Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 14 (5): 426-435, 2014
ISSN 1812-5654 / DOI: 10.3923/jas.2014.426.435

Swap Mechanism for Medical Clustering Problems

[1]Anmar Abuhamdah, [1]Rawnaq Kittaneh, [1]Jawad Alkhatib and [2]Mohd Zakree Ahmad Nazri
[1]Department of Computer Science, College of Computer Science and Engineering,
Taibah University, Madinah, KSA
[2]Data Mining and Optimisation Research Group (DMO),
Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia,
43600, UKM, Bangi, Selangor, Malaysia

Abstract: Clustering problem is an important area of data mining. The goal of the clustering is to partitioning N objects into K clusters with similar characteristics for one cluster and dissimilar to other clusters by using any algorithm to generate the initial clusters partition. This initial clusters quality will be measured based on a certain distance (or object) function. The initial partitioning can be improve by iteratively explores its neighbor solutions looking for a better one by any algorithm. The neighbor's solutions are achieved by reallocating some patters (objects) by using any neighborhoods structures. However, the way of employing the neighborhoods structures will affects the solution quality. Therefore, in this work, a swap mechanism is proposed to increase the diversification for the neighborhood structure. The proposed mechanism tested on two algorithms previously implemented over six medical clustering problem (that are available in UCI Machine Learning Repository) by using two way of the distance calculation. This work, also present a third distance calculation to minimize the distances between the clusters. Experimental results show that, using the swap mechanism and the third distance function on the algorithms is able to produce significantly better quality solutions than not using it on all datasets.

Key words: Clustering, multi k-means, minimal distance, swap mechanism, medical clustering problems

INTRODUCTION

Clustering is the process of grouping a set of physical data (objects) into classes of similar data, which is consider as unsupervised data mining technique (Brucker, 1978; Jain *et al*., 1999). Clustering is dividing the input space based on some similarity measures to number of clusters (Saha *et al*., 2010). A cluster is a collection of similar objects within the same cluster that can be treated as one group and dissimilar to the objects in other clusters (groups) (Brucker, 1978; Saha *et al*., 2010). Clustering is considered as NP-hard problem, when the number of clusters exceeds three clusters (Saha *et al*., 2010) or two clusters (Dasgupta and Freund, 2009; Mahajan *et al*., 2009). All clusters are categorized depending on a certain distance function (that calculate the minimal distance) used to achieve the similarity/dissimilarity condition of data representation, so that the cluster contains most "similar" objects and different clusters enclose "dissimilar" patterns in terms of the data set attributes (Saha *et al*., 2010). Clustering approaches are used to partition the objects into clusters (Holland, 1975) to generate an initial clusters partition (or initial solution) such as K-Means approach and Multi K-Means approach (Davidson and Satyanarayana, 2003), the Fuzzy C-Means approach (Hong, 2006) and many more. For further information please refer to (Berry and Linoff, 1997).

Clusters quality measured by calculating the minimal distance (that calculates based on the distance function) quality (Saha *et al*., 2010). The smaller value overall the minimal distance values indicates better clustering quality (Hong, 2006). Finding a good clustering (solution) quality depends on the methodology used and the problem representation employed during the search (Jain *et al*., 1999). Optimization algorithms are used in clustering problems in order to minimize the minimal distance quality (Jain *et al*., 1999), which starts with an initial clusters partition generation and iteratively explores its neighbor solutions (partition), seeking a better one. Recently, there are many optimization algorithms were applied to improve the current solution in order to minimize the differences (or minimal distances) within the same cluster and to maximize the differences between the set of clusters, such

Corresponding Author: Anmar Abuhamdah, Department of Computer Science, College of Computer Science and Engineering, Taibah University, Madinah, KSA

as Tabu Search (Liu *et al*., 2008), Artificial bee colony (Zhang *et al*., 2010) and many more. However, these algorithms are employing neighborhood structures, which reallocate the neighbors on the clusters and affect the solution quality (Hong, 2006; Liu *et al*., 2008).

Therefore, in this study, we propose a swap mechanism as a neighborhood structure to increase the diversification of the neighborhood structures, which may help in improving the solutions (or minimal distances) qualities. In order to evaluate the performance of the swap mechanism, the swap mechanism is embedded with two algorithms previously implemented for the medical clustering problem, to compare their performances. The algorithms are the Modified Great Deluge algorithm (MGD) (Abuhamdah, 2012) and hybridize Modified Great Deluge and Iterative Simulated Annealing (ISA-MGD) (Abuhamdah *et al*., 2012). In both algorithms, the initial solutions (or clusters) are generated by using Multi K-Means algorithm (Davidson and Satyanarayana, 2003) and iteratively improve the current solutions by using two neighborhood structures. Both algorithms measures the solution qualities by using two ways (i.e., between objects and between centers) of the cluster distance (or minimal distance) calculation over six medical benchmark datasets clustering problem (which are available in UCI Machine Learning Repository). However, the two ways are aims to minimize distances between the same clusters and maximize the distances between the set of clusters. Where, in the between object calculation is calculates the distances between the patterns in the same cluster and maximize the dissimilarity in the other clusters, while in the between centers calculation is calculates the distances between each pattern and its center in the same cluster to group the patterns around the center and achieve similar objects and maximize the distances between patterns.

