Investigating the Particle Swarm Optimization Clustering Method on Nucleic Acid Sequences

Barileé B. Baridam

Department of Computer Science, University of Pretoria
South Africa

***Abstract*— Particle swarm optimization (PSO) has been employed on several optimization problems, including the clustering problem. PSO has also been employed in the clustering of data of different structure and dimensionality. In this paper it is employed in the clustering of nucleic acid sequences. The application of clustering, as a statistical tool, in the analysis of data of varied complexity has been treated by several researchers. Besides PSO, distance-based algorithms have been widely proposed for the clustering problem. This paper investigates the efficiency of PSO clustering on nucleic acid sequences through the introduction of distance measures among which are the Euclidean distance measure, Manhattan distance, edit distance and the codon-based scoring method (COBASM). Sub-objective weights were introduced to observe the behaviour of PSO under various conditions. From the result obtained, PSO-based clustering produces compact and well-separated clusters. However, the result varied with distance measure.**

Keywords: Nucleic acids, clustering, similarity measure, PSO

IV. INTRODUCTION

Clustering, as an important aspect of knowledge discovery, has as its main aim to group related elements based on some predefined measure of closeness or proximity. Clustering involves the discovery of relationships in data without the application of any prior knowledge of the relationships. The final result of clustering depends on the perception of the user through the application of some subjective decisions. These decisions are (1) the definition and measurement of the relationships between the data elements that would warrant clustering, (2) the actual number of clusters expected in the clustering task, and (3) the representation of the generated clusters. Most conventional clustering algorithms employ the use of distance or similarity measures to determine objects proximity and to generate clusters [1].

Clustering, in computational biology, goes beyond a mere statistical tool for information retrieval. It actually reveals the genetic information of participating sequences. Such information helps in the determination of gene families and the establishment of implicit links between them. Clustering of biological sequence data presents a great challenge to the computing society as well as to biologists. This challenge arises from the fact that sequence data cannot be easily clustered by the application of conventional distance or similarity measures Also, string edit distance algorithms employed in string comparisons and string similarity searches are mostly not suitable in biological sequence data clustering [2]. This is basically because the structural nature of biological sequences makes string edit distance not appropriate. For example, the edit between the strings *bbbbbbbddddddd* and *ddddddbbbbbbb* clearly shows there is no similarity between the strings. However, looking at the strings biologically, there is an element of structural similarity which the edit distance neglects. Since the issue of structural similarity is major in biological sequence analysis the edit distance and other distance-based algorithms are incapable of clustering biological sequences.

The introduction of particle swarm optimization (PSO) becomes necessary at this point to since it has been proven to be robust in the handling of optimization problems [3]. This means, then, that distance measures will have to be used with the PSO-based clustering method to observe their performance under various conditions. Since PSO has already been successfully applied to data clustering and image segmentation [4], [3], this paper investigates the efficiency of PSO-based clustering method in clustering nucleic acid sequences with respect to the distance measures. The measures used are the Euclidean distance, the Manhattan distance, and the

edit distance. The codon-based scoring method (COBASM) [5] is also used with PSO in the clustering of nucleic acids sequences. COBASM considers the application of codons[1] to maintain the structural similarity of sequences.

The remainder of the paper is organized as follows: Section II presents related work, Section III discusses particle swarm optimization, Section IV describes distance measures employed in the PSO clustering task, Section V is devoted to the experimental results obtained with the PSO-based sequence clustering, and Section VI presents the conclusion, and directions for further research.

V. RELATED WORK

Several methods have been proposed for data clustering tasks [6]. These methods have been divided into two broad categories: Hierarchical and partitional. One of the highly researched partitional algorithm is the K-means algorithm. It is a partitional iterative clustering approach [7] to data clustering. The K-means algorithm is popular and most criticized for its demanding the number of clusters for a clustering task a priori. However, K-means algorithm is simple and easier to implement with linear time complexity.

The Fuzzy-C means (FCM) is a clustering method that introduces the fuzzy version of the K-means [8], [9]. Although FCM still demands the provision of the value of K a priori, it outperforms the K-means in that it is less affected by the presence of uncertainty in the data [10].

The K-harmonic means algorithm computes the harmonic means of each cluster centre to every pattern and then updates the cluster centroids accordingly [11]. The K-harmonic means is less affected by the initial conditions. Experimental results show that it outperforms the FCM and K-means [12].

Yang and Wang [2] proposed CLUSEQ for the clustering of sequences based on sequence structural features and exhibited statistical properties. CLUSEQ builds a probabilistic suffix tree in the initialization of sequence. Although this method seems better than most sequence clustering methods, CLUSEQ does not consider that some sequences can exhibit closer similarity than others depending on whether the sequences and amino acids or nucleic acids [13].

Most clustering methods employ distance measures to determine the proximity of data elements. Some of these distance/similarity measures are mentioned in Section IV. However, the edit distance, originally designed for similarity search is also employed in clustering tasks. It has been proven that the edit distance lacks the ability to handle sequences based on their structural similarities [2]. Muthukrishnan and Sahinalp [14] proposed the edit distance with the use of block operations all in an attempt to optimize the edit distance's performance. Furthermore, to still optimize the efficiency of the edit distance, Cormode and Muthukrishnan [15] introduced a greedy algorithm to reduce moves of substrings to moves of characters and convert moves of characters to only inserts and deletes.

