International Journal of Industrial Engineering & Production Research (2010) pp. 71-79

September 2010, Volume 21, Number 2

International Journal of Industrial Engineering & Production Research

ISSN: 2008-4889

http://IJIEPR.iust.ac.ir/

Tabu-KM: A Hybrid Clustering Algorithm Based on Tabu Search Approach

M. Yaghini* **& N.Ghazanfari**

M.Yaghini, *Faculty of School of Railway Engineering, Iran University of Science and Technology*

N. Ghazanfari**,** *E-Learning Center-Iran University of Science and Technology*

KEYWORDS

Clustering problem,
Hybrid algorithm,
Tabu search algorithm,
k-Means algorithm.

ABSTRACT

The clustering problem under the criterion of minimum sum of squares is a non-convex and non-linear program, which possesses many locally optimal values, resulting that its solution often falls into these trap and therefore cannot converge to global optima solution. In this paper, an efficient hybrid optimization algorithm is developed for solving this problem, called Tabu-KM. It gathers the optimization property of tabu search and the local search capability of k-means algorithm together. The contribution of proposed algorithm is to produce tabu space for escaping from the trap of local optima and finding better solutions effectively. The Tabu-KM algorithm is tested on several simulated and standard datasets and its performance is compared with k-means, simulated annealing, tabu search, genetic algorithm, and ant colony optimization algorithms. The experimental results on simulated and standard test problems denote the robustness and efficiency of the algorithm and confirm that the proposed method is a suitable choice for solving data clustering problems.

1. Introduction

Clustering is an important process in engineering and other fields of scientific research. It is the process of grouping patterns into a number of clusters, each of which contains the patterns that are similar to each other according to a specified similarity measure. Clustering is a sequential process, which takes data as a raw material and produces clusters as a result without any predetermined goal [16].

To analyze the clusters, the objects are represented by points in N-dimensional space, where the objects of the vector are values for the attributes of the object and the objective is to classify these points into *K* clusters such that a certain similarity measure is optimized.

We consider clustering problem stated as follows: given *N* objects in R^m, allocate each object to one of *K* clusters such that the sum of squared Euclidean distances between each object and the center of belonging cluster is minimized. The clustering problem can be mathematically described as follows:

$$Min\ F(W,C) = \sum_{i=1}^{N}\sum_{j=1}^{K} wij \left\| x_i - c_j \right\|^2 \qquad (1)$$

Where $\sum_{j=1}^{K} w_{ij} = 1$, $i = 1,\ldots, N$. If object x_i allocated to cluster C_j, then w_{ij} is equal to 1; otherwise w_{ij} is equal to 0. In equation 1, *N* denotes the number of objects, *K* denotes the number of clusters, $X=\{x_1,x_2,\ldots,x_N\}$ denotes the set of *N* objects, $C=\{c_1,\ldots,c_K\}$ denotes the set of *K*

* **Corresponding author. M. Yaghini**
Email*: yaghini@iust.ac.ir*
Paper first received April. 07. 2010 ,and in revised form June. 24. 2010.

clusters, and W denotes the $N \times K$ 0-1 matrix. Cluster center c_j is calculated as follows:

$$c_j = \frac{1}{n_j} \sum_{x_i \in c_j} x_j, \quad j = 1,...,k \tag{2}$$

Where n_j denotes the number of objects belonging to cluster c_j. It is known that this clustering problem is a non-convex and non-linear which possess many locally optimal values, resulting that its solution often falls into these traps [24]. *k*-Means algorithm is one of the popular center based algorithms [18] which proved to fail to convergence to a local minimum under certain condition. The criterion it uses minimizes the total mean squared distance from each point in *N* to that point's closest center in *K*. However there are two main problems for *k*-means method [21] and [19]. First is that the algorithm depends on the initial states and the value of *K*. Second problem is that it is easily converges to some local optima which is much worse than the desired global optima solution.

In this paper, a new efficient algorithm is designed and implemented based on tabu search approach for escaping from local optima. The key idea of proposed algorithm is to produce tabu space and select new center of cluster from the objects not in tabu space. Then the *k*-means algorithm is run to local search.

This paper is organized as follows: the tabu search approach for clustering and also related works are reviewed in section 2. In section 3, we propose the Tabu-KM algorithm and give detailed descriptions.

Section 4 presents experimental results with simulated and standard datasets that show our method outperforms some other methods. Finally, conclusions of the current work are reported in section 5.

2. Tabu Search Approach for Clustering

Metaheuristic algorithms can be used to find very high-quality solutions to hard optimization problems in a reasonable time without being able to guarantee optimality [14]. They often come out as a result of imitation of the real world. Tabu search algorithm [22] is among those solution-based metaheuristic algorithms which start from an initial solution maybe randomly generated. Continue the search process by performing some transformation to the solution and corresponding values of the objective function. In its simple form, choose the next potential solution in a random way and in its evolutionary form; it explores the neighborhood in a systematic manner based on a logic in order to improve the quality of the solution.