Therefore, in this study, we present a third distances function calculation by using the combination of between centers and objects to produce better clustering that minimize the distances between the patterns with similar objects. Results demonstrate that, using the swap mechanism is able to produce better cluster quality than not using it, in addition that, using the third calculation way is also produce a good cluster with better similarity.

MATERIALS AND METHODS

Problem description: In this study, six well-known benchmark datasets are used as a test domain (which are summarize in Table 1 to test the performance of the propose swap mechanism. These datasets are available in UCI machine learning repository (http://archive.ics.uci.edu/ml/ index.html). These datasets are varied in terms of the number of records and attributes to show a different complexity for the tested data. All of the information for these datasets are about the diseases that taken from a real infected patients and denoted for research purposes.

For example, Table 1 shows that Dataset 1 is Wisconsin Breast Cancer Database (B.C), Dataset 2 is Lung Cancer Database (L.C), Dataset 3 is BUPA Liver Disorders Database (B.L.D), Dataset 4 is Pima Indian Diabetes Database (P.I.D), Dataset 5 is Haberman's Survival Database (H.S) and Dataset 6 is Thyroid gland data Disease Database (T.D).

In cluster analysis, it is very important issue to evaluate the clustering results quality that is produced by a certain measure. Those measures can be used to compare solutions obtained from different algorithms and also can be used to guide some optimization search processes to find the best partitioning procedure which fits the underlying dataset (Halkidi *et al*., 2001). In this work, three different ways (i.e., between objects, or centers, or their combinations) of the distance function (or the minimal distance) calculations are used to measure the clusters and their centers. These measurement on the cluster objects and/or changes on the cluster centers have the direct effect to the cluster quality. The three ways of the distance function calculations are portrayed as follow:

Between objects: This method depends on the calculation on the distance (cluster quality) between each data pattern.

Between centers: This method depends on calculating the distance (cluster quality) using the sum of distance between each data pattern and the cluster center that it belongs to.

Table 1: Six benchmark datasets

Dataset No.	Dataset name	No. of instances	No. of attributes	Attributes type	No. of clusters used in K-Means
1	B.C	699	10	Integers	3 (i.e., 123, 240 and 363)
2	L.C	32	56	Integers	3 (i.e., 9, 13 and 10)
3	B.L.D	345	7	Integer, categorical and real	2 (i.e., 145 and 200)
4	P.I.D	768	8	Integer and real	2 (i.e., 500 and 268)
5	H.S	306	4	Integer	2 (i.e., 225 and 81)
6	T.D	215	21	Categorical and real	3 (i.e., 150, 30 and 35)

Combination of "between Centers and between Objects": This is a combination between the previous two above function. This will finally direct the cluster objects to be highly compressed on itself and at the same time it will be more distinct from the other clusters. It is the idea of running the algorithm with two objective functions rather that only one, where the first objective function is to minimize the distance between data patterns and clusters centers and the second objective function is to minimize the distances between the data patterns themselves in the same cluster.

SWAP MECHANISM

In this study, a propose swap mechanism motivated by the neighborhood selection and its effect on the solution quality (Hong, 2006; Halkidi *et al*., 2001). The main aim of proposing the swap mechanism (N3 neighborhood structure) is to improve the minimal distance quality for the clustering problem or a similar optimization problem, which can be applied to any algorithm. So the discussion below is for the swap mechanism idea and not for the algorithm process.

In clustering problems, we minimize or improve the minimal distance by explores the current solution neighbor solutions (other solution), looking for a better one by any algorithm (Hong, 2006). The neighbor solution is accomplished by restructure the current solution (or partition) using some neighborhood structures. However, the way of employing the neighborhoods structures will affects the solution quality (Hong, 2006). Therefore, in this work, a swap mechanism is propose as a neighborhood structure (i.e., N3) to increase the diversification and the neighbor solutions selection.

In the swap mechanism, we randomly select two different patterns and swap their data. Where, each cluster is divided into 3 partitions, i.e., the first 33% of the cluster patterns goes to the first partition, the second 33% of the cluster patterns goes to the second partition and the rest of the percentage goes the third partition (which is 34%). Whereas, some solution corresponding between these clusters.

The reason behind choosing that percentages is refer to the smallest medical dataset i.e., the Lung Cancer dataset out of the six test datasets. This dataset contains only 32 rows, with 56 columns of attributes, with donor recommendation to be distributed as follows; the first cluster contains 9 patterns, the second cluster contains 13 patterns and third cluster contains 10 patterns. Where 33% of the first cluster (i.e., 9 patterns) is 3 patterns. The idea of the swap mechanism is to check the solution in different level of the same cluster and try to diversify the search.

For example in Fig. 1, we have two clusters (i.e., cluster 1 and 2) and there are nine solutions obtained, as follow:

- Randomly select a pattern (R1) from the first partition (33%) in cluster 1, a pattern (R2) from the first partition (33%) in cluster 2 and then swap R1 data with R2 data, in which consider as solution-1 (S1). Repeat the same process between the first partition (33%) in cluster 1 and the second partition (33%) in cluster 2 to obtain solution-2 (S2). After that, repeat the same process between the first partition (33%) in cluster 1 and the third partition (34%) in cluster 2 to obtain solution-3 (S3)
- Repeat as in (i) between the second partition (33%) in cluster 1 and the first, second and third partitions in cluster 2 to obtain S4, S5 and S6
- Repeat as in (i) between the third partition (34%) in cluster 1 and the first, second and third partitions in cluster 2 to obtain S7, S8 and S9

Then, the best solution between S1 to S9 will be selected as candidate solution (working solution). Meanwhile, the centers of the clusters are prevent to swap, so if the random number (when we select the pattern) hit the center pattern, then we ignore it and generate another random number.