In the same vein, Lopresti and Tomkins [16] proposed block edit models for approximate string matching, which could be extended to sequence clustering, by examining string edit distance in which two strings are compared by extracting collections of substrings and placing the two strings into correspondence with each other.

VI. PARTICLE SWARM OPTIMIZATION

Particle swarm optimization (PSO) is derived from the social behaviour of, and the implicit rules adhered to by birds in a flock that enable them move synchronously without colliding [17]. The belief that social sharing of information by members of a population may provide an evolutionary advantage was the basic idea behind the development of PSO [18]. Naturally, our problems are sometimes solved by our interaction one with another. Our interaction produces socio-cognitive experience which ultimately affects our behaviours and attitudes, otherwise referred to as the social and cognitive components. The cognitive component represents the particle's own experience

[1] A codon is simply a tri-nucleotide (triplets of bases - A, C, G, and U or T, typifying Adenine, Cytosine, Guanine, Uracil and Thymine, respectively) sequence that is used to identify or specify an amino acid.

as to where the best solution is, while the social component represents the belief of the entire swarm as to where the best solution is. PSO simulates this idea of a social optimization where social organisms tend to move towards the direction of optimal benefit.

The two early variants of the PSO algorithm are referred to as the *gbest* (global best) PSO and the *lbest* (local best) PSO. The particles (or a swarm of individuals) in the *gbest* PSO move toward their best previous positions and toward the best particle in the entire swarm. In the *lbest* PSO each particle moves towards its best previous positions and towards the best particle in its restricted neighbourhood [19]. The *gbest* PSO has been employed in unsupervised image classification and is considered efficient in cluster analysis in comparison to *lbest* PSO [3]. The personal best position, y_i, of particle *i* is the best position the particle has ever visited. The best position is the position that resulted in the best fitness value. Considering *f* to represent a fitness function, then, the personal best position of particle *i* at time step *t* is computed as:

$$\mathrm{y}_i(t+1) = \begin{cases} \mathrm{y}_i(t) & \text{if} \quad f(\mathrm{x}_i(t+1)) \geq f(\mathrm{y}_i(t)) \\ \mathrm{x}_i(t+1) & \text{if} \quad f(\mathrm{x}_i(t+1)) < f(\mathrm{y}_i(t)) \end{cases} \quad (1)$$

The current position of particle *i* is denoted by x_i. The velocity of particle *i* for the *lbest* PSO is calculated as in equation (3). For the *gbest* PSO, $\hat{y}_{ij} = \hat{y}_j$, for all $i = 1, \ldots, n_x$ (the total size of the swarm) where $\hat{y}_{ij}$ is the neighbourhood best position of the particle and $\hat{y}_j$ is the position of the global best particle.

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_{ij}(t) - x_{ij}(t)] \quad (2)$$

$$x_{ij}(t) = x_{ij}(t) + v_{ij}(t+1) \quad (3)$$

where $v_{ij}(t)$, $y_{ij}(t)$ and $x_{ij}(t)$ are the velocity, the personal best position and the current position, respectively, of particle *i* in an N_d-dimensional swarm, $\boldsymbol{P}$, for $j = 1, \cdots, N_d$ at time step *t*, c_1 and c_2 are positive acceleration constants used to scale the contribution of the cognitive and social components respectively, and $r_{1j}(t)$, $r_{2j}(t) \sim U(0, 1)$ are random values in the range [0,1]. Equation (3) is used to update the particle's new position at every iteration.

A. PSO Clustering Method

PSO has been used by Van der Merwe and Engelbrecht [4] to cluster sets of multidimensional data using a fitness function consisting of quantization error only. In general, the results show that the PSO-based clustering algorithm performs better than the K-means algorithm. PSO is more likely to find near-optimal solutions than K-means. This is because, whereas PSO is less sensitive to the effect of the initial conditions owing to its population-based nature, K-means, as a greedy algorithm, depends on the initial conditions.

PSO-based clustering has also been used by Omran [3] in the clustering of image pixels. In his work, several versions of PSO were examined. The *gbest* PSO was found to outperform most of the other versions on most data sets.

Tillett *et al.* [20] employed PSO in the clustering of sensors in a sensor network. When the PSO technique was tested against random search and simulated annealing, it was found to be more robust.

PSO has also been applied in document clustering [21]. Cui *et al.* demonstrated that the hybrid PSO algorithm employed in the task of document clustering was able to generate more compact clusters in comparison to the K-means algorithm.

Gene clustering was done by Xiao *et al.* [22] by proposing the application of Self-Organizing Map (SOM) and PSO. SOM and PSO were applied independently in gene clustering. The result obtained when both methods were used was better than when the the individual methods were used.