2.1. Tabu Search

Tabu search [7] and [22, 23] is based on a local or neighborhood search procedure with local optima avoidance but in a rather deterministic way. The key idea of tabu search is the acceptance of non-improving neighboring solutions. In order to avoid returning to the local optimal solution just visited, the reverse move must be forbidden. This is done by short term memory called tabu list.

The basic components of the TS algorithm are defined as follows:

1. *Configuration* is an assignment of values to variables. It is a solution to the optimization problem.
2. *Move* is a specific procedure for getting a trial solution, which is feasible to the optimization problem that is related to the current configuration.
3. *Search space and neighborhood* is the set of all neighbors, which are the adjacent solutions that can be reached from any current configuration. It may also include neighbors that don't satisfy the given customary feasible conditions.
4. *Candidate subset* is a subset of the neighborhood. It is to be examined instead of the entire neighborhood, especially for large problems where the neighborhoods have many elements.
5. *Tabu restrictions* are constraints that prevent the chosen moves to be reversed or repeated. They play a memory role for the search by making the forbidden moves as *tabu.* The tabu moves are stored in a list, called *tabu list.*
6. *Aspiration criteria* are rules that determine when the tabu restrictions can be overridden, thus removing a tabu classification otherwise applied to a move. If a certain move is forbidden by some tabu restrictions then the aspiration criteria, when satisfied, can make this move allowable.

Simple TS as described above can sometimes successfully solve difficult problems, but in most cases, additional elements have to be included in the search strategy to make it fully effective. These elements are intensification and diversification.

1. *Intensification*: The idea behind the concept of search intensification is that, as an intelligent human being would probably do, one should explore more thoroughly the portions of the search space that seem "promising" in order to make sure that the best solutions in these areas are indeed found.
2. *Diversification:* One of the main problems of all methods based on *local search* approaches, and this includes TS in spite of the beneficial impact of tabus, is that they tend to be too local, they tend to spend most, if not all, of their time in a restricted portion of the search space. Diversification is used to lead the search into new regions of the solution space.

2.2. Related Works

In recent years, the various metaheuristic algorithms are used to solve clustering problem so as to avoid converging in local optima solutions. Clustering algorithms may be improved through the following methods: (1) determination of initial parameters, (2) changing the base algorithm, and (3) combination of

clustering algorithm with other metaheuristic algorithms.

Al-sultan (1995) [1] employed the string-of group-numbers encoding and proposed a tabu search-based clustering (TSC) algorithm for clustering problem. After a specified number of iterations, the best obtained solution is viewed as the clustering result. Al-sultan and Fedjki (1997) [3] used tabu search to solve the fuzzy clustering problem.

Sung and Jin (2000) [20] presented a heuristic algorithm to partition datasets with non overlapping clusters by combining tabu search and two functional procedures, packing procedure and releasing procedure.

Liu et al. (2008) [12] presented a tabu search based clustering approach called TS-Clustering to deal with the minimum sum of square clustering problem. In this algorithm, five improvement operations and three neighborhood modes are given.

The improvement operation is used to enhance the clustering solution obtained in process of iterations and the neighborhood mode is used to create the neighborhood of tabu search. Their method uses the generalized string property to group similar objects together and set up the initial solution. Then the releasing procedure separates packed elements from each other so as to promote the effectiveness of the solution search. Sung and Jin (2000) [20] is employed tabu search as a sub-module to improve the solution so as to avoid the reallocation trial of packed elements trapping in local minima. Al-sultan and Khan (1996) [2] compared four clustering algorithms and found that the tabu search and other metaheuristic algorithms based clustering methods are comparable in terms of the solution quality and better than the *k*-means algorithm.

Since *k*-means is computationally attractive, the combination of *k*-means and tabu search is suitable for the clustering problem. Liu et al. (2005) [11] presented a tabu search based clustering method called *k*-means tabu clustering (KTC).

KTC owns the same time complexity as TSC and gets better results sooner than TSC. Gungor et al. (2008) [8] developed *k*-Harmonic means data clustering which used tabu search method (tabuKHM) to solve the initialization problem trapping to the local minima of this algorithm.

Pan et al. (2007) [15] proposed a framework of automatic clustering algorithms called ETSAs that do not require users to give each possible value of required parameters including the number of clusters. ETSAs treat the number of clusters as a variable, and evolve it to an optimal number. It is seen that the clustering results are improved by the combinations of the metaheuristic and *k*-means algorithms [5, 6] and [9, 10] and [13].

In this paper, our aim is to design and implement a new hybrid algorithm using tabu search and *k*-means method for clustering problem. The proposed algorithm improves the objective function values of *k*-means by investigating search space through the structure of tabu search. The new idea in this paper is producing tabu space for escaping from the trap of local optima and finding better solutions effectively.

