International Congress of Sciences and Innovative Technologies

■ Sept. 5-6, 2018 Babol Noshirvani University of Techonology- Babol, Iran

Automatic clustering of big datasets using a swarm intelligence method

Iman Behravan[1,*], Seyed Hamid Zahiri[2], Seyed Mohammad Razavi[3] , Roberto Trasarti[4]

[1]Department of Electrical Engineering, University of Birjand, Birjand, Iran
[2] Department of Electrical Engineering, University of Birjand, Birjand, Iran
[3] Department of Electrical Engineering, University of Birjand, Birjand, Iran
[4] KDD Lab, ISTI-CNR, Pisa, Italy
**Corresponding Author: (Phone: + 98 915 163 2809; Email: i.behravan@birjand.ac.ir)*

Abstract

Mining and discovering knowledge from big datasets have become a new interesting field of research among data scientists. In fact, extracting hidden patterns in big datasets using traditional data mining algorithms in a reasonable period of time and with an acceptable accuracy is impossible due to high volume of data and their complexity. Generally, the term big data is referred to massive datasets with huge number of high dimensional samples which makes them very hard to be analyzed by conventional data mining techniques. So designing new and effective algorithms for analyzing big datasets is necessary. Clustering, which is the process of dividing the data points into different groups based on their similarities and dissimilarities, is one of the most important data mining and big data mining methods. *K-means*, which is one of the most popular clustering algorithms and has been widely used in several researches, suffers from some drawbacks such as: its tendency to converge to a local optimum point, the quality of its final results depends on the initial centroids generated randomly and its inability in finding the number of clusters. In this paper a new automatic big data clustering method, based on a swarm intelligence algorithm, is introduced which has a great ability in finding the number of clusters and escaping from local optimum point. The proposed method is tested on 13 synthetics and 2 real big mobility datasets. Final results demonstrate its power in big data clustering.

Key words: Automatic clustering, Big data analytics, K-means, Swarm intelligence

1. Introduction

The huge amount of data created constantly with increasing rate from different sources such as smart phones, social media, imaging technologies and etc. becomes difficult to be analyzed by conventional data analytic tools. For this reason a new field of research called Big Data Analytics [1] is growing faster in the research and industrial communities. Big data is defined as the dataset whose size is beyond the classical data management tools and processing techniques, in other words all data life phases must be reconsidered such as the storage, the management and the

analysis [2]. Big data analytics, which has attracted more and more attention among researchers, is to automatically extract knowledge from large amounts of data. It can be seen as mining or processing of massive datasets. [3]. There are several challenges in analyzing massive datasets such as large volume of data, dynamical changes of data, data noise and etc. These challenges cause difficulties in extracting hidden patterns from big data, so new and efficient algorithms should be designed to solve big data analytic problems. One of the most important tools in data mining is clustering which is used in different fields such as, biology, ecology, social science, marketing and psychology as useful tool for pattern recognition and profiling [4]. Actually the aim of clustering is, to divide the objects of a dataset to a specified number of groups (or clusters) based on their similarities. Many techniques have been introduced for clustering such as *K-means* [5] and *fuzzy c-means* [6]. These traditional clustering techniques, fail to give accurate results when dealing with huge amount of data because of their complexity and computational costs. For example, the traditional *K-means* clustering is NP-hard even when the number of clusters is k = 2. Many real-world applications can be formulized as optimization problems that needs kinds of algorithms capable of solving such optimization problems [7]. Most traditional optimization methods are only able to solve non-complex and continuous problems. So heuristic algorithms are proposed to solve complex and also discrete optimization problems which cannot be solved by the traditional optimization methods. Recently swarm intelligence (SI) and evolutionary algorithms (EA), two kinds of heuristics, are attracting more and more attentions from researchers. In swarm intelligence methods, the solutions cooperate with each other to search different areas of the solution space and find the best possible solution [8]. Several kinds of swarm intelligence algorithms have been proposed up to now such as *Particle swarm optimization (PSO)* [9], *Inclined Planes system Optimization (IPO)* [10] , *Gravitational Search Algorithm (GSA)* [11], *Ant Colony Optimization (ACO)* [12] and etc. In this paper a new methodology for clustering big datasets using particle swarm optimization algorithm, is introduced. The paper is organized in the following manner: in section 2, some related researches are analyzed, after that particle swarm optimization algorithm is discussed. In section 4 the proposed method (APSO-Clustering) is explianed and section 5 and 6 contain the experimentation results over well-known synthetic datasets and a real trajectory dataset respectively. Finally, section 7 presents the conclusion.

2. Related works

In [13] a clustering method, based on genetic algorithm, is introduced. They encode the chromosomes so that each chromosome includes *n* number of genes which is equivalent to the number of data points in the dataset. Each gene holds the label of its corresponding sample. In the case of dealing with big datasets with huge number of samples, the length of the chromosomes will be very high and because of the large scale optimization problem, the algorithm may fail to find the best solution in a reasonable time. Also high computation and complexity of GA in compare to SI algorithms makes it slower for big datasets. In [14] Abraham et al. introduced a clustering algorithm called MEPSO. They used *Xie-Benni* index for evaluating the quality of clusters. In this method, the maximum number of clusters should be determined by the user which is a big drawback. Also the algorithm's performance was tested on 4 synthetic dataset with low number of

samples and features. They used it for image segmentation. In [15] a hybrid method for clustering text documents is introduced which is a combination of PSO and *K-means*. In the first stage, PSO performs a globalized search and in the next step *K-means* performs a local search. The output of the PSO, in the first step is used as the initial points for the *K-means* in the second stage. The main drawback of this method is its inability in finding the number of clusters, which is very important specially in big data clustering. In this methodology the distance between the data points and their correponding centroid is used as fitness function. Although the complexity of this function is very low, which makes the algorithm faster in compare to other indexes, but its accuracy in finding overlapped clusters is not high enough. In [16] Omran *et al.* proposed a new segmentation method based on particle swarm optimization algorithm. In this method first, a pool of centroids from the data points is selected and after that the best set of centroids among them will be found. Like the previous, method *K-means* is used in the second stage to refine the detected centroids. This may increase the probablity of finding local opimtum instead of global optimal point but the main problem of this research is predefining the maximum number of clusters. In [17] Zhang *et al.* proposed a clustering method based on artificial bee colony (ABC) optimization algorithm [18] for clustering problem. They used ABC algorithm to find the best possible centroids and a total mean-square quantization error for evaluating the solutions. Again the proposed method is unable to find the proper number of clusters. Krishnasamy *et al.* proposed a clustering method using a new heuristic algorithm called Cohort Intelligence (CI) [19]. This algorithm is inspired from natural and society tendency of cohort candidates of learning from one another [20]. The proposed algorithm benefits from the advantages of both *K-means* and a modified version of CI (MCI). This combination allows the proposed algorithm to converge more quickly but it still suffers from lack of ability in finding the number of clusters. Lu *et al.* [21] proposed an improved *K-means* using a heuristic algorithm called Tabu search (TS). In fact, they used Tabu search to overcome the drawbacks of *K-means* including the effect of random initial centers and finding local optimum point as the final solution. But still their method is unable to find the number of clusters. Also they calculate the distances between the data points and centroids for fitness evaluation which is not good for complex datasets. The main drawback of the most methods mentioned above is, lack of ability in finding the number of clusters which is very important in big data clustering. The main novelty of this research is proposing a new accurate method based on PSO for automatic clustering.

3. Particle Swarm Optimization algorithm

Particle swarm optimization algorithm, makes use of a population (or a swarm) of members to search the solution space. The algorithm is generated by mathmatically modelling the meaningful movement of different species of the animals like birds' flocks searching for corn. Each particle, represents a possible solution for the objective function. The position of each solution is improved by the interaction with the other particles and after that evaluated using the fitness function. In addition to the position, each particle contains a vector (v) which shows its velocity. Also each particle has a memory to preserve its best position from the beginning of the process to the current iteration. In each iteration, the best solution (best particle) which has the best fitness amount is

considered as the leader of the population and the other members of the population tend to reach its position through equations number 1 and 2. Equations 1 and 2 show how particles move:

$$v_{id}^{t+1}=\text{w}.v_{id}^{t}+c_1.rand.(p_{best}^{d}-x_{id}^{t})+c_2.rand(p_{gbest}^{d}-x_{id}^{t}) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t} \quad (2)$$

where, v_{id} is the *d*th dimension of the velocity of the *i*th particle, *x* denotes the position of the particle, *t* is the number of iteration, c_1 and c_2 are learning factors, *rand* is a positive random number between 0 and 1 under normal distribution, *w* is the inertia weight coefficient, p_{best} is the best position of the particle from the beginning to current iteration and p_{gbest} shows the position of the leader in each iteration. c_1 and c_2 are two controlling factors which are called social and cognitive factors respectively. They define whether the particle moves through p_{best} or p_{gbest}. These two factors control the exploring and exploiting ability of the algorithm in searching the solution space. In fact, c_1 and c_2 are important in searching the solution space efficiently. In this paper c_1 and c_2 are exponentially changed during the search process. In the beginning iterations c_1 has a high amount while c_2 is low and in the last iterations vice versa. Using this method the algorithm can escape from local optima.

4. Proposed method

4.1. PSO-Clustering algorithm

As mentioned before many real world problems can be converted to optimization problems. Clustering is the process of dividing the data points in a dataset into different clusters based on their similarities. Based on this, a new methodology for clustering can be designed using PSO algorithm. For this purpose, the particles should be encoded suitably and also a suitable fitness function for evaluating the particles should be used. Each particle should contain the position of a specified number of clusters. In other words, each particle is a solution to the clustering problem which divides the dataset into different partitions by assigning data points to the closest centroid of the particle. So the length of each particle is $k \times p$ where k is the number of clusters and p is the number of features in the dataset. Figure 1 shows a particle with n centroids for a 2-D dataset. In this figure Cij is the *j*th dimension of the *i*th centroid.

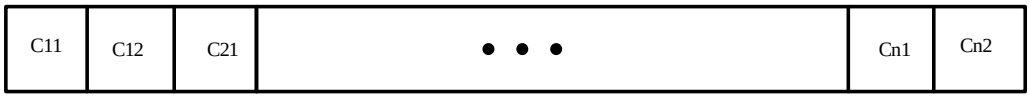


Figure 1- A particle containing n centroids for a 2-D dataset

Different clustering validity indices for measuring quality of clustering are introduced such as *Silhouette index* [22], *Davies-Bouldin index* [23], *Dunn index* 24], *Calinski-Harabasz index* [25] and etc. These indexes can be used as fitness function in the PSO algorithm. In this research

Calinski-Harabasz index is chosen due to its low complexity in compare to *Silhouette* and also its effectiveness in finding good solutions. This index calculates the quality of clustering using the following equations:

$$VRC = \frac{SS_B}{SS_W} \times \frac{(N-k)}{k-1} \quad (3)$$

$$SS_B = \sum_{i=1}^{k} n_i \|m_i - m\|^2 \quad (4)$$

$$SS_W = \sum_{i=1}^{k} \sum_{x \epsilon c_i} \|x - m_i\|^2 \quad (5)$$

In these equations, SS_B is the overall between-cluster variance, SS_W is the overall within-cluster variance, k is the number of clusters, N is the number of data points, m_i is the centroid of the ith cluster, m is the overall mean of the samples, x is a data point, c_i is the ith cluster and $\|m_i - m\|$ is the Euclidean distance between two vectors. Better solutions have higher amount of VRC. In fact, well defined-clusters have a large SS_B and a small SS_W. So finding the best solution for the clustering problem, PSO should search the solution space for a solution with the highest amount of VRC. Hence if we minimize $f = \frac{1}{VRC}$, we can find the best possible solution of the clustering problem. At the beginning of the process, a random population should be generated. Each population member (particle) contains the position of a predetermined number of clusters (k). For generating a random population, for each particle we choose k centroids from the selected samples randomly and consider them as the position of the particles. In the next step, the quality of each particle should be evaluated. For evaluating a particle, first, all of the data points in the dataset are assigned to their closest centroid, then the quality of the particle (or its fitness amount) will be calculated using *Calinski-Harabasz* index. After calculating the fitness amount for each particle and finding the leader, the particles move in the space through equations 1 and 2. For each particle, after movement a set of samples will be selected randomly from the dataset, and among them, the closest sample to each of the cluster centers of the particle, will be selected as the centroids. In fact, a new step is added to the standard PSO which makes it search the solution space for clustering problem, more effectively. The pseudo code of the algorithm is shown in Figure 2.

4.2. Automatic PSO-Clustering algorithm

Finding the number of clusters is very important in big data clustering. Generally, PSO and other swarm intelligence algorithms, are search algorithms, which search the solution space of an optimization problem. So to find the number of clusters we need to use a search algorithm. For this purpose, we added another level of searching to find the number of clusters before the *PSO-Clustering* methodology. In this level we use some tree structures of *PSO-Clustering* algorithm. Each tree consists of some nodes and each node is a *PSO-Clustering* algorithm which searches the solution space to find k centroids. At the beginning of each tree, first we generate a random population as we explained in the previous section. This population will be used for all of the other nodes in the tree. The first node will be run with a random k. the results of this node including the centroids, k and the fitness amount are considered as the *current state*.

```
Inputs: k, number of particles, maxit
Itinialization:
For i=1 to number of particles do:
        A. Generate random k positions.
        B. Select j samples from the dataset randomly.
        C. Find the closest samples to the centroids.
        D. Assign the position of the closest samples to the position of centroids.
End.
Evaluate each particle:
For i=1 to number of particles do:
        A. Assign each data point in the dataset to the closest centroids.
        B. Calculate Calinski-Harabasz index.
End.
Find the leader
Main loop:
For it =1 to maxit do:
        For i=1 to number of population do:
                1. Calculate the velocity of the particle.
                2. Update the position of each particle.
                3. Select j samples from the dataset randomly.
                4. Find the closest samples to the new position of the centroids.
                5. Assign the position of the closest samples to the position of the
                   centroids.
                6.Update the amount of C1 and C2
                7. Calculate fitness amount of the particle.
        End.
    Update Pbest and Pgbest.
End.
```

Figure 2- Pseudo code of the PSO-Clustering algorithm

Then for the next node *k* is changed using the following equation:

$$k_{new} = k_{old} \pm \varepsilon \quad (6)$$

Where ε is a random integer number. After running the *PSO-Clustering* in the next node with k_{new} and the generated population, its results will be compared with the *current state*. If the fitness amount is better, then the *current state* will be replaced with the new results. This procedure continues until the end of the tree. We run *p* number of trees, each tree starts with its own beginning population and searches the solution space for the number of clusters. After running all of the trees, the best solution among those found by different trees will be selected and used as the input of the next level which is a *PSO-Clustering* algorithm. In fact, the output of the first stage, including beginning population and *k*, is used as the input of *PSO-Clustering* algorithm in the second stage to find *k* centroids. Pseudo code of the *APSO-Clustering* algorithm is shown in figure 3.

5. Simulations and experimental results on synthetic datasets

To evaluate the effectivness of the proposed method, it is tested on 13 synthetic datasets [26] with different characteristics which are briefly indicated in table 1. Tables 2, 3, 4 and 5 indicate the average results achieved for *S*, *A*, *G2*, and *high dimensional* datasets respectively, by four different methods, *APSO-Clustering* (our method), standard PSO, *K-means* and another automatic clustering algorithm called *X-means*. *Rand index* [27] is used for measuring the accuracy. Table 3 indicates that the proposed method is very accurate and effective for datasets with high number of clusters. According to this table, *APSO-Clustering* algorithm not only gives accurate results for these kind of datasets, but also its ability in finding the number of clusters is remarkable.

```
Stage 1:
Inputs: number of sequences, length of sequence, number of population
For i=1 to number of sequences do:
        1.k = Generate a random integer number
        2. beginning population = Generate a random population
        3.Current state.fitness = inf
        4. Current state.k=Ø
        for j=1 to length of sequence do:
                if (j==1) do:
                        current state.k=k
                end
                best fitness = PSO-Clustering(k , beginning population)
                if best fitness < current state.fitness
                        current state.fitness=best fitness
                        curent state.k=k
                end
                k = current state.k ± ε
        end
Bestk[i] = current state.k
End
Output: find the best solution which has the best fitness amount and its corresponding k
and its corresponding beginning population
Stage 2:
Final output = PSO-Clustering(best k, beginning population)
```

Figure 3- Pseudo code of the *APSO-Clustering* algorithm

The proposed method overcomes *X-means* in finding the number of clusters and also in finding the position of centroids. Besides that, the accuracy of the proposed method is better than *K-means* which is a non-automatic clustering algorithm. Tables 4 and 5 demonstrate the performance of our algorithm on *high dimensional* datasets. These tables again show the power of our *APSO-Clustering* method in finding the number of centroids. According to Table 4, our two-stage method not only finds the right number of clusters for a 1024-D dataset, but also it clusters the data points with a 100 % accuracy. This is also clear in Table 5. Analyzing these tables, we can figure out that our *APSO-Clustering* algorithm is a powerful and efficient clustering method which not only outperforms *K-means* in terms of accuracy but also it has a higher accuracy in finding the number of clusters than *X-means*, which is also an automatic clustering algorithm. Besides that, these tables indicate that the *APSO-Clustering* algorithm can perform well for big data clustering. In other words, it can be a good clustering method for big datasets due to its accuracy in finding the position and number of centroids.

6. Simulations and experimental results on real big mobility datasets

We concentrate in this paper on massive real life GPS datasets, obtained from private vehicles with on-board GPS receivers. The owners of these cars are subscribers of a pay-as-you-drive car insurance contract, under which the tracked trajectories of each vehicle are periodically sent (through the GSM network) to a central server for anti-fraud and anti-theft purposes. This dataset has been donated for research purposes by Octo Telematics Italia S.r.l, the leader for this sector in Europe. In particular, the dataset used is about ≈40,000 cars tracked during 5 weeks (from June 14th through July 18th, 2011) in Tuscany, a 100 km × 100 km square centered on the city of Pisa.

Table 1- overview of the synthetic datasets

	Dataset	*Number of data points*	*Number of features*	*Number of clusters*
S datasets	*S1*	*5000*	*2*	*15*
	S2			
	S3			
	S4			
A datasets	*A1*	*3000*	*2*	*20*
	A2	*5250*	*2*	*35*
	A3	*7500*	*2*	*50*
G2 datasets	*G2-32-60*	*2048*	*32*	*2*
	G2-128-60		*128*	
	G2-256-60		*256*	
	G2-1024-70		*1024*	
High dimensional datasets	*Dim032*	*1024*	*32*	*16*
	Dim064		*64*	

The average sampling rate of the GPS receivers is 30 seconds. Globally, the dataset is composed of ≈20 Million observations, each consisting of a quadruple (*id*, *lat*, *long*, *t*), where *id* is the car identifier, (*lat*, *long*) are the spatial coordinates, and *t* the time of the observation. The car identifiers are pseudonymized, in order to achieve a basic level of anonymity. The resolution of the spatial coordinates is at 10−6 degrees, and the error of the positioning system is estimated at 10-20 m in normal conditions. All the observations of the same car *id* over the entire observation period are chained together in increasing temporal order into a global trajectory of car id. The global trajectory is then split into several sub-trajectories, corresponding to trips or travels, by using a cut-off threshold of 30 minutes: if the time interval between two subsequent observations of the car is larger than 30 minutes, the second observation is considered as start of another travel; using this reconstruction procedure we obtained ≈ 1,500,000 different travels. For the analysis we split it into 7 datasets one for each of the days.

Table 2- Average results achieved for S datasets by four different methods

Method	*S1*		*S2*		*S3*		*S4*	
	accuracy	*Number of clusters*	*accuracy*	*Number of clusters*	*accuracy*	*Number of clusters*	*accuracy*	*Number of clusters*
APSO-Clustering	***0.9984***	***17***	***0.9903***	***15.5***	***0.9658***	***16***	***0.9545***	***15.5***
X-means	*0.9161*	*8*	*0.9361*	*9*	*0.9156*	*9*	*0.9199*	*10*
K-means	*0.9901*	*-*	*0.9777*	*-*	*0.9522*	*-*	*0.9472*	*-*
Standard PSO	*0.9883*	*-*	*0.9850*	*-*	*0.9557*	*-*	*0.9503*	*-*

Table 3- Average results achieved for A datasets by four different methods

Method	*A1*		*A2*		*A3*	
	accuracy	*Number of clusters*	*accuracy*	*Number of clusters*	*accuracy*	*Number of clusters*
APSO-Clustering	***0.9975***	***20.5***	***0.9986***	***35.5***	***0.9981***	***54.5***
X-means	*0.8569*	*24*	*0.908*	*38*	*0.901*	*55*
K-means	*0.9877*	*-*	*0.9924*	*-*	*0.9949*	*-*
Standard PSO	*0.9617*	*-*	*0.9811*	*-*	*0.9889*	*-*

In particular, in the following analysis we will show the result on *Pisa_Monday* and *Pisa_Sunday* datasets which have 49,000 and 29,000 trips respectively. In this research for measuring the distance between the data points (trajectories or trips) we considered the first and last point of each trajectory. In fact, we considered each trajectory as an array with four elements including the longitude and latitude of the first and last point of each trajectory. We applied our methodology (*APSO-Clustering*) in a hierarchical form to obtain more accurate results with more details. It means that, first we group the dataset into *k* clusters using *APSO-Clustering* algorithm and in the next levels for each cluster we repeat this procedure to gain more details about the trajectories. We tested our methodology for 3 levels on the *Pisa_Monday* and *Pisa_Sunday* datasets. The achieved results are shown in Table 6. According to this table, the algorithm in the first level grouped the whole *Pisa_Monday* dataset into 2 clusters. In the next level, these 2 clusters are divided into 5 sub clusters and in the last level the sub clusters from the previous level again divided into 28 sub clusters. For *Pisa-Sunday*, the number of clusters and sub clusters for each level are 2, 10 and 47

respectively. In the next two subsections the details of the achieved results for the two datasets are described separately.

Table 4- Average results achieved for G2 datasets by four different methods

Method	***G2-32-60***		***G2-128-60***		***G2-256-60***		***G2-1024-70***	
	accuracy	*Number of clusters*	*accuracy*	*Number of clusters*	*accuracy*	*Number of clusters*	*accuracy*	*Number of clusters*
APSO-Clustering	***1***	***2***	***1***	***2***	***1***	***2***	***1***	***2***
X-means	*0.5*	*10*	*0.5*	*11*	*0.5*	*15*	*0.5*	*17*
K-means	*1*	-	*1*	-	*1*	-	*1*	-
Standard PSO	*0.8517*	-	*0.9422*	-	*0.8998*	-	*0.9095*	-

Table 5- Average results achieved for high dimensional datasets by four different methods

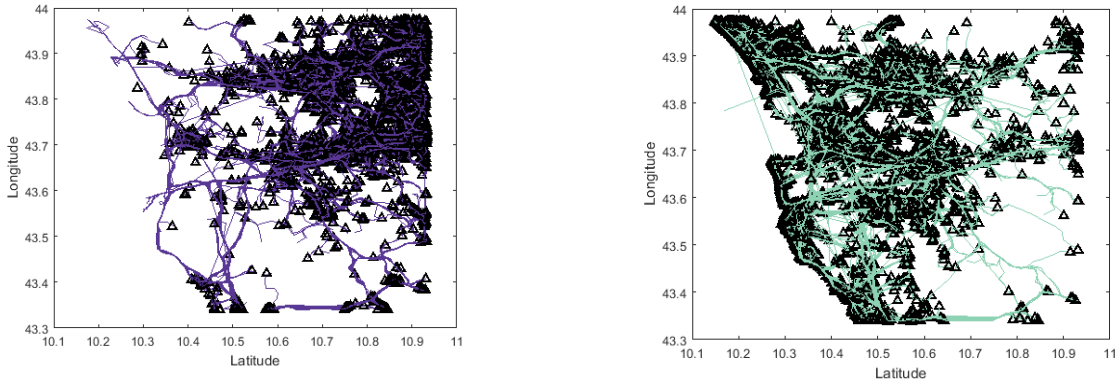
Method	***Dim032***		***Dim064***	
	accuracy	*Number of clusters*	*accuracy*	*Number of clusters*
APSO-Clustering	***1***	***16.5***	***1***	***17.5***
X-means	*0.984*	*14*	*0.984*	*14*
K-means	*1*	-	*1*	-
Standard PSO	*0.9576*	-	*0.966*	-

6.1. Results achieved for Pisa_Monday dataset

In Figure 4 a and b two clusters, detected in the first level, are shown. In this figure the lines are trajectories and the triangles define the destination of each trajectory. According to this figure generally the first cluster contains the trips to east and the second one contains the trips to west. In Figures 5 and 6, two examples of the whole sub clusters of each of these two macro clusters, are demonstrated. These sub clusters are found in the last level. As expected, the trip's destination of sub clusters shown in Figure 5 is east while the destination of the trips shown in sub clusters in Figure 6, is west. According to these figures, the algorithm finds the trips with the same destination and put them in the same cluster. The next subsection includes the corresponding results for *Pisa_Sunday* dataset.

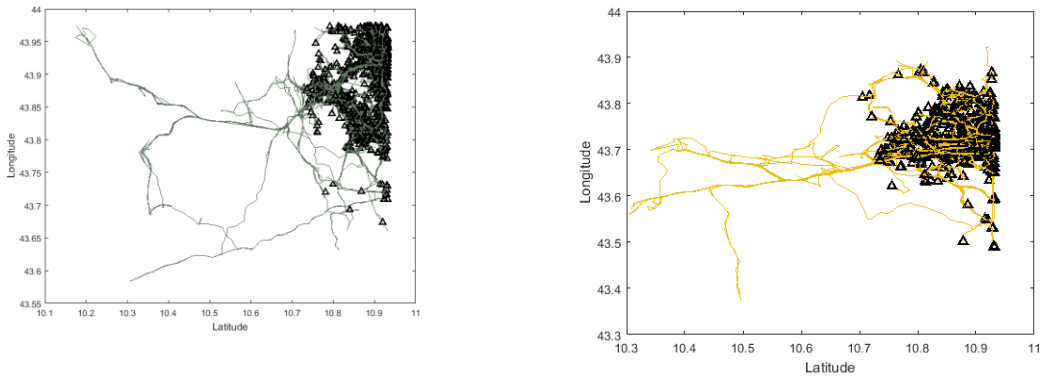
Table 6- Results achieved by hierarchical *APSO-Clustering* for *Pisa_Monday* and *Pisa_Sunday* datasets

dataset	*Number of clusters in level 1*	*Number of clusters in level 2*	*Number of clusters in level 3*
Pisa_monday	*2*	*5*	*28*
Pisa_sunday	*2*	*10*	*47*



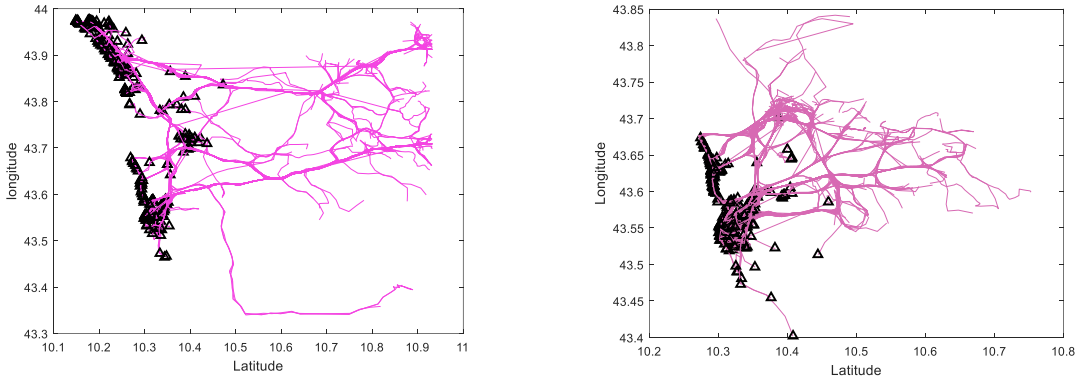
a *b*

Figure 4 a, b- the first two clusters found in the first level for *Pisa_Monday* dataset



a *b*

Figure 5- Two sub clusters extracted from the first macro cluster



a *b*

Figure 6- Two sub clusters extracted from the second macro cluster

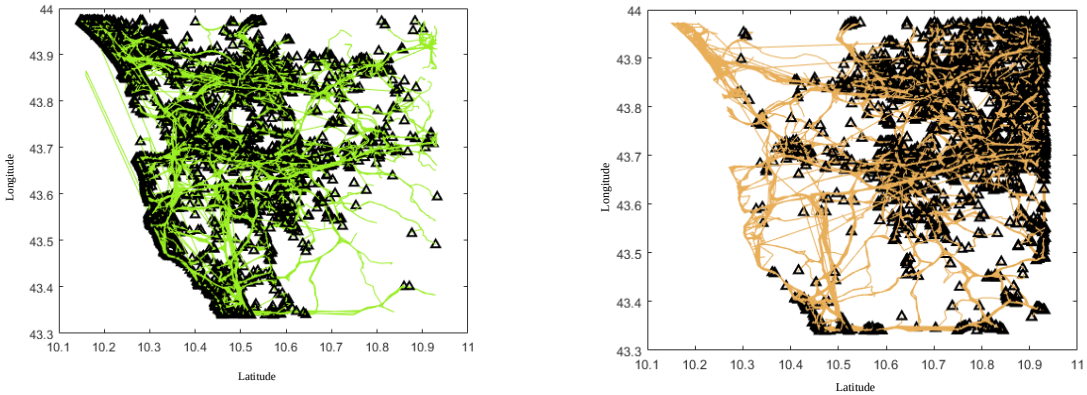
6.2. *Results achieved for Pisa_Sunday dataset*

According to Table 6 the algorithm has detected 47 clusters for this dataset after three level of searching. Like *Pisa_Monday* dataset, the algorithm has found 2 macro clusters after the first level and has divided them into 47 sub clusters at the end of the third level. Figure 7 contains the first two macro clusters for *Pisa_Sunday* dataset. In this figure, generally the first macro cluster contains the trips to west while the second one contains the trips to east. Figures 8 and 9 include 2 examples of the whole sub clusters extracted from each of the macro clusters. For this dataset the proposed method has found more clusters than the *Pisa_Monday* dataset and because Sunday is the weekend the result is reasonable. According to Figures 4 to 9, it can be seen that the power of the proposed method in finding different clusters, with different destinations is remarkable which is also demonstrated in section 5. In fact, Table 6 and Figures 4 to 9 show the effectiveness of the proposed method in big data clustering and extracting knowledge from huge amount of real data.

7. Conclusion

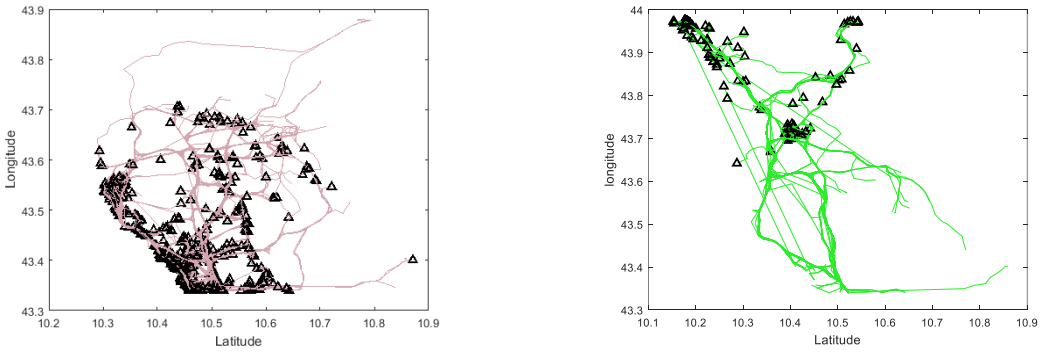
Clustering is an important data mining and big data mining technique, which is the process of dividing the objects of a dataset, into different clusters. *K-means*, which is the most popular clustering algorithm, has some drawbacks such as its tendency to converge to local optima, its dependency on the initial value of cluster centers and its inability in finding the number of clusters. These drawbacks, prevent it from performing well for big datasets with high number of features or high number of clusters. In this research we introduced a new clustering method based on a swarm intelligence algorithm (PSO) for big data clustering. we tested the proposed method on 13 synthetic datasets with different characteristics and 2 real big mobility datasets. According to the tables and figures, our *APSO-Clustering* algorithm, not only outperforms *K-means* in finding the position of the centroids, but also it finds the number of clusters accurately. Also the *APSO-*

Clustering's performance is perfect for high dimensional datasets and datasets with high number of clusters. This shows its power and effectiveness in clustering. Besides that, the results achieved for real big mobility datasets show the power and effectiveness of the proposed method when dealing with real big datasets. In another point of view, the reported results both for synthetic datasets and real big datasets demonstrate the power and accuracy of the swarm intelligence methods in solving complex optimization problems.



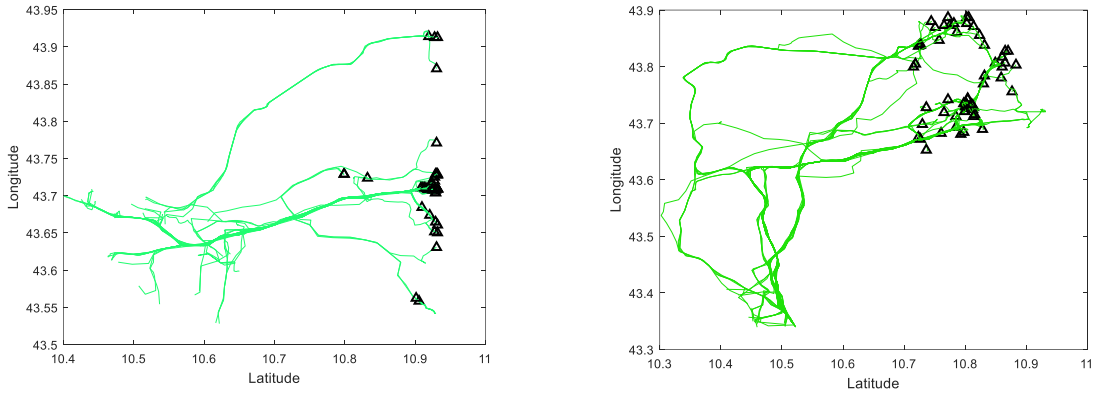
a *b*

Figure 7 a, b- The first two clusters found in the first level for *Pisa_Sunday* dataset



a *b*

Figure 8- Two sub clusters extracted from the first macro cluster



a *b*

Figure 9- Two sub clusters extracted from the second macro cluster

References

[1] Srinivasa S, Bhatnagar V, editors. Big data analytics. Proceedings of the First International Conference on Big Data Analytics BDA; 2012: Springer.

[2] Manyika J, Chui M, Brown B, Bughin J, Dobbs R, Roxburgh C, et al. Big data: The next frontier for innovation, competition, and productivity. 2011

[3] Cheng S, Shi Y, Qin Q, Bai R, editors. Swarm intelligence in big data analytics. International Conference on Intelligent Data Engineering and Automated Learning; 2013: Springer.

[4] Jain AK, Murty MN, Flynn PJ. Data clustering: a review. ACM computing surveys (CSUR). 1999;31(3):264-323.

[5] Hartigan JA, Wong MA. Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society Series C (Applied Statistics). 1979;28(1):100-8.

[6] Bezdek JC, Ehrlich R, Full W. FCM: The fuzzy c-means clustering algorithm. Computers & Geosciences. 1984;10(2-3):191-203.

[7] Cheng S, Liu B, Ting T, Qin Q, Shi Y, Huang K. Survey on data science with population-based algorithms. Big Data Analytics. 2016;1(1):3.

[8] Bonabeau E, Marco DdRDF, Dorigo M, Théraulaz G, Theraulaz G. Swarm intelligence: from natural to artificial systems: Oxford university press; 1999.

[9] Kennedy J. Particle swarm optimization. Encyclopedia of machine learning: Springer; 2011. p. 760-6.

[10] Mozaffari MH, Abdy H, Zahiri SH. IPO: an inclined planes system optimization algorithm. