist$ swarm-level fractions. The time needed to calculate SSE and $InterDist$ decreases

TABLE II
PURITY RESULTS

Data set	CGSO-F1	CGSO-F2	CGSO-F3	K-Means	HC	FF	LVQ
Iris	**0.919** ± 0.090 [0.933]	0.903 ± 0.014 [0.927]	0.909 ± 0.012 [0.933]	0.887	0.887	0.860	0.507
Ecoli	**0.792** ± 0.006 [0.801]	0.779 ± 0.029 [0.801]	0.789 ± 0.012 [0.801]	0.774	0.654	0.599	0.654
Glass	0.541 ± 0.018 [0.570]	0.533 ± 0.036 [0.607]	0.529 ± 0.020 [0.575]	**0.542**	0.463	0.481	0.411
Balance	**0.726** ± 0.039 [0.805]	0.685 ± 0.061 [0.810]	0.694 ± 0.074 [0.882]	0.659	0.632	0.653	0.619
Seed	**0.900** ± 0.016 [0.929]	0.889 ± 0.026 [0.924]	0.897 ± 0.018 [0.929]	0.876	0.895	0.667	0.667
Mouse	0.837 ± 0.013 [0.880]	0.834 ± 0.018 [0.876]	0.833 ± 0.018 [0.896]	0.827	**0.910**	0.800	0.843
VaryDensity	**0.956** ± 0.006 [0.967]	0.956 ± 0.007 [0.967]	0.957 ± 0.006 [0.967]	0.953	0.667	0.667	0.567

TABLE III
ENTROPY RESULTS

Data set	CGSO-F1	CGSO-F2	CGSO-F3	K-Means	HC	FF	LVQ
Iris	**0.209** ± 0.018 [0.170]	0.241 ± 0.020 [0.210]	0.233 ± 0.018 [0.176]	0.264	0.230	0.307	0.790
Ecoli	0.325 ± 0.013 [0.295]	0.342 ± 0.050 [0.293]	0.324 ± 0.014 [0.305]	**0.307**	0.522	0.611	0.579
Glass	**0.543** ± 0.023 [0.495]	0.569 ± 0.022 [0.519]	0.568 ± 0.030 [0.507]	0.567	0.662	0.646	0.754
Balance	**0.622** ± 0.078 [0.446]	0.690 ± 0.068 [0.560]	0.669 ± 0.099 [0.395]	0.701	0.739	0.654	0.753
Seed	**0.302** ± 0.031 [0.250]	0.317 ± 0.039 [0.253]	0.305 ± 0.027 [0.239]	0.327	0.298	0.537	0.577
Mouse	0.299 ± 0.015 [0.253]	0.302 ± 0.021 [0.248]	0.304 ± 0.020 [0.234]	0.319	**0.165**	0.351	0.262
VaryDensity	0.141 ± 0.013 [0.116]	0.141 ± 0.017 [0.116]	**0.138** ± 0.016 [0.116]	0.145	0.421	0.466	0.728



(a) Original Data set (b) Clustering with CGSO-F3 (c) Clustering with K-means

Fig. 3. Clustering results for the Mouse data set, where the black boxes represent the centroids. 3(a) The original Mouse data set. 3(b) The clustering results with CGSO using fitness function F3. 3(c) The clustering results with K-means.

TABLE IV
RUNNING TIME AND NUMBER OF ITERATIONS

Data set	CGSO-F1		CGSO-F2		CGSO-F3	
	Average Time (s)	Average #Iterations	Average Time (s)	Average #Iterations	Average Time (s)	Average #Iterations
Iris	10.79	101.80	11.58	108.92	10.74	101.04
Ecoli	3.37	17.56	3.50	18.44	3.82	20.16
Glass	15.95	78.72	18.35	89.88	23.02	82.20
Balance	14.68	96.40	13.87	95.84	13.65	94.04
Seed	5.04	27.36	5.00	27.24	4.82	26.12
Mouse	1.61	17.72	1.79	19.24	1.40	15.20
VaryDensity	7.72	122.40	7.07	110.16	7.88	125.88

with consequent iterations since the number of the candidate centroid set size $|c|$ is also reduced. Table IV shows the average running time and average number of iterations (over 25 runs) are required to achieve the optimal number of centroids. We can note that CGSO-F3 has a shorter average running time for Iris, Balance, Seed, and Mouse data sets compared to CGSO-F1, and CGSO-F2. For example, CGSO-F3 needs 10.74 seconds to converge for the Iris data set, whereas CGSO-F1 and CGSO-F2 need 11.58 and 10.74 seconds, respectively, for the same data set. Furthermore, CGSO-F3 converges faster than the other two for the Iris, Balance, Seed, and Mouse data sets. For instance, CGSO-F3 needs 101.04 iterations on average for Iris, whereas CGSO-F1 and CGSO-F2 need 101.8, and 108.92, respectively, for the same data set. In addition, CGSO-F1 has shorter average running time for Ecoli and Glass data sets as well as smallest average number of iterations, for example, CGSO-F1 needs 3.364 seconds and 17.56 iterations on average to converge for the Ecoli data set. Furthermore, CGSO-F2 has a shorter average running time for the VaryDensity data set and has the smallest average number

of iterations such as CGSO-F2 needs 7.07 seconds and 110.16 iterations on average to converge.

VI. Conclusion

In this paper, we have presented a new clustering algorithm based on glowworm swarm optimization which takes into account the advantages of the GSO multimodal search capability to locate optimal centroids. The proposed algorithm CGSO can discover the clusters without needing to provide the number of clusters in advance. Experimental results on several real and artificial data sets with different characteristics show that our proposed algorithm is efficient compared to well-known clustering methods that have been used in the literature. In addition, three different fitness functions were proposed to add flexibility and robustness to the proposed algorithm. The average clustering quality, in terms of purity and entropy results over 25 runs, shows that our proposed algorithm is robust since the variances are relatively small. Our future research will include the verification of our proposed algorithm on other types of data sets with higher dimensions as well as we will investigate the effectiveness of our proposed algorithm with larger data set sizes. Furthermore, we will investigate to find an efficient way to determine the radial range r_s parameter, for which preliminary experiments are needed.

Acknowledgment

The authors acknowledge the support of the NDSU Advance FORWARD program sponsored by NSF HRD-0811239 and ND EPSCoR through NSF grant EPS-0814442.

References

[1] P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, 1st ed. Addison Wesley, May 2005.

[2] A. Baraldi and P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition." *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 29, no. 6, pp. 778–785, Dec. 1999.

[3] A. Silic, M.-F. Moens, L. Zmak, and B. Basic, "Comparing document classification schemes using k-means clustering," vol. 5177, pp. 615–624, 2008.

[4] D. K. Tasoulis, V. P. Plagianakos, and M. N. Vrahatis, "Unsupervised clustering of bioinformatics data," in *In ESIT, Hybrid Systems and their implementation on Smart Adaptive Systems*, 2004, pp. 47–53.

[5] J. Han, *Data Mining: Concepts and Techniques*. CA, USA: Morgan Kaufmann Publishers Inc., 2005.

[6] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.*, vol. 1. Univ. of Calif. Press, 1967, pp. 281–297.

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD'96*, 1996, pp. 226–231.

[8] Y. Zhao, G. Karypis, and U. Fayyad, "Hierarchical clustering algorithms for document datasets," *Data Min. Knowl. Discov.*, vol. 10, no. 2, pp. 141–168, Mar. 2005.

[9] A. Engelbrecht, *Computational Intelligence, An Introduction*, second edition ed. Wiley, 2007.

[10] J. Handl, J. Knowles, and M. Dorigo, "Strategies for the increased robustness of ant-based clustering," in *Engineering Self-Organising Systems*, ser. Lecture Notes in Computer Science. Springer, 2004, vol. 2977, pp. 90–104.

[11] E. A. Omran M. and S. A., "Particle swarm optimization method for image clustering," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 3, pp. 297–322, 2009.

[12] K. Krishnanand and D. Ghose, "Detection of multiple source locations using a glowworm metaphor with applications to collective robotics," in *IEEE Swarm Intelligence Symposium*, CA, USA, june 2005, pp. 84 – 91.

[13] K. N. Krishnanand and D. Ghose, "Glowworm swarm optimization algorithm for hazard sensing in ubiquitous environments using heterogeneous agent swarms," *Soft Computing Applications in Industry*, vol. 226, pp. 165–187, 2008.

[14] D. S. Hochbaum and D. B. Shmoys, "A best possible heuristic for the k-Center problem," *Mathematics of Operations Research*, vol. 10, no. 2, pp. 180–184, May 1985.

[15] T. Kohonen, "Learning Vector Quantization: The Handbook of Brain Theory and Neural Networks," pp. 631–634, 2003.

[16] U. Maulik and S. Bandyopadhyay, "Genetic algorithm-based clustering technique," *Pattern Recognition*, vol. 33, no. 9, pp. 1455 – 1465, 2000.

[17] P. Shelokar, V. Jayaraman, and B. Kulkarni, "An ant colony approach for clustering," *Analytica Chimica Acta*, vol. 509, no. 2, pp. 187 – 195, 2004.

[18] X. Cui, T. Potok, and P. Palathingal, "Document clustering using particle swarm optimization," in *IEEE Swarm Intelligence Symposium*, Ca, USA, 2005, pp. 185–191.

[19] K. Krishnanand and D. Ghose, "Glowworm swarm optimisation: a new method for optimising multi-modal functions," *International Journal of Computational Intelligence Studies*, vol. 1, pp. 93–119, 2009.

[20] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD*, vol. 11, pp. 10–18, Nov. 2009.

[21] Y. Zhao and G. Karypis, "Evaluation of hierarchical clustering algorithms for document datasets," in *Proceedings of the eleventh CIKM '02*, NY, USA, 2002, pp. 515–524.