B. PSO-based Clustering Algorithm

In this paper PSO-based clustering is employed in the clustering of nucleic acid sequence data, with minor modifications on the data type. Several nucleotides combine to form a nucleic acid sequence which are referred to in this paper as patterns. Each sequence represents a particle (a candidate solution) in the swarm. Patterns identify particles, and a single particle represents the cluster centroids in the individual clusters. To measure the fitness of each particle, Equation (4) was used.

$$f(\mathrm{s}_i) = w_1 Int_{max}(\mathrm{s}_i) + w_2(n_{max} - Int_{min}(\mathrm{s}_i)) \quad (4)$$

where Int_{max} and Int_{min} are respectively the intra and inter-cluster distances, w_1 and w_2 are user-defined constants used respectively to specify the weight that

influences how much the intra and the inter-cluster distances will contribute to the final fitness, and n_{max} is the maximum value in the data set (between 0 and 5 in this paper, i.e. 4). The intra and inter-cluster distances are measured by calculating the maximum and minimum average distance within and between the clusters, respectively [3], and are given as

$$Int_{max}(s_i) = \max_{k=1,\cdots,K} \left\{ \sum_{\forall s_i \in S_k} d(s_i, c_k)/m_k \right\} \quad (5)$$

and

$$Int_{min}(s_i) = \min_{\forall k, k+1} \{d(c_k, c_k + 1)\} \quad (6)$$

where S_k is the k^{th} cluster, s_i is the i^{th} sequence in cluster S_k, c_k is the centroid of S_k, m_k is the number of sequences in S_k, and K is the number of clusters formed for the clustering problem. The notation $d(x,y)$ is used in equations (5), (6) and (7) to denote the distance between the properties x and y. Quantization error function is employed to determine the quality of the clustering and is defined as:

$$Q_e = \frac{\sum_{k=1}^{K} \frac{\left[\sum_{\forall s_i \in S_k} d(s_i, c_k)\right]}{m_k}}{K} \quad (7)$$

In summary, the PSO clustering algorithm is given in

Figure1.

Initialize each sequence to contain c_k cluster centroids;
  for t =1 to I_{max} do
    for each sequence (s_i)
      (i) calculate the distance, $d(s_i, c_k)$ for all clusters c_k –centroid of cluster S_k
      (ii) allocate sequence si to cluster S_k for $d(s_i, c_k)$ = min □ $k=1,\cdots,K$ $\{d(s_i, c_k)\}$
      (iii)calculate fitness using equation(4)

    Update the *pbest* position and the *gbest* solution.

Fig. 1. The PSO clustering algorithm

VII. DISTANCE MEASURES

This section examines distance/similarity measures employed in this paper in the clustering of nucleic acid sequences. Most clustering tasks are performed based on some similarity or dissimilarity measures. Distance or similarity measures are mathematical representations of closeness or similarity. The selection of distance measures for clustering is an important task. This is because it has the ability to influence the shape of the clusters, as some patterns may be close to one another according to one distance measure and farther away according to another. This was observed in the under-listed distance measures.

A. *Euclidean Distance*

The most widely-used distance measures are the Euclidean distance and the squared Euclidean distance. The Minkowski metric from which the Euclidean distance is derived, is defined as

$$d(s_u, s_v) = \sqrt{\sum_{i=1}^{N_d} (s_{u,i} - s_{v,i})^{\beta}} \quad (8)$$

The Euclidean distance is a special case of the Minkowski metric where β = 2 [23]. The Euclidean distance tends to form hyper-spherical clusters [23]. The squared Euclidean distance metric uses the same equation as the Euclidean distance metric, but without the square root. This makes clustering with the squared Euclidean distance metric faster than with the regular Euclidean distance.

B. *Edit Distance*

The edit distance (also called the Levenshtein distance) is another distance measure developed by Levenshtein [24], and employed in sequence similarity search. The edit distance is a generalization of the Hamming distance. It is used in DNA sequence analysis, plagiarism detection, speech recognition, and spell checking [25].The edit distance is the minimum number of edit operations (insertions, deletions and substitutions) needed to transform one

sequence into another. For two sequences $S_1[1..i]$, and $S_2[1..j]$ the edit distance (ED) between S1and S2 (denoted by $d(i, j)$) is defined as

$$d(i,j) = min[d(i-1,j)+1, d(i,j-1)+1, d(i-1,j-1)+t(i,j)] \quad (9)$$

$$where \quad t(i,j) = \begin{cases} 1 & \text{if } S_1(i) \neq S_2(j) \\ 0 & \text{if } S_1(i) = S_2(j) \end{cases} \quad (10)$$

The value $d(i,j)$ is, therefore, the minimum edit operations needed to transform the first i characters of S_1 into the first j characters of S_2. Using the algorithm in Figure 2, the edit distance $d(I,j)$ is calculated using a bottom-up dynamic programming approach as is common to most string algorithms [26]. From the algorithm, if the lengths of S_1 and S_2 are denoted by n and m, respectively, the edit distance between the two sequences is the value $d(n,m)$, obtained by computing $d(i,j)$ for all combinations of i and j, for $0 \leq i \leq n$ and $0 \leq j \leq m$.

The edit distance is simple and easy to implement. However, it has the following disadvantages:

- The edit distance has an order of mn time and space complexity ($O(mn)$), which makes it rather too slow when the dataset is large.
- It parallelizes poorly as a result of large data dependencies.

```
int ED(char s[1..m], char t[1..n])
    declare  int d[0..m, 0..n]      // d is a table with m+1
rows
                                    //and n+1 columns
    for i = 0,…,m  do
      d[i, 0] = i
    endfor
    for j = 0,…,n  do
      d[0, j] := j
    endfor
     for i = 1,…,m do
      for j = 1,…, n do
        if s[i] = t[j] then
          cost = 0
        else
          cost = 1
          // deletion, insertion and substitution
          d[i, j] = minimum(d[i-1, j] + 1, d[i, j-1] + 1, d[i-
1, j-1]
              + cost)
         endif
      endfor
```

Fig. 2. Edit Distance Algorithm.