3. The Tabu-KM Algorithm

In this paper, new algorithm has observed the architecture of tabu search. Based on the structure of tabu search, the hybrid algorithm of Tabu-KM is designed and implemented.

The new algorithm gathers the optimization property of tabu search and the local search capability of *k*-means algorithm together. In proposed algorithm, the spherical tabu space around the best-so-far solution is produced in each repetition of algorithm. Moreover, other objects, which are not in tabu space, are permitted to choose as center of new cluster. This object has the least radius of best-so-far center. This causes escaping from the trap of local optima and finding better solutions effectively.

3.1 The Elements of Tabu-KM Algorithm

We will now detail the algorithm used in our study, starting with a short description of five basic components.

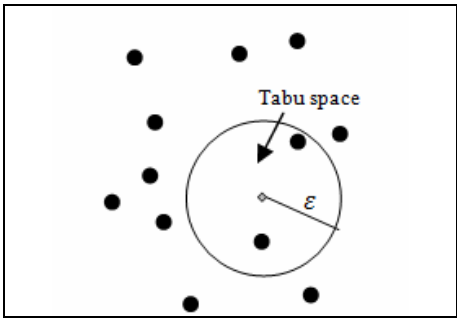
Configuration is a solution or an assignment of values to variables. At the beginning of the algorithm, *k*-means algorithm generates a feasible solution. Then, the centroids of all clusters are calculated. After that, clusters are selected sequentially in order to generate new solution though the logic of algorithm. The center of selected cluster is considered as the starting point of algorithm.

There are two types of tabus in the Tabu-KM algorithm. They are tabu space and tabu list. The key idea of tabu search in the Tabu-KM algorithm is the use of tabu space for containing forbidden centers of current cluster. The two strategies for implementing tabu spaces are investigated:

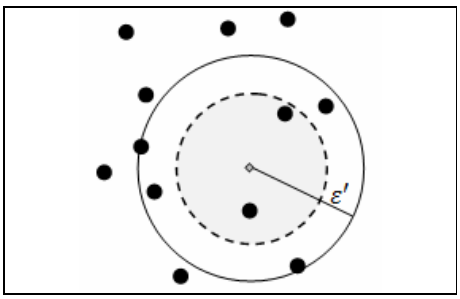
- *Static strategy.* This strategy considers the spherical space around the center of cluster with fixed radius as tabu space called neighborhood radius. All points in this space are forbidden to choose as center of cluster. All objects in tabu space area are added to tabu list. If the algorithm could not escape from local optima, the radius is increased with a fixed value. In this strategy, the radius is a parameter which is difficult to be tuned for practical problems. Also, the clustering result is highly dependent on this parameter.
- *Dynamic strategy.* This strategy considers the spherical space around the center of cluster with dynamic radius as tabu space. In dynamic strategy, the radius of tabu space is computed based on the distance between the object and center of cluster, which is least.

If the algorithm could not escape from local optima, the radius is increased by selecting the next nearest object to the center of cluster, which is not in tabu space.

After making or extending tabu space, an object located out of tabu space is selected as a new center of cluster. After an iteration of algorithm, it is investigated if the center of cluster is going back to tabu space or not. In case the new cluster center is in tabu space, it means that we are in local optima condition and we should change movement direction by restricting the reselection of this object as the cluster center. In case the new cluster center is not in tabu space, the direction of next movement will be determined according to the objective function value as well as determination of improvement or non-improvement of solution. An example of the tabu space in the static strategy is illustrated in Fig. 1. In Fig. 1(a) the static strategy makes a tabu space with neighborhood radius called ε. If the algorithm could not escape from local optima, the radius is increased with a fixed value α. Then tabu space is extended with new radius $\varepsilon' = \varepsilon + \alpha$ (see Fig. 1(b)). Fig. 2(a) shows the dynamic strategy to make a tabu space. The radius of tabu space, as *r,* is computed based on the least distance between the object and center of cluster. If the algorithm could not escape from local optima, the radius, as $\acute{r}$ and $\check{r}$, is increased by selecting the next nearest object to the center of cluster which is not in tabu space, as shown in Fig. 2(b).

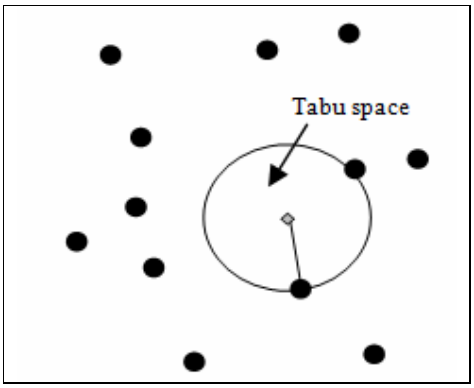


(a) Making tabu space

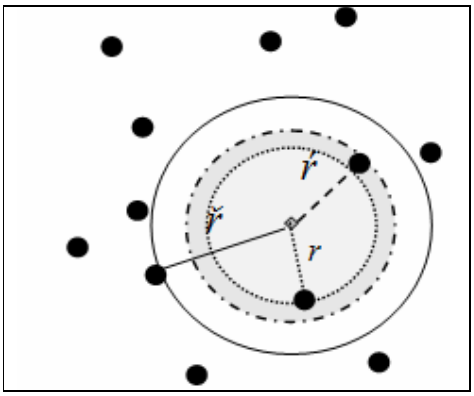


(b) Extending tabu space

Fig. 1. An example of tabu space in static strategy



(a) Making tabu space



(b) Extending tabu space

Fig. 2. An example of tabu space in dynamic strategy

A move is a process of generating a feasible solution to the problem that is related to the current solution. The duty of this generator is to select the new center of cluster from the objects in the neighborhood of current cluster center, which is not in tabu space. Then, the *k*-means algorithm is performed to generate new solution. The three strategies for candidate moves in clusters are investigated.

(a) *Move to closest object to the center of k-means solution*: the spherical space around the center of selected *k*-means algorithm is investigated. In this strategy, the radius of spherical search space is increased according to the distance of the object to the center of initial *k*-means cluster.

(b) *Move to closest object to the center of current solution*: the spherical space around the current center of cluster is investigated. The new center of cluster is selected from the objects in the neighborhood of current center, which is not in tabu space. It should be considered that current solution is not always better solution. As a result, a feasible solution is generated through the different direction of search space.

(c) *Move to closest object to the center of best-so-far solution*: the spherical space around the cluster center of best-so-far solution is investigated. In this strategy,

the radius of spherical search space is increased to permit other objects not in tabu space, selected as center of new cluster. In fact, after finding better solution, the spherical space is considered around the center of best-so-far solution, which might be in different direction of search space.

In our experiment, given the above strategies, "*move to closest object to the center of best-so-far solution*" is chosen to investigate search space around the center of best-so-far solution. In this strategy, intensification search is utilized to extend search space to explore more spaces that seems better solutions would probably find.

After each performing of *k*-means in Tabu-KM algorithm, if the new center of cluster is in tabu space, the value of solution is computed according to the objective function. If the solution has improved the value of best solution generated so far, the solution is accepted as best-so-far solution. In fact, although we came back to tabu space and we should restrict the reselection of this object as the cluster center, because of satisfying the aspiration criterion, the best-so-far solution is updated by the current solution.

The process is stopped when all clusters are investigated sequentially and no improvement is achieved in the best-so-far solution. In the investigation of each cluster, the centers of other clusters may be changed, so it is necessary to reinvestigate of all clusters to find better solution.

3.2. Specification of the Tabu-KM Algorithm

The Tabu-KM algorithm included two parts: main loop of algorithm and the second part which we named *Escape_Local_Optima()*. However, before we state our algorithm, we need to introduce some notation:

- K: the number of clusters.
- *Max_Tabu*: the maximum number of extending tabu space for each cluster.
- *K_Cluster*: the counter of cluster is being processed.
- J_b: the objective function value of the best-so-far solution S_b.
- C_b(*K_Cluster*): the center of cluster by *K_Cluster* counter of the best-so-far solution S_b.
- *Checked[ ]*: A logical array with K elements. The ith element is equal to *true* when ith cluster is processed, and *false*, otherwise. At the beginning of algorithm, set all elements with *false* value.
- *Tabu_Extend*: the current number of extending tabu space.
- *Tabu_List*[]: an array which denotes tabu objects.
- *Tabu_Space*: the area of search space, which is tabu.

The main loop of Tabu-KM algorithm is implemented as follows:

Step 1: Initialize *Checked*[]=*false* for all elements and *K_Cluster=1*.
Step 2: Perform *k*-means algorithm. Let S_c be an initial solution by *k*-means algorithm and J_c be the corresponding objective function value of the current solution computed using equation (1) and (2).
Let $S_b=S_c$ and $J_b=J_c$.
Let $C_b(K\_Cluster) = C_c(K\_Cluster)$ which $C_c(K\_Cluster)$ denotes the center of cluster by *K_Cluster* counter of cuurent solution.
Step 3: Call *Escape_Local_Optima()*.
Step 4: Let *Checked*[*K_Cluster*]=*true*.

Step 5: If $J_c < J_b$, (best solution is improved) Then Set *Checked[i]* with *false* for all elements except i = *K_Cluster*.

Step 6: *K_Cluster=K_Cluster+1*.
Step 7: If *K_Cluster* > K Then *K_Cluster=1*.
Step 8: If *Checked[K_Cluster] = true* Then Stop, otherwise go to step 3.