The explanation of the propose swap mechanism above prevents the swapping process that involves the center of the cluster, where the solution quality depends on the good center for each cluster. Meanwhile, we are dealing with different calculation (i.e., between object) that not take into consideration the cluster center and calculated based on distance between patterns. Therefore, a modification on the idea is done to handle the centers and object calculations by swapping the center of a cluster by any patterns of other cluster as in Fig. 2a,b. Figure 2a show that two clusters (i.e., C1 and C2) and their centers (i.e., S1 and S2) are chosen to swap their data with other patterns in the other cluster (i.e., S1 with any pattern in C2 and S2 with any pattern in C1) in case of an improvement is achieved (better minimal distance). However, after some preliminary experiments, we found out the swapping the center to other cluster sometimes is considered hard to obtain an improvement in terms of the quality of the solution. Therefore, we decided to alter the centers by internally swap the center within the cluster itself to achieve better improvement (or better minimal distance) as in Fig. 2c (i.e., S1 with any pattern in C1) and Fig. 2d (i.e., S2 with any pattern in C2), which we termed as swap mechanism (N3).

The idea of the swap mechanism is to check the solution in different level of the same cluster with different

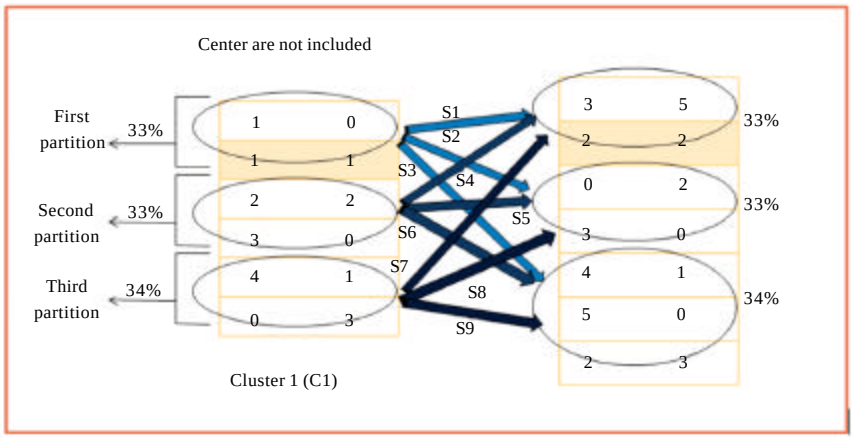

Fig. 1: N3 Swap mechanism, where the centers of the clusters are prevent to swap

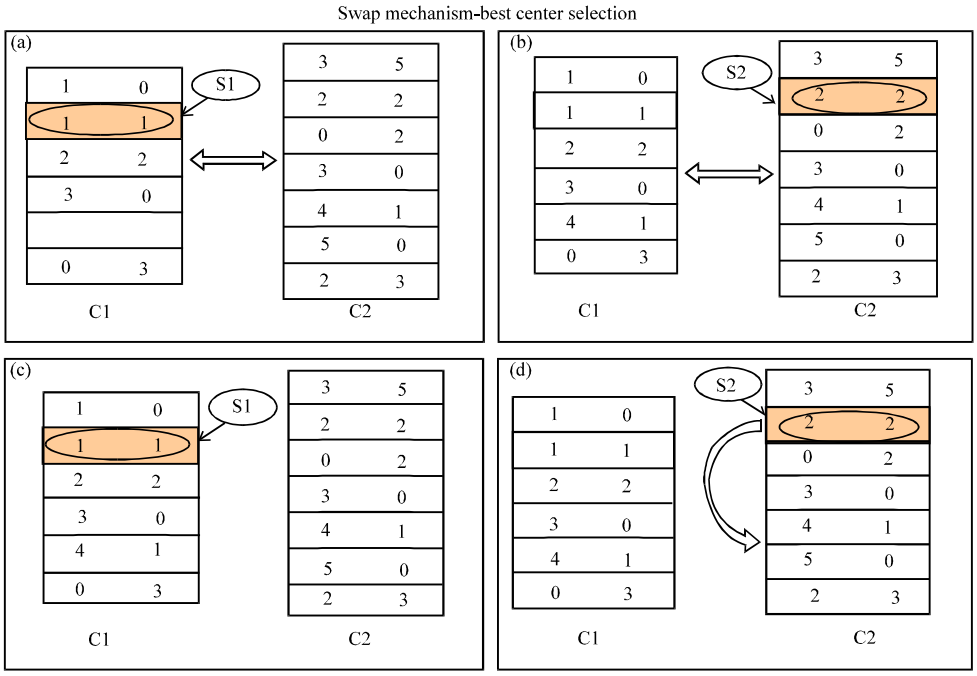

Fig. 2(a-d): Handling the centers in N3 swap mechanism, (a) Swap C1 center with any pattern in C2, (b) Swap C2 center with any pattern in C1, (c) Swap C1 center with any pattern within C1 and (d) Swap C2 center with any pattern within C2

solutions level in other cluster, which may prevent repeating the swapping with the same solution and try to evaluate the solutions in other levels. However, we can decrease the percentage of the division if we are dealing with a bigger number of rows of the smallest cluster.