C. *The Codon-based Scoring Method*

The codon-based scoring method (COBASM) [27], [5] takes an entire source sequence and compares each character with the target the same way the edit distance does. However, instead of scoring mismatches, COBASM scores a match. Where there are matches, between the characters compared, COBASM scores 1 per character and 0 otherwise. If there are consecutive blocks of three characters that are similar, an additional 1 is added to the score. This procedure continues until all the characters are compared. In other to capture all the codons in the target sequence, COBASM continues the search on the second position in the target sequence. The idea is to capture the principle governing the construction of the codon table used in the formation of the twenty amino acids found in protein.

Nucleic acid (DNA/RNA) sequences are only considered similar if the percentage similarity is 70% [13]. Therefore, the value obtained from COBASM must be up to 70% the entire length of the source sequence before it could be considered a member of the cluster. The algorithm is given in Figure 3.

A contiguous collection of nucleotide symbols is what is referred to as sequence. The symbols are A, C, G, T in DNA, and a replacement of T with U in RNA. In sequence clustering, data are represented in symbolic form and need to be converted to numeric form to implement PSO. To achieve this, the nucleotides are assigned values to convert them to numeric as follows: A=1, C=2, G=3, U=T=4. The resultant sequence data can be interpreted to mean a series of events that are separated by intervals. A symbol (now represented in numeric form) is regarded as an event and a comma (,) an interval. An event interval is, therefore, represented by a lower and an upper bound, as (1, 3, 2) with an interval between in a 3-dimensional plane, to mean AGC. A sequence of length 60 will have 60 events of 59 intervals, i.e. 60-dimensions. COBASM is simple to implement and results have proved that it is robust in the task of sequence clustering as compared to edit distance. In the experiment performed in this paper, when Euclidean distance is replaced with COBASM in PSO-based sequence clustering, the result obtained shows a significant improvement over other methods.

It is proven by Baridam [5] that COBASM satisfies

the condition for metrics. This justifies the usage of COBASM alongside other distance metrics in this paper.

D. *Manhattan Distance*

The Manhattan distance metric is defined as:

$$d(S_1, S_2) = \sum_{i=1}^{N_d} | S_{1i} - S_{2i} | \qquad (11)$$

where N_d is the number of variables, and S_{1i} and S_{2i} are the values of the i^{th} variable, at points S_1 and S_2 respectively.

The Manhattan distance is measured as the sum of the displacements along the vertical and horizontal axes. This implies that the Manhattan distance function computes the distance between points through a grid-like path. The Manhattan distance metric is poor with datasets of high dimensionality [28].

VIII. EXPERIMENTAL RESULTS

This section compares the results of applying different distance/similarity measures with the PSO clustering algorithm in the clustering of six sequence datasets. The distance measures are Euclidean distance, edit distance (ED), Manhattan distance measures and COBASM. The six datasets used were emblFasta *Rickettsia typhi str*. RNA sequences with Accession Number AE017197 from Wilmington Complete Genome of 1111500 nucleotides, Homo sapiens' *melanatonic melanoma* DNA sequences, mRNA *bos taurus* sequences from Genetic Sequence Databank with Accession Number BE484664 obtained from the work of Sonstegard, et al [29], and DNA dental sequences from Department of Microbiology, University of Pretoria, South Africa.

```
    Initialize S1 and S2;
    for | S1 |: i = 1 to n do
       for| S2 |: j = 1 to m do
//determine length of longest sequence
//if sequences are unaligned or unequal if n<m
//if length of sequences less than longest sequence
//do pattern-element-search
//Compare s1[i] with s2[j],s2[j +1], ⋯ ,s2[m−n] and
//s1[i + 1] with s2[j + 1],s2[j + 2], ⋯ ,s2[m − n + 1]

      if s1[i]= s2[j]
         score =1
      else
        score =0
      endif
      if n = m //length of sequences are equal
       if s1[i]= s2[j] //examine each character of S1 and
                       // S2
          score =1
       else
          score =0
       endif
      endif
      //split sequence S1 and S2 (including gaps if
      aligned) //into blocks of three nucleotides each
      and compare //adjacent blocks
        for i, j ≥ 0 do
            //do a total block-match
          if s1[i +1,i +2,i +3] = s2[j +1,j +2,j + 3]
             score = score +1
          endif
        endfor
      endfor
  endfor
  return score
```

Fig. 3. A pseudo-code for the codon-based scoring method

The main purpose was to compare the quality of the clusters generated by each distance measure based on

- the quantization error, Qe
- the intra-cluster distances, Int_{max} and
- the inter-cluster distances, Int_{min}. The intra-cluster and inter-cluster distances defines the degree of compactness and separability of generated clusters. For all the results obtained, averages of 30 simulations over 100 iterations are reported with standard deviations to indicate the range of values to which the distance measures converge.