The *Escape_Local_Optima()* algorithm is implemented as follows:
Step 1: Initialize *Tabu_Extend=1*.
Step 2: Make (or extend) a *Tabu_Space* around the $C_b(K\_Cluster)$ which the radius is the closest object to the center of S_b as O_i with least distance of center of cluster as d, if is not in tabu space.

Tabu_Space = Tabu_Space $\cup$ O_i
Tabu_Extend = *Tabu_Extend* + 1.

Step 3: Let $C_i(K\_Cluster) = O_i$ and perform *k*-means algorithm to find new solution S_c according to the objective function value J_c and C_i(*K_Cluster*) as the center of current solution.

Step 4:
If C_i(*K_Cluster*) located in *Tabu_Space* (is in tabu space) and $J_c < J_b$ (the current solution satisfies the aspiration criterion), Then let $S_b = S_c$, $J_b = J_c$, $C_b(K\_Cluster)=C_i(K\_Cluster)$, Empty *Tabu_Space* and Delete *Tabu_List* and set *Tabu_Extend=1,* then go to step 5.
If C_i(*K_Cluster*) located in *Tabu_Space* (is in tabu space) and $J_c > J_b$, Then
Tabu_Extend=Tabu_Extend+1, and go to step 5.
If $C_i(K\_Cluster) \notin$ *Tabu_Space* (is not in tabu space) and $J_c < J_b$ Then let $S_b = S_c$ and $J_b = J_c$, $C_b(K\_Cluster)= C_i(K\_Cluster)$, Empty *Tabu_Space*, and Delete *Tabu_List*, and set *Tabu_Extend*=1, go to step 5.

If $C_i(K\_Cluster) \notin Tabu\_Space$ (is not in tabu space) and $J_c > J_b$ Then *Tabu_Extend=Tabu_Extend*+1 and go to step 5.
Step 5: If *Tabu_Extend* > *Max_Tabu* , Then return to main loop of Tabu-KM algorithm.
In addition, the flow diagram of *Escape_Local_Optima()* is shown in Fig. 3.

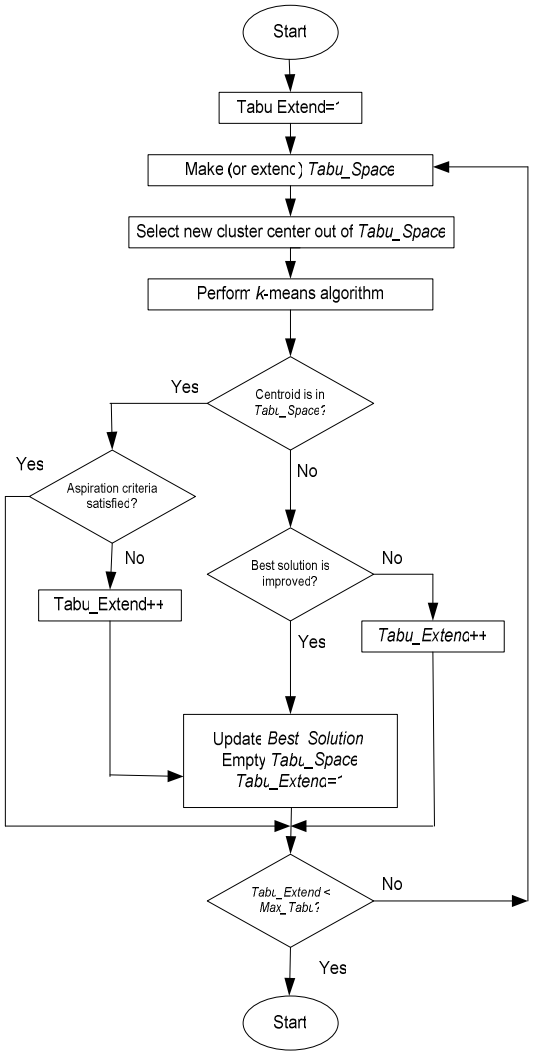


Fig. 3. Escape_Local_Optima() algorithm flow diagram

4. Experimental Studies

The new algorithm is coded in visual basic and tested on a computer with 2 GHz CPU and 1 GB memory on several datasets. To evaluate the performance of the algorithm, two groups of simulated and standard datasets are used. The employed measure for all datasets is the Euclidean distance, which computes the sum of squared deviations between all data objects as *X* and their belonging cluster center as *C*.

$$d(x_i, c_j) = (|x_{i1} - c_{j1}|^2 + |x_{i2} - c_{j1}|^2 + \ldots + |x_{ip} - c_{j1}|^2)^{\frac{1}{2}}. \quad (3)$$

By the way, all datasets are normalized according to (5). It is performed by a linear transformation on the original data. In fact, the attribute data are scaled so as fall within a small specified range 0 to 1.

$$x_i' = \frac{x_i - \min(x)}{\max(x) - \min(x)} \quad (4)$$

The comparison of results for each dataset is based on the best, average and worst values of the clustering metric ($\mathbf{F_{Best}}$, $\mathbf{F_{Avrg}}$, $\mathbf{F_{worst}}$) after 100 runs for each of the algorithm. The results of proposed algorithm are compared with five other algorithms for their effectiveness and efficiency. They include *k*-means, simulated annealing, tabu search, genetic algorithm, and ant colony optimization algorithms.

4.1. Simulated Problems

The effectiveness of any clustering algorithm is usually dependent on datasets. In this paper, the design strategy is adapted to generate our test dataset by incorporating several design factors including number of objects, number of clusters and number of dimensions similar to Sung and Jin (2000) [12]. The simulated test problems are randomly generated in R^2 where the data sets have 100 records and three dimensional.
As we discuss below, the simulated problems are selected from a bigger set of generated problems.
First, we generated different 100 random problems. Then, we solve them by k-means algorithm 1000 times. Because the initial cluster centers are selected randomly, in each run of k-means algorithm, we get different solutions.
Finally, we selected 10 problems that they have more diverse solutions. It means we choose the problems that they have more local optima. Consequently, the ten selected problems are most difficult problems among those 100 random problems.
Some test problems were generated for tuning of Tabu-KM parameters. Important parameter in our algorithm is *Max_Tabu,* which is the number of extending of tabu space. After testing different values, we finally choose "the ten percent of all objects" for this parameter.
The results obtained by the Tabu-KM algorithm and a comparison with the *k*-means algorithm for simulated problems mentioned in table 1.
The Tabu-KM and *k*-means algorithms were run a hundred times and considered the worst, average and best results. The percentage of improvement is computed by comparing the average results of *k*-means

and Tabu-KM algorithms. The %Improve is defined as=

$$\%\text{Improve} = ABS\left(\frac{F_{Avg}(Tabu-KM)-F_{Avg}(k-means)}{F_{Avg}(k-means)}\right)\times 100 \quad (5)$$

Tab. 1. Results obtained by Tabu-KM algorithm for 10 simulated datasets

DataSet	K-means			Tabu-KM			%Improve (F_{avrg})
	F_{Best}	F_{Avrg}	F_{Worst}	F_{Best}	F_{Avrg}	F_{Worst}	
SD1	5.69	5.94	6.55	5.69	5.74	6.06	3.37
SD2	5.48	5.59	6.39	5.48	5.48	5.48	1.97
SD3	5.97	6.17	6.66	5.97	6.13	6.46	0.65
SD4	6.12	6.38	6.77	6.12	6.24	6.71	2.19
SD5	5.78	6.00	6.68	5.78	5.84	6.07	2.67
SD6	5.82	6.03	7.08	5.82	5.84	6.20	3.15
SD7	6.08	6.33	7.51	6.08	6.22	7.11	1.74
SD8	5.48	5.66	7.04	5.48	5.48	5.48	3.18
SD9	5.45	5.65	6.69	5.45	5.47	5.62	3.19
SD10	5.46	5.47	5.53	5.46	5.46	5.47	0.18
Average	5.73	5.92	6.69	5.73	5.79	6.06	2.20

Comparing results with *k*-means, we notice that Tabu-KM algorithm would be able to improve the result of *k*-means algorithm and generate solution close to best solution in most of the iterations. By comparing the average and worst solution values of proposed algorithm with *k*-means, we notice that Tabu-KM algorithm would be able to improve the result of *k*-means algorithm and generate solution close to best solution in most of the iterations. The comparison result is shown in Fig. 4.

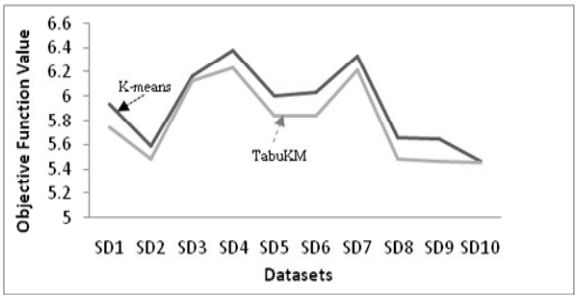


Fig. 3. A comparison of *k*-means, Tabu-KM and Best solution for simulated datasets

4.2. Standard Problems

Many authors have considered the iris and human thyroid disease datasets from the UCI repository of machine learning, as data-clustering problems to study and evaluate the performance of their algorithms [4].
Iris dataset contains of N=150 of samples of three iris flowers (K=3) as Virginica, Setoza, and Versicolor. Each object is defined by four attributes as: sepal length, sepal width, petal length, and petal width (n=4). Thyroid dataset categories of N=215 samples of patients suffering from three human thyroid diseases, (K=3) as: euthyroid, hyperthyroidism and hypothyroidism patients where 150 individuals are tested euthyroid thyroid, 30 patients are experienced hyperthyroidism thyroid while 35 patients are suffered by hypothyroidism thyroid. Each individual was characterized by the result of five laboratory tests (n=5) as: total serum thyroxine, total serium tri-iodothyronine, serum tri-iodothyronine resin uptake, serum thyroid-stimulating hormone (TSH), and increase TSH after injection of TSH-releasing hormone. The effectiveness of stochastic algorithms is greatly dependent on the generation of initial solutions. The results obtained by the new algorithm and a comparison with the *k*-Means algorithm for Iris and Thyroid datasets are mentioned in table 2.