In order to evaluate the performance of the proposed swap mechanism (N3), then we need to compare the results obtained by any algorithm to the results of the same algorithm using N3. Therefore, in this work to test the propose N3, two algorithms are used; the Modified Great Deluge Algorithm (MGD) (Abuhamdah, 2012) (as in Fig. 1) and the hybridize Modified Great Deluge and Iterative Simulated Annealing (ISA-MGD) (Abuhamdah *et al*., 2012) (Fig. 2), where both of them are tested over the six datasets discussed above. MGD (Abuhamdah, 2012) and ISA-MGD (Abuhamdah *et al*., 2012) algorithms use two (i.e., N1 and N2) neighborhood structures for medical clustering problems as follows:

```
Procedure modified Great Deluge Algorithm
  Step-1: Initialization Phase
   Determine initial candidate solution S_o and f(S_o);
   S_Arrange = S_o;   f(S_Arrange) = f(S_o);   S_source = S_o;   f(S_source) = f(S_o);
   Set N.iters; (stopping condition)
   Set estimated quality of final solution, est.q;
   Set not_improving_length_GD;//maximum number of GD not improved
   level = f(S_o);   // initial level
   Initialize all element in MGD list (L_MGD) = Level;
   Set L_size ;   Countr_MGD =0;
   decreasing rate â = ( ( f(S_o) - est.q ) / (N.iters) );
   Iterations=0;   not_improving_counter=0;
Step-2: Improvement (Iterative) Phase
    repeat  ( while termination condition is not satisfied)

    Step-2.1: Selecting candidate solution S_working
              Generate candidate solutions by applying neighborhood structure (N1 and
             N2) and the best solution consider as candidate solution (S_working);
    Step-2.2: Accepting Solution
          if f(S_working) < f(S_Arrange)
              S_Arrange = S_working;   f(S_Arrange)=f(S_working);
              S_source = S_working;   f(S_source)=f(S_working);
             not_improving_counter = 0;
             Countr_MGD = Countr_MGD +1;
             Index_MGD = Countr_MGD mod L_size;
             L_MGD(Index_MGD) = level;
          else
              if f(S_working) = level
                 S_source = S_working;
              else
                 Increase not_improving_counter by one;
                 if not_improving_counter ==not_improving_length_GD,
                   RN= random number between 1 and L_size;  // MGD
                   level =  L_MGD(RN) // MGD
              end if
             level = level - â;
          end if
       Iterations= Iterations+1;
    until  Iterations > N.iters (termination condition are met)
    Step-3:  Termination phase
           Return the best found solution S_Arrange
```

Fig. 3: Pseudo code for modified great deluge algorithm for medical data clustering problems (Abuhamdah, 2012)

- **N1:** Randomly selects one pattern from each cluster to swap their data as in Wang (2006)
- **N2:** Randomly select two different patterns from the same cluster and swap their data as in Wang (2006)

Both MGD and ISA-MGD starts with a given Multi K-Means partitions (Davidson and Satyanarayana, 2003) i.e., the initial solution ($S_{initial}$) is generated by K-Means algorithm (constructive method) and then iteratively improve the constructed solution by generating a neighbor solution (candidate solution) by using neighborhood structure(s). In addition, both of them are employ N1 and N2 neighborhood structure. Again, we our discussion in this study is to apply N3 over any algorithm and not discuss the algorithm process. For any further information about MGD or ISA-MGD algorithms, the please see their implementations (i.e., MGD in (Abuhamdah, 2012) and ISA-MGD in Abuhamdah *et al.* (2012).

Figure 3 shows MGD pseudo code, the algorithm starts by initializing the required parameters as in Step-1 by setting the stopping condition (N.iters), estimated quality of the final solution (est.q), the initial water level (level), decreasing rate (β), maximum number of not

```
Procedure ISA-MGD Algorithm
Step 1: Initialization Phase
   Determine initial candidate solution S_o and f(S_o);
   S_Arrange = S_o;      f(S_Arrange) = f(S_o);    S_source = S_o;      f(S_source) = f(S_o);
   Set N.iters; (stopping condition)   Set estimated quality of final solution, est.q;
   Set not_improving_length_GD;//maximum number of GD not improved
   level= f(S_o);      decreasing rate â = ( ( f(S_o) - est.q ) / (N.iters) );
   Set initial temperature T_o;   Set final temperature T_f;  Set Temp = T_o;
   Set decreasing temperature rate as á, where á = 0.7;
   Set list size, to store level values into list LL;
   Iterations=0;  not_improving_counter=0;
Step 2:  Improvement (Iterative) Phase
    repeat  ( while termination condition is not satisfied)

    Step 2.1:  Selecting candidate solution S_working
          Generate candidate solutions by applying neighborhood structure ( N1 and
           N2) and the best solution consideras candidate solution (S_working);
    Step 2.2: Accepting Solution
        if f(S_working) < f(S_Arrange)
            S_Arrange = S_working;     f(S_Arrange)=f(S_working);
            S_source = S_working;   f(S_source)=f(S_working);
            Update LL by the level value (FIFO);    not_improving_counter = 0;
        else
            ä = f(S_working) - quality (S_source)
                    Generate a random number called RN between 0 and 1;
            if  RN = e^(-ä/Temp)
                S_source = S_working;
            elseif  f (S_working) =  level
               S_source = S_working;
            else Increase not_improving_counter by one;
               if not_improving_counter ==not_improving_length_GD,
                  Temp = T_o;     Level = random level from the list LL;
            end if
           Temp = Temp-Temp*á;    level = level - â;
        end if
      Iterations= Iterations+1;
  until  Iterations > N.iters && Temp < T_f  (termination conditions are met)
     Step 3:  Termination phase
          Return the best found solution S_Arrange;
end Procedure
```