TABLE I
PERFORMANCE COMPARISON

		Distance Measures	Euclidean			COBASM			ED			Manhattan		
w_1	w_2	Problem	Q_e	Int_{max}	Int_{min}	Q_e	Int_{max}	Int_{min}	Q_e	Int_{max}	Int_{min}	Q_e	Int_{max}	Int_{min}
0.5	0.5	Dataset 1	60.7738± 0.1119	9.9047 ± 0.0278	0.4989 ± 0.0838	60.7711 ± 0.1293	9.9053 ± 0.0239	0.5060 ± 0.0709	60.7631 ± 0.1051	9.8950 ± 0.0203	0.4795 ± 0.0781	1805.25 ± 0.7029	67.6081 ± 0.3647	4.1603 ± 0.8510
		Dataset 2	60.8851 ± 0.1668	9.9143 ± 0.0240	0.4845 ± 0.0653	60.8474 ± 0.1508	9.9205 ± 0.03194	0.4855± 0.0704	60.8616 ± 0.1299	9.9248 ± 0.0272	0.5003 ± 0.0814	1805.844 ± 0.8187	67.3133 ± 0.2424	4.2233 ± 1.0363
		Dataset 3	60.8745 ± 0.1610	9.9680 ± 0.0311	0.5181 ± 0.0948	35.0578 ± 0.2023	10.0075 ± 0.0239	0.5412 ± 0.10732	35.0372 ± 0.1786	10.0047 ± 0.0315	0.5222 ± 0.0763	602.6902 ± 1.6594	68.4156 ± 0.3382	4.5094 ± 0.9256
		Dataset 4	70.2717 ± 0.3676	19.2598 ± 0.3117	3.44001 ± 0.2171	21.70064 ± 0.1978	8.5085 ± 0.1313	1.5938 ± 0.1134	70.3538 ± 0.3260	19.1593 ± 0.3516	3.3959 ± 0.2051	2414.627 ± 5.4514	251.6287 ± 4.1417	43.9289 ± 2.5739
		Dataset 5	51.674 ± 0.4349	13.0211 ± 0.0773	1.8588 ± 0.1783	51.7583 ± 0.2635	13.0188 ± 0.0911	1.8349 ± 0.1578	51.7625 ± 0.2944	12.9959 ± 0.1176	1.8394 ± 0.1907	1339.026 ± 3.2458	127.7245 ± 0.68214	18.4992 ± 1.6066
		Dataset 6	63.0623 ± 0.1903	16.0823 ± 0.0954	2.7899 ± 0.1722	20.0152 ± 0.0447	7.3233 ± 0.0415	1.3600 ± 0.0742	63.0957 ± 0.1330	15.2276 ± 3.3506	2.8019 ± 0.1597	1981.613 ± 2.1157	194.9271 ± 0.8404	33.5833 ± 2.7209
0.3	0.7	Dataset 1	80.1160± 0.1316	9.8988±0 .0176	0.4898±0 .0761	24.7662± 0.0685	5.2623± 0.0205	0.3146± 0.0521	80.1315 ± 0.1390	9.8997± 0.0283	0.4981± 0.0734	2496.4925± 0.7261	67.5464± 0.3477	4.0260± 0.7811
		Dataset 2	80.2008 ± 0.1311	9.9229 ± 0.0283	0.5034 ± 0.0642	24.8005 ± 0.0770	5.2756 ± 0.0198	0.3213 ± 0.0478	80.1883 ± 0.1215	9.9280 ± 0.0273	0.5016 ± 0.0770	2496.8919 ± 0.7808	67.2957 ± 0.2215	4.1312 ± 0.6747
		Dataset 3	80.2032± 0.1560	9.9596±0 .0340	0.4829±0 .0723	16.0987 ± 0.0981	5.3038 ± 0.0183	0.3326 ± 0.0531	43.8821 ± 0.2016	9.9917 ± 0.0285	0.5266 ± 0.0996	811.2010 ± 1.4106	68.4135 ± 0.3949	4.5986 ± 0.7821
		Dataset 4	87.3926 ± 0.3454	19.1561 ± 0.2502	3.4159 ± 0.1781	25.3778 ± 0.1810	8.5378 ± 0.1249	1.6231 ± 0.1017	87.2947 ± 0.4573	19.1415 ± 0.3122	3.4773 ± 0.2303	3239.9415 ± 6.0238	251.0483 ± 3.1136	43.7564 ± 3.5086
		Dataset 5	65.0743± 0.2348	12.9746± 0.0819	1.8036±0 .1530	20.7434± 0.1157	6.2959 ±0.0560	0.9818± 0.0952	65.0130 ± 0.3162	12.9988 ± 0.0871	1.8346 ± 0.1553	1806.5475 ± 3.1131	127.6100 ± 0.8693	18.9557 ± 1.7553
		Dataset 6	79.1620± 0.2545	16.0924± 0.1139	2.7670±0 .1936	23.7156± 0.0828	7.3259± 0.0499	1.3788± 0.0809	79.1556 ± 0.1849	16.0815 ± 0.1174	2.8535 ± 0.2036	2670.758 ± 2.243585	194.8186 ± 0.6397	32.6472 ± 1.8056
0.6	0.4	Dataset 1	51.0146 ± 0.1382	9.9026 ± 0.0203	0.5171 ± 0.0748	17.1488 ± 0.0708	5.2588 ± 0.0172	0.3196 ± 0.0465	51.0587 ± 0.1179	9.9067 ± 0.0277	0.4998 ± 0.1074	1459.3374 ± 0.8651	67.5299 ± 0.3152	4.2890 ± 0.7797
		Dataset 2	51.0948 ± 0.1122	9.9330 ± 0.0382	0.5035 ± 0.0827	17.1679 ± 0.0803	5.2796 ± 0.0266	0.3004 ± 0.0351	51.0402 ± 0.1326	9.9319 ± 0.0278	0.4790 ± 0.0610	1460.0931 ± 0.8520	67.2572 ± 0.2477	4.4126 ± 0.9876
		Dataset 3	51.0897 ± 0.1278	9.3274 ± 2.3936	0.4880 ± 0.0794	12.1859 ± 0.1494	5.3044 ± 0.0244	0.3150 ± 0.0389	30.4096 ± 0.1994	9.9982 ± 0.0291	0.5250 ± 0.0720	497.9529 ± 1.3166	68.4840 ± 0.3083	4.5842 ± 0.9417