Tab. 2. Results obtained by Tabu-KM algorithm for Iris and Thyroid datasets

Dataset	*k*-means			Tabu-KM			%Improve (average)
	F_{Best}	F_{Avrg}	F_{Worst}	F_{Best}	F_{Avrg}	F_{Worst}	
Iris	78.94	90.51	145.28	78.94	86.61	104.88	4.5
Thyroid	10151.83	10251.19	10411.28	9402.75	10162.13	10340.63	0.87

Tab. 3. Results obtained on Iris dataset

Method	F_{Best}	F_{Avrg}	F_{Worst}	CPU Time(s)
ACO	97.100777	97.171546	97.808466	33.72
GA	113.986503	125.197025	139.778272	105.53
TS	97.365977	97.868008	98.569485	72.86
SA	97.100777	97.136425	97.263845	95.92
Tabu-KM	78.9408410	86.611700	104.8788	0.9
%Improve*	18.70	10.52	-7.61	99.06

* Comparing with the result of SA algorithm

In our experiment, we also compared the result of new algorithm with other metaheuristic algorithms such as genetic algorithm, simulated annealing, ant colony optimization, and tabu search which are published in Shelokar et al. (2004) [17]. The results are presented for Iris and Thyroid datasets in table 3 and table 4 respectively.

Tab. 4. Results obtained on Thyroid dataset

Method	F_{Best}	F_{Avrg}	F_{worst}	CPU Time(s)
ACO	10111.82776	10112.1269	10114.8192	102.15
GA	10116.29486	10128.82315	10148.3896	153.24
TS	10249.72917	10354.315o21	10438.7804	114.01
SA	10111.82776	10114.04527	10118.9344	108.22
Tabu-KM	9402.752272	10162.12641	10340.6308	1.1
%Improve*	7.01	-0.49	-2.23	98.92

• Comparing with the result of ACO algorithm

Metaheuristic algorithms have to be used in order to find near-optimal solutions. They seek for high quality solutions at a reasonable computational time, but cannot guarantee that a problem will be solved in terms of obtaining the optimal solution. In this study, it can be observed that *k*-means algorithm is combined with tabu search in order to overcome local optima problem in clustering. The comparing results of the clustering obtained by five methods with Iris and Thyroid datasets respectively are shown in above tables. For Iris dataset, the best and average solution values are significantly improved. For Thyroid dataset, the best solution value is improved in compare with other metaheuristic algorithms. The CPU time of proposed algorithm for both datasets is showed a very drastically improved. The results illustrate that the proposed hybrid algorithms can escape from local optima trap and find better solutions in most of the cases.

5. Conclusions

In this paper, an effective hybrid clustering algorithm based on tabu search approach called Tabu-KM is developed by integrating the tabu space and move generator for restricting objects to select as center of cluster. Tabu-KM algorithm is used to escape from the trap of local optima and finding better solutions, in the clustering problem under the criterion of minimum sum of squares. To produce the tabu space, two strategies are investigated: the spherical space around the center of cluster with fixed or dynamic radius. In addition, three different strategies are discussed to select objects as center of new cluster and generate a feasible solution: (1) move to the closest object to the center of initial *k*-means cluster, (2) move to the closest object to the center of current cluster, (3) move to the closest object to the center of best-so-far cluster. All above-mentioned strategies were investigated. According to the result, the dynamic radius for tabu space and move to the closest object to the center of best-so-far solution is utilized to select objects out of tabu space as center of current cluster. To evaluate the performance of Tabu-KM algorithm, it is tested on several simulated and well known datasets. Tabu-KM is compared with several typical metaheuristic algorithms including simulated annealing, genetic algorithm, tabu search, and ant colony optimization algorithms. As a result, the Tabu-KM algorithm can be considered as an effective algorithm to find better solutions for clustering problems of allocating *N* objects to *K* clusters.

References

[1] Al-Sultan, K.S., "*A Tabu Search Approach to the Clustering Problem,*" Pattern Recognition, Vol. 28, 1995, pp. 1443–1451.

[2] Al-sultan, K.S., Khan, M.M., "*Computational Experience on Four Algorithms for the Hard Clustering Problem,*" Pattern Recognition Letters, Vol. 17, 1996, pp. 295–308.