Fig. 4: Pseudo code for hybridize modified great deluge algorithm with iterative simulated annealing for medical data clustering problems (Abuhamdah *et al*., 2012)

improving solutions (not_improving_length_GD). Again, note that the initial solution is generated using K-Means (S_o). In the improvement phase (i.e., Step-2), N1and N2 neighborhood structures are applied to generate candidate solutions (i.e., 5 candidate solutions) and the best candidate is selected as the candidate solution ($S_{working}$) as shown in Step-2.1. There are two cases to accept the solution, the first case (i.e., Case 1), If f ($S_{working}$) is better than the best solution found f ($S_{Arrange}$), then $S_{working}$ is accepted as a current solution ($S_{source} \leftarrow S_{working}$) and the best solution is updated ($S_{Arrange} \leftarrow S_{working}$) as shown in Step-2.2. Then the level will be updated by the decreasing value β (i.e., level = level-β). While in case 2 (i.e., Case 2), If f ($S_{working}$) is less than f ($S_{Arrange}$), then the quality of $S_{working}$ is compared against the level. If it is less than or equal to the level, then $S_{working}$ is accepted and the current solution is updated ($S_{source} \beta S_{working}$). Otherwise, $S_{working}$ will be rejected. The level will be updated by the decreasing value β (i.e., level = level-β). The counter for the non-improving solution is increased by 1. If this counter is equal non_improving_length_GD, then the process terminates. Otherwise, the process continues the stopping condition is met (i.e., Iterations>N.iters) and return the best solution found $S_{Arrange}$. (Step-2).

From Fig. 4 show ISA-MGD pseudo code, the algorithm starts by initializing the required parameters as in Step-1 by setting the stopping condition (N.iters), the temperature (Temp) is equal to the initial temperature (T_o), initialize the decreasing temperature rate (α), initialize the estimated quality of the final solution (est.q), the initial water level (level), decreasing rate (β), maximum number of not improving solutions (not_improving_length_GD) and

a list of LL size to store the value of the level. Again, note that, the initial solution is generated using K-Means (S_o). In the improvement phase (Step-2), basically the initial solution is iteratively improved by employing the hybridization method (ISA-MGD) until the stopping condition is met. In Step-2.1, the neighborhood structures N1 and N2 are applied similar to MGD. There are two cases to be taken into account similar to MGD except that in the first case LL is updates (FIFO) by level (LL←level). The Temp will be decreased by the value α (i.e., Temp = Temp-Temp*α) and the level will be updated by the value β (i.e., level = level-β). In the other case, if $S_{working}$ accepted, then the difference between the quality of $S_{working}$ and S_{source} is calculated. A random number [0, 1], RN, is generated. If the probability (i.e., $e^{-\delta/Temp}$, where δ = f ($S_{working}$)-f(S_{source})) is less than or equal to RN) then $S_{working}$ is accepted. Otherwise, the quality of $S_{working}$ is compared against the level same as MGD. Otherwise, $S_{working}$ will be rejected. The level will be updated similar to MGD and thecounter for not improving solution is increased by 1. If this counter is equal non_improving_length_GD, then we reinitialize the temperature (Temp) equal to the initial temperature (T_o) and the level is updated by a new level that is randomly selected from the list (where the size of the list is set to 10 based on preliminary experiments). Otherwise, the process continues until the stopping conditions are met (i.e., Iterations>N.iters && Temp <T_f) and return the best solution found $S_{Arrange}$ (Step-2). Note that in this work the est.q is set to 0 and non_improving_length_GD is set to 10 (after some preliminary experiments).

The process will continue until the termination conditions are met (Iterations>N.iters && Temp <T_f) and return the best solution found so far $S_{Arrange}$ (Step-3).

RESULTS AND DISCUSSION

In this study, we ran the algorithms 20 times across 6 datasets as in MGD (Abuhamdah, 2012) and ISA-MGD (Abuhamdah *et al.*, 2012) with the same parameters setting. Also, the algorithms were programmed using Java language and were tested on a PC with a CPU of Intel dual core 1800 MHz and 2 GB RAM. In the analysis part, the terms were used as follows:

- **N-NI:** Number of not improved iterations
- **N-I:** Number of improved iterations
- **Std:** Standard deviation

Table 2 shows the results obtained based on the minimal distance calculation (i.e., between objects and between centers) for Modified Great Deluge algorithm (MGD) (Abuhamdah, 2012) and hybridize Modified Great Deluge and Iterative Simulated Annealing (ISA-MGD) (Abuhamdah *et al.*, 2012) by using N1 and N2. The "Avg" represents the average results out of 20 runs. The best results "Best" are presented in bold.

Table 3 shows MGD and ISA-MGD algorithms using N3 (swap mechanism) neighborhood structure based on the minimal distance calculation (i.e., between objects and between centers). Again, the best results are presented in bold.