		Dataset 4	61.4176 ± 0.6157	18.6049 ± 2.8011	3.4621 ± 0.2102	19.4668 ± 0.3388	8.5053 ± 0.1143	1.6411± 0.0966	61.4809 ± 0.6111	19.1474 ± 0.3189	3.4562 ± 0.1705	1996.3942 ± 8.6544	251.6544 ± 4.0079	43.9666 ± 2.0084
		Dataset 5	44.4239 ± 0.4914	12.9875 ± 0.1013	1.8116 ± 0.1711	15.4373 ± 0.3786	6.2937 ± 0.0443	0.9718 ± 0.0750	44.5291 ± 0.6085	13.0154 ± 0.0992	1.8082 ± 0.1415	1101.5271 ± 3.7201	127.8871 ± 0.7442	19.1975 ± 1.9563
		Dataset 6	54.8362 ± 0.2050	16.0966 ± 0.0925	2.7983 ± 0.1512	18.0446 ± 0.1137	7.3300 ± 0.0418	1.3462 ± 0.0742	54.8123 ± 0.2136	16.0604 ± 0.0974	2.7936 ± 0.1530	1635.0039 ± 2.6378	195.0844 ± 0.8064	32.6830 ± 2.4751
0.8	0.2	Dataset 1	31.1953 ± 0.0666	9.9017 ± 0.0246	0.4988 ± 0.0883	11.6264 ± 0.1096	5.2676 ± 0.0182	0.3116 ± 0.0404	31.2735 ± 0.0751	9.9267 ± 0.0250	0.5520 ± 0.0577	766.6623 ± 0.4321	67.5238 ± 0.2543	4.5600 ± 0.8648
		Dataset 2	31.3024 ± 0.0696	9.9255 ± 0.0389	0.4924 ± 0.0680	11.7404 ± 0.0642	5.2744 ± 0.0174	0.3100 ± 0.0525	31.3774 ± 0.0781	9.9712 ± 0.0331	0.5537 ± 0.0742	767.2099 ± 0.5314	67.2750 ± 0.2301	4.2123 ± 0.7835
		Dataset 3	31.3331 ± 0.0998	9.9642 ± 0.0299	0.4942 ± 0.0839	8.8950 ± 0.2581	5.2975 ± 0.0122	0.3384 ± 0.0556	21.1077 ± 0.2148	10.0434 ± 0.0357	0.6397 ± 0.1020	286.9121 ± 0.8161	83.0324 ± 55.1245	4.3988 ± 0.8416
		Dataset 4	41.4626 ± 0.5637	19.1772± 0.2348	3.4206 ± 0.2240	14.3894 ± 0.1574	8.5062 ± 0.1123	1.5966 ± 0.0682	41.6926 ± 0.4088	19.5468 ± 0.3398	3.7297 ± 0.2301	1141.0030 ± 8.4909	250.9030 ± 4.4503	43.9669 ± 3.2781
		Dataset 5	29.6030 ± 0.4513	13.0043 ± 0.0979	1.8455± 0.1939	11.2893± 0.3201	6.2762± 0.0424	0.9683± 0.0682	30.1591 ± 0.5093	13.1051 ± 0.1042	1.9950± 0.1557	624.3768±3 .1370	127.6209 ± 0.8149	18.6676± 1.8371
		Dataset 6	37.1314 ± 0.1813	16.1090± 0.0784	2.8200± 0.1873	13.2710± 0.2332	7.3424± 0.0375	1.3593± 0.0965	37.2932 ± 0.1562	16.1762 ± 0.0927	2.9297± 0.1773	932.6633± 1.2619	194.9600 ±0.6258	33.4586± 2.5571
0.1	0.9	Dataset 1	99.4764 ± 0.1563	9.9127 ± 0.0312	0.5003 ± 0.0883	29.7737 ± 0.0838	5.2648 ± 0.0188	0.3122 ± 0.0354	99.3712 ± 0.1378	9.9013 ± 0.0281	0.4861 ± 0.0691	3187.3690 ± 1.1334	67.5087 ± 0.1910	4.3666 ± 0.7901
		Dataset 2	99.4596 ± 0.1275	9.9264 ± 0.0341	0.4826 ± 0.0617	29.7849 ± 0.0991	5.2767 ± 0.0170	0.3193 ± 0.0442	99.4929 ± 0.1579	9.9296 ± 0.0279	0.5284 ± 0.0775	3187.5269 ± 0.7328	67.2612 ± 0.2134	4.4597 ± 0.9502
		Dataset 3	99.3642 ± 0.2167	9.9533 ± 0.0279	0.5137 ± 0.0675	18.4851 ± 0.1106	5.2996 ± 0.0141	0.3341 ± 0.0472	52.7816 ± 0.2197	10.0027 ± 0.0283	0.5111 ± 0.0803	1019.5872 ± 1.8585	68.3847 ± 0.3320	4.3752 ± 0.8745
		Dataset 4	104.1556 ± 0.6565	19.1808 ± 0.3053	3.5284 ± 0.2281	28.9889 ± 0.1879	8.5370 ± 0.1749	1.6235 ± 0.1033	104.252 ± 0.4949	19.1772 ± 0.2980	3.5070 ± 0.2192	4063.8071 ± 5.0419	251.7903 ± 4.5413	43.9224 ± 2.6434
		Dataset 5	78.0838 ± 0.4096	12.9945 ± 0.0957	1.8189 ± 0.1920	23.9722 ± 0.1715	6.2998 ± 0.0545	0.9785 ± 0.0950	78.2283 ± 0.2550	12.9976 ± 0.1108	1.8778 ± 0.1578	2270.5756 ± 3.3577	127.6384 ± 0.8054	18.7456 ± 1.6993
		Dataset 6	95.2169 ± 0.2659	16.0942 ± 0.0935	2.8167 ± 0.1917	27.3380 ± 0.1222	7.3271 ± 0.0444	1.3764 ± 0.0755	95.2532 ± 0.2699	16.0879 ± 0.0918	2.8065± 0.2053	3357.6494 ± 3.0733	194.9610 ± 0.8436	34.1684 ± 2.8350