[3] Al-sultan, K.S., Fedjki,C.A. "*A Tabu Search-Based Algorithm for the Fuzzy Clustering Problem,*" Pattern Recognition, Vol. 30, 1997, pp. 2023–2030,.

[4] Asuncion, A., Newman, D.J., UCI Machine Learning Repository, (http://www.ics.uci.edu/~mlearn/MLRepository.html). Irvine, CA: University of California, School of Information and Computer Science, 2007.

[5] Bandyopadhyay, S., Maulik, U., "*An Evolutionary Technique Based on k-Means Algorithm for Optimal Clustering in RN,*" Information Sciences, Vol. 146, 2002, pp. 221–237.

[6] Bandyopadhyay, S., Maulik, U., Akhira, M.K.P, "*Clustering Using Simulated Annealing with Probabilistic Redistribution,*" International Journal of

Pattern Recognition and Artificial Intelligence, Vol. 15, 2001, pp. 269–285.

[7] Glover, F., Laguna, M., "*Tabu Search,"* In Handbook of Applied Optimization, P.M.Pardalos, & M.G.C.Resende (Eds.), Oxford University Press, 2002, pp. 194-208.

[8] Gungor, Z., Unler, A., "*k-Harmonic Means Data Clustering with Tabu Search Method,"* Applied Mathematical Modelling, Vol. 32, 2008, pp. 1115-1125.

[9] Kivijarvi, J., Franti, P., Nevalainen, O., "*Self-Adaptive Genetic Algorithm for Clustering,"* Journal of Heuristics, Vol. 9, 2003, pp. 113–129.

[10] Krishna, K., Murty, M.N., "*Genetic k-Means Algorithm,"* IEEE Transactions on Systems, Man, and Cybernetics, Part B, Vol. 29, 1999, pp. 433–439.

[11] Liu, Y., Wang, L., Chen, K., "*A Tabu Search Based Method for Minimum Sum of Squares Clustering,"* ICAPR2005, Lecture Notes in Computer Science, 3686, 2005, pp. 248-256.

[12] Liu, Y., Yi, Z., Wu, H., Ye, M., Chen, K., "*A Tabu Search Approach for the Minimum Sum-of-Squares Clustering Problem,"* International Journal of Information Sciences, Vol. 178, No. 12, 2008, pp. 2680-2704.

[13] Liu, Y., Wang, L. , & Chen, k., "*A Hybrid Tabu Search Based Clustering Algorithm,"* KES 2005, LNAI3682, Springer-Verlag Berlin Heidelberg, 2005, pp. 186-192.

[14] Misevicius, A., Blazauskas,T., Blonskis, J., Smolinkas, J., "*An Overview of Some Heuristic Algorithms for Combinatorial Optimization Problems,"* Informacines Technologijos Ir Vadymas, Vol. 30, No.1, 2004.

[15] Pan, S.M., Senior, K.S., "*Evolution-Based Tabu Search Approach to Automatic Clustering,"* IEEE Transaction on Systems, Man, and Cybernetics, Part C-Application and Reviews, Vol. 37, No.5, 2007, pp. 827-838.

[16] Selim,S.Z., Ismail, M.A., "*k-Means-Type Algorithm,"* generalized convergence theorem and characterization of local optimality, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 6, 1984, pp. 81–87.

[17] Shelokar, P.S., Jayaraman, V.K., Kulkarni, B.D., "*An ant Colony Approach for Clustering,"* Analytica Chimica Acta 509, 2004, pp. 187–195.

[18] Steinley, D., Brusco, M.J, "*Initializing k-Means Clustering,"* A critical evaluation of several techniques, Journal of Classification, Vol. 24, 2007, pp. 99-121.

[19] Steinly, D., Hubert, L., "*Order-Constrained Solutions in k-Means Clustering: Even Better Than Being Globally Optimal,"* The Psychometric Society, 2008, pp. 647-664.

[20] Sung, C.S., Jin, H.W., "*A Tabu-Search-Based Heuristic for Clustering,"* Pattern Recognition, Vol. 33, 2000, pp. 849–858.

[21] Zhang, Z., Tian, B., Tung, A.K.H., "*On the Lower Bound of Local Optimum in k-Means Algorithm,"* Proc. 6th Int. Conf, Data Mining, (ICDM'06), IEEE, 2006, pp. 2701-2709.

[22] Glover, F., Laguna M., Tabu Search, Kluwer Academic Publishers, Boston, 1997.

[23] Gendreau M., "*An Introduction to Tabu Search,"* In Handbook of metaheuristics, Kochenberger, G., Glover, F., (Eds.), Dordrecht, Kluwer Academic Publishers, 2003.

[24] Jain, A.K., Dubes, R., "*Algorithms for Clustering Data. Prentice- Hall,"* New Jersey, 1988.