Table 2: Results "between objects and between centers" obtained from MGD (Abuhamdah, 2012) and ISA-MGD (Abuhamdah *et al.*, 2012) algorithms using N1 and N2 neighborhood structures on six datasets

	20 Runs-minimal distance							
	MGD between objects		ISA-MGD between objects		MGD between centers		ISA-MGD between centers	
Dataset	Best	Avg	Best	Avg	Best	Avg	Best	Avg
B.C	1937	2088.40	2090.61	2124.76	3014.72	3316.570	3007.32	3281.79
T.D	893.33	946.69	892.57	9290.28	2039.89	2047.040	2070.16	2079.52
B.L.D	5509.21	5809.55	5466.33	5631.02	10498.9	10836.970	10498.9	10969.26
L.C	159.27	161.38	157.8	1590.20	151.62	153.490	152.37	153.04
H.S	1023.96	1069.57	1014.05	1043.21	2721.36	2721.594	2721.36	2730.16
P.I.D	23281.12	24239.71	22919.27	24118.18	48909.2	56159.740	48909.2	57098.70

Table 3: Results analysis "between objects and between centers" for MGD and ISA-MGD algorithms using N3 neighborhood structure on six datasets

	20 Runs-minimal distance							
	MGD between objects		ISA-MGD between objects		MGD between centers		ISA-MGD between centers	
Dataset	Best	Avg	Best	Avg	Best	Avg	Best	Avg
B.C	2019.83	2084.28	2014.71	2062.46	3007.32	3281.24	3007.32	3250. 7 5
T.D	844.74	892.63	842.55	887.01	1998.44	1998.68	1998.44	1998.44
B.L.D	5346.36	5578.24	5300.64	5440.04	10493.95	10683.24	10493.95	10785.90
L.C	157.92	159.05	157.8	158.85	151.62	151.65	151.62	151.65
H.S	966.63	1012.69	947.64	979.50	2702.34	2702.55	2702.34	2702.46
P.I.D	21025.33	22229.00	20804.16	21608.88	48769.89	48780.64	48769.89	48769.90

Table 4: Results analysis "between objects" for ISA-MGD algorithm using N3 neighborhood structure on six datasets

Dataset description	B.C	T.D	B.L.D	L.C	H.S	P.I.D
Best	2014.71	842.55	5300.64	157.80	947.64	20804.16
Iterations for best	97599	98734	95245	12836	99557	99898
Time	01:17:5	00:23:21	00:17:20	00:35:29	00:08:06	01:09:29
N-I	64389	64724	59523	20355	34852	68242
N-NI	33407	34567	39343	79604	64435	29135
Average	2062.46	887.01	5440.04	158.85	979.51	21608.88
Result range	2014.71	842.55	5300.64	157.80	947.64	20804.16
	2122.12	926.16	5615.87	159.66	1013.35	22216.95
Iterations range	97599	88372	95245	10925	94081	99479
	99979	99940	99821	89723	99996	100000
Time range	01:16:45	00:15:30	00:15:29	00:26:08	00:06:53	00:58:42
	02:36:19	00:23:39	00:18:06	00:38:50	00:08:50	01:16:32
Std	34.63	20.97	95.14	0.51	17.04	396.53

Table 5: Results analysis combination of "between centers and between objects" for MGD and ISA-MGD algorithms using N3 neighborhood structure on six datasets

		20 Runs-minimal distance calculated as combination of "between centers and between objects"			
		MGD between objects and centers		ISA-MGD between objects and centers	
Dataset	Initial length by K-Means	Best	Avg	Best	Avg
B.C	11740.404	5049.97	5211.63	5041.76	5249.35
T.D	5638.334	2931.63	3000.13	2929.61	2966.75
B.L.D	39905.605	18970.6	20469.85	20897.07	21394.17
L.C	351.097	311.79	313.31	311.24	312.4428
H.S	6090.502	5152.66	5276.37	5569.56	5722.56
P.I.D	203278.975	100660.9	102626.77	111420.73	112718.59

Table 3 with referring to Table 2 shows that, the best results and average score using N3 neighborhood structure obtained indicate that, if MGD or ISA-MGD algorithm is using N3 neighborhood structure, then it will outperformed the same algorithm using N1 and N2 neighborhood structures in all or some datasets in both calculation way, which indicate that using N3 is better in exploring the search space than using N1 and N2, except in one dataset (i.e., B.C) GD were better in the calculation between objects. Table 3 also shows that ISA-MGD outperformed MGD between objects using N3 in all datasets, where in between centers they obtain equal results. Table 4 shows a further analysis on ISA-MGD using N3 neighborhood structure between Objects over all tested datasets. For example, the best results for H.S dataset is 947.64 that is obtained within 8 minutes 6 seconds under 99557 iterations. Meanwhile, the range for minimum and maximum iterations is in between 94081 and 99996. In most of the cases, the results are obtained between 6 min 53 sec to 8 min 50 sec that are considered acceptable. Figure 5 shows a 3D scatter graph for K-Means, MGD algorithms using N3 neighborhood structure over H.S dataset. Two clusters are represented by two colors. Figure 5a shows that, the two clusters (colors) are mixed with initial minimal distance "between centers" obtained by K-Means is 3626.530 as shown in Fig. 5a-c shows that, the initial minimal distance "between objects" obtained by K-Means is 2463.972. Whereas, Fig. 5b shows that, there is an improvement in terms of the minimal distance of between objects which is equal to 966.63 and between centers 2702.34 in Fig. 5d after employing the MGD algorithm. Meanwhile, MGD results obtain by referring to Fig. 5, we can see that the dispersion of the colors in MGD is less (more concentrated) shows it is better clustered than MGD in Abuhamdah (2012) (which is more scattered).