The following data sets, of varying complexities, were employed.

- Dataset 1: 500 *Rickettsia typhi str*. RNA sequences consisting of 30000 nucleotides.
- Dataset 2: 200 *Rickettsia typhi str*. RNA sequences consisting of 12000 nucleotides.
- Dataset 3: 100 *Rickettsia typhi str*. RNA sequences consisting of 6000 nucleotides.
- Dataset 4: 31 DNA dental sequences of varying lengths consisting of approximately 12550 nucleotides.

- Dataset 5: 20 Homo sapiens' *melanatonic melanoma* DNA sequences of varying lengths and a total of 15658 nucleotides with the longest sequence having 1471, and the shortest 134 nucleotides long.
- Dataset 6: 141 mRNA *bos taurus* sequences of 29718 nucleotides with the longest sequence having 508, and the shortest 198 nucleotides long. Accession date: June 15, 2008.

Table 1 summarizes the results obtained for each of the four distance measures.

Investigations of the influence of sub-objective weights on the intra-and inter-cluster distances on the final fitness were done. To determine the quality of clusters generated using Equation (4), weights were employed as follows: w_1 = 0.5, 0.6, 0.3, 0.8, 0.1 and w_2 = 0.5, 0.4, 0.7, 0.2, 0.9, respectively. The values are chosen to ensure sum of the weights (w_1 and w_2) equals 1.0. The final results obtained from this parametric clustering are very much dependent on the number of iterations, hence the results in Table I.

The results obtained show some remarkable improvement in quality, compactness and separability of clusters generated with COBASM on virtually all the datasets as indicated by the values generated in Table 1. The performance of PSO when the other distance measures were employed also showed some significant results. This shows the robustness of the PSO-based sequence clustering. However, it was observed that Manhattan distance performed very poorly in all cases. This confirms that Manhattan distance measure is poor in the handling of high dimensional data [28].

For Dataset 1, the quality of clusters generated improved from 60.7711 with w_1 = w_2 =0.5 to 24.7662 with the weights set to 0.3 and 0.7, respectively with COBASM. The quality further improved with the weights set to 0.6 and 0.4, respectively with all the distance measures. A significant result was obtained when the weights were set to 0.8 and 0.2, respectively. The results, again, became poor with the weights set to 0.1 and 0.9, respectively. From these results, it is clear that an increase in the value of w_1 produced better quality of generated clusters. These trends were observed on all the other datasets. The results obtained further demonstrate that numeric-based distance measures do not produce best clustering results on nucleic acid sequences.

IX. CONCLUSION AND FURTHER RESEARCH

This paper investigated the performance of PSO-based clustering method as applied to the clustering of nucleic acid sequences by introducing distance measures. The performances of the three distance measures namely edit distance, Manhattan distance and COBASM were examined alongside Euclidean distance, as they were applied in the clustering of the high-dimensional problems. Several sub-objective weights were used to observe the robustness of the method. PSO was found to perform best when COBASM was introduced in the clustering problem. The performance was evaluated based on the quality, compactness and separability of formed clusters. The results demonstrate that numeric-based distance measures are not capable of producing quality clusters on nucleic acid sequences.

This work can be extended by applying PSO with the codon-based scoring method in the clustering of amino acids (protein) sequences. In the experiment conducted in this paper, multi-dimensional problems were avoided by truncating the sequences to the nearest available dimension that could be handled by PSO clustering functions. An extension to multi-dimensional problems will be a novel contribution.

REFERENCES

[1] M. Kirsten, S. Wrabel, and T. Horvath, "Distance-based approaches to relational learning and clustering," in Relational data mining. New York, Inc.: Springer-Verlag, 2000, pp. 213–230.

[2] J. Yang and W. Wang, "CLUSEQ: e□cient and e□ective sequence clustering," in Proceeding of 19th International Conference Data Engineering, Mar. 2003, pp. 101–1125.
[3] M. G. H. Omran, "Particle swarm optimization methods for pattern recognition and image processing," PhD thesis, University of Pretoria, Faculty of Engineering, Built Environment and Information Technology, Department of Computer Science, Nov. 2004.
[4] M. Van der and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in IEEE Congress on Evolutionary Computation, Canberra, Australia, 2003, pp. 215–220.
[5] B. B. Baridam, "Block-based similarity measure and optimization techniques for nucleic acid sequence clustering," 2011.
[6] B. S. Everitt, Cluster Analysis, 3rd ed. New York: Halsted Press, 1993.
[7] E. Forgy, "Cluster analysis of multivariate data: Efficiency versus interpretability of classification," Biometrics, vol. 21, pp. 768–769, 1965.
[8] J. Bezdek, "A convergence theorem for the fuzzy ISODATA clustering algorithms," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 2, pp. 1–8, 1980.
[9] ——, Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, 1981.
[10] A. Liew, S. Leung, and W. Lau, "Fuzzy image clustering incorporating spatial continuity," in IEEE Proceedings Vision, Image and Signal Processing, vol. 147, no. 2, 2000.
[11] Y. Leung, J. Zhang, and Z. Xu, "Clustering by space-space filtering," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 12, pp. 1396–1410, 2000.
[12] G. Hamerly and C. Elkan, "Alternatives to the k-means algorithm that find better clusterings," in Proceedings of the ACM Conference on Information and Knowledge Management(CIKM-2002), 2002, pp. 600–607.
[13] J. Claverie and C. Notredame, Bioinformatics for dummies, 2nd ed. Indiana: Wiley, 2007.
[14] S. Muthukrishnan and S. C. Sahinalp, "Approximate nearest neighbors and sequence comparison with block operations," in Proceeding of the thirty-second annual ACM symposium on Theory of computing, 2000, pp. 416–424.
[15] G. Cormode and S. Muthukrishnan, "The string matching problem with moves," ACM Transactions on Algorithms, vol. 3, no. 1, Feb. 2007.
[16] D.LoprestiandA.Tomkins,"Block edit models for approximate string matching," Theoretical Computer Science, vol. 181, pp. 159–179, 1997.
[17] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in Proceedings of Sixth Symposium on Micro Machine and Human Science. IEEE Service Center, Piscataway, NJ, 1995, pp. 39–43.
[18] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," Natural Computing, vol. 1, pp. 235–306, 2002.
[19] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization: An overview," Swarm Intelligence, vol. 1, pp. 33–57, 2007, available online:[doi:10.1007/s11721-007-0002-0].
[20] J. C. Tillett, R. M. Rao, F. Sahin, and T. M. Rao, "Particle swarm optimization for clustering of wireless sensors," in Proceeding of the Society of Photo-Optical Instrumentation Engineers, vol. 5100, no. 73, 2003.
[21] X. Cui, P. Palathingal, and T. E. Potok, "Document clustering using particle swarm optimization," in IEEE Swarm Intelligence Symposium, Pasadena, California, 2005, pp. 185–191.
[22] X. Xiao, E. Dow, R. Eberhart, B. Z. Miled, and R. Oppelt, "Gene clustering using self-organizing maps and particle swarm optimization," in Proceeding of Second IEEE International-Workshop on High Performance Computational Biology, Nice, France, 2003.
[23] R. Xu and D. Wunsch II, "Survey of clustering algorithms," IEEE Transactions on Neural Networks, vol. 16, no. 3, pp. 601– 614, May 2005.
[24] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," Doklady Akademii Nauk SSSR, vol. 163, no. 4, pp. 845–848, Jan. 1965.
[25] M. Gilleland, "Levenshtein distance, in three flavours," Merriam Park Software, Available online: [www.merriampark.com/ld.htm], Tech. Rep.
[26] D. Gusfield, Algorithms on strings, trees and sequences: Computer Science and computational Biology. Cambridge, UK: Cambridge University Press, 1997.
[27] B. B. Baridam and O. O. Owolabi, "Conceptual clustering of RNA sequences with codon usage model," Global Journal of Computer Science and Technology, vol. 10, no. 8, pp. 41–45, 2010.
[28] G.Hamerly, "Learning structure and concepts in data using data clustering," PhD thesis, University of California, San Diego, 2003.
[29] T. Sonstegard, A. V. Capuco, J. White, C. P. Van Tastell, E. E. Connor, J. Cho, R. Sultana, L. Shade, J. E. Wray, K. D. Wells, and J.Quackenbush, "Analysis of bovine mammary gland EST and functional annotation of the Bos Taurus gene index," Mammary Genome, vol. 13, no. 7, pp. 373–379, 2002.