Since, ISA-MGD and MGD algorithm using N3 outperformed ISA-MGD and MGD using N1 and N2 neighborhood structures in both calculation (i.e., between objects and between centers) as shown in Table 2 and 3, then Table 5 shows MGD and ISA-MGD algorithms using N3 neighborhood structures based on the minimal distance calculation (i.e., Combination of "between Centers and between Objects"). Again, the best results are presented in bold.

Table 5 shows, the best results and average score using N3 neighborhood structure for MGD and ISA-MGD algorithms in all datasets for the combination of "between Centers and between Objects" calculation way. The best minimal distance obtained in Table 5 shows that the results obtained by MGD algorithm has outperformed ISA-MGD algorithm in three datasets (i.e., BL.D, H.S and P.I.D datasets), whereas, ISA-MGD algorithm has outperformed MGD algorithm in the other three datasets (i.e., B.C, T.D and L.C datasets).

Table 6 shows a further analysis on ISA-MGD using N3 neighborhood structure for the combination of "between Centers and between Objects" over all tested

Table 6: Results analysis for the combination of "between centers and between objects" for ISA-MGD algorithm using N3 neighborhood structure on six datasets

Dataset description	B.C	T.D	B.L.D	L.C	H.S	P.I.D
Best	5041.76	2929.61	20897.07	311.2408	5569.56	111420.73
Iterations for best	99996	76436	23271	41083	1227	59446
Time	03:06:12	00:19:52	00:10:06	0:40:14	00:08:06	01:05:35
N-I	57135	39667	96962	20814	88072	90802
N-NI	40571	59767	2574	79126	11876	8826
Average	5249.35	2966.75	21394.17	312.44	5722.56	112718.59
Result range	5041.76	2929.61	20897.07	311.24	5569.56	111420.73
	5425.82	3001.97	21895.99	314.11	5843.56	
Iterations range	98242	68577	11695	15069	53	17567
	99996	99948	95654	98222	34534	99654
Time range	3:04:07	00:19:35	00:10:06	0:34:58	00:08:03	00:43:17
	03:26:25	00:20:40	00:14:01	1:07:28	00:13:40	01:05:52
Std	165.63	22.23	276.31	0.69	77.29	607.46

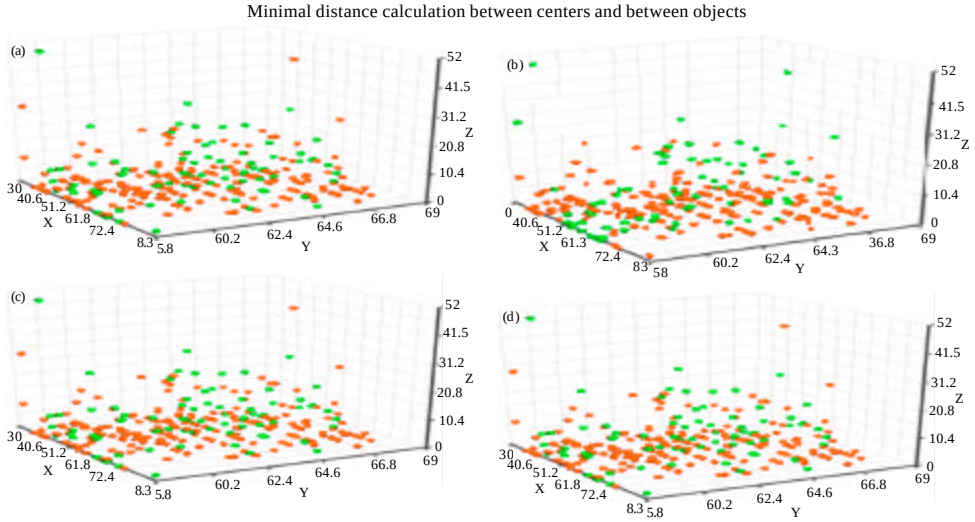

Fig. 5(a-d): Scatter graph for K-Means, MGD algorithm over H.S dataset for minimal distance calculation between centers and between objects using N3 neighborhood structure, (a) Initial length by K-Means = 2463.972, (b) Best length found between objects by MGD = 966.63, (c) Initial length by K-Means = 3626.530, (d) Best length found between centers by MGD = 2702.34

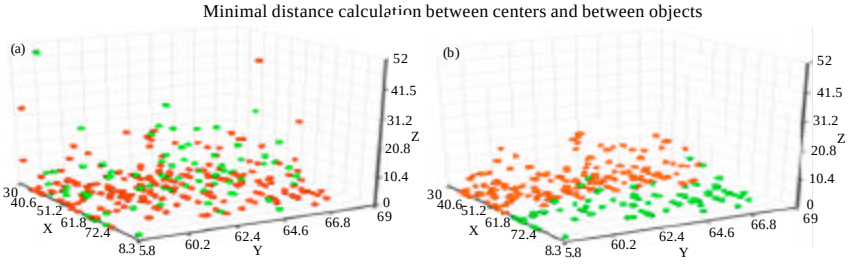

Fig. 6(a-b): Scatter graph for K-Means, MGD algorithm over H.S dataset for minimal distance calculation of the combination of between centers and between objects using N3 neighborhood structure, (a) Initial length by K-Mean = 6090.502 and (b) Best length found by MGD = 5152.66

datasets. For example, the best results for T.D dataset is 2929.61 that is obtained within 19 min 52 sec under 76436 iterations. Meanwhile, the range for minimum and maximum results is in between 2929.61 and 3001.97. In most of the cases, the results are obtained between 19 min 35 sec to 20 min 40 sec that are considered acceptable. Figure 6 shows a 3D scatter graph for K-Means, MGD algorithms using N3 neighborhood structure of the calculation between centers and between objects over H.S dataset.

CONCLUSION

This study proposed swap mechanism (N3) to increase the search diversification and improve the minimal distances quality. Two algorithms previously proposed were utilized for medical clustering problem for a comparison need, the Modified the Great Deluge algorithm (MGD) and hybridize Modified Great Deluge and Iterative Simulated Annealing (ISA-MGD). The comparison made between those algorithms employing N1 and N2 neighborhood structures to those algorithms employing N3 neighborhood structure based on the minimal distance that is calculated based on (i) Between objects and (ii) Between centers. Then, present a different minimal distance calculation using the combination of the between centers and objects to produce better clustering quality. The algorithms have been tested over six well known real medical clustering datasets.

Experimental results show that, when we use N3 in MGD or ISA-MGD algorithms will performs better than using N1 and N2 neighborhood structures in both minima distance calculations (i.e., i and ii), which indicates that using the swap mechanism (N3) is better in exploring the search space than using N1 and N2. Meanwhile, ISA-MGD algorithm has outperformed MGD algorithm using N3 in all datasets using between objects calculation, where they obtain same results in between centers calculation. This motivates to extend the test to use the third calculation (i.e., iii) using N3. The experimental results using iii calculation show that, MGD and ISA-MGD algorithms can produce cluster with more concentrated than using other calculations. Meanwhile, both MGD and ISA-MGD algorithms has outperformed each other in some datasets. Generally, it can be concluded that, the algorithms behave differently due to the different measurements imposed during the search process.

REFERENCES

Abuhamdah, A., 2012. Modified great deluge for medical clustering problems. Int. J. Emerg. Sci., 2: 345-360.

Abuhamdah, A., B.M. El-Zaghmouri, A. Quteishat and R. Kittaneh, 2012. Hybridization between iterative simulated annealing and modified great deluge for medical clustering problems. World Comput. Sci. Inform. Technol. J., 2: 131-136.

Berry, M.J. and G.S. Linoff, 1997. Handbook on Data Mining Techniques: For Marketing, Sales and Customer Relationship Management. John Wiley and Sons, New York, USA.

Brucker, P., 1978. On the Complexity of Clustering Problems. In: Optimization and Operations Research, Beckmenn, M. and H.P. Kunzi (Eds.). Springer, Berlin, Germany, pp: 45-54.

Dasgupta, S. and Y. Freund, 2009. Random projection trees for vector quantization. IEEE Trans. Inform. Theory, 55: 3229-3242.

Davidson, I. and A. Satyanarayana, 2003. Speeding up k-means clustering by bootstrap averaging. Proceedings of the IEEE Workshop on Clustering Large Data Sets, November 19, 2003, Melbourne, FL., USA., pp: 16-25.

Halkidi, M., Y. Batistakis and M. Vazirgiannis, 2001. On clustering validation techniques. J. Intell. Inform. Syst., 17: 107-145.

Holland, J., 1975. Handbook of Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, Michigan.

Hong, S., 2006. Experiments with K-means, fuzzy C-means and approaches to choose K and C. Ph.D. Thesis, University of Central Florida, Orlando, FL., USA.

Jain, A.K., M.N. Murty and P.J. Flynn, 1999. Data clustering: A review. ACM Comput. Surveys, 31: 264-323.

Liu, Y., Z. Yi, H. Wu, M. Ye and K. Chen, 2008. A tabu search approach for the minimum sum-of-squares clustering problem. Inform. Sci., 178: 2680-2704.

Mahajan, M., P. Nimbhorkar and K. Varadarajan, 2009. The Planar k-Means Problem is NP-Hard. In: WALCOM: Algorithms and Computation, Das, S. and R. Uehara (Eds.). Springer, USA., ISBN: 978-3-642-00201-4, pp: 274-285.

Saha, I., D. Pewczynski, U. Maulik and S. Bandyopadhyay, 2010. Consensus Multiobjective Differential Crisp Clustering for Categorical Data Analysis. In: Rough Sets and Current Trends in Computing, Szczuka, M., M. Kryszkiewicz, S. Ramanna, R. Jensen and Q. Hu (Eds.). Springer, USA., ISBN: 978-3-642-13528-6, pp: 30-39.

Wang, X., 2006. Fuzzy clustering in the analysis of fourier transform infrared spectra for cancer diagnosis. Ph.D. Thesis, School of Computer Science and Information Technology, University of Nottingham.

Zhang, C., D. Ouyang and J. Ning, 2010. An artificial bee colony approach for clustering. Exp. Syst. Appl., 37: 4761-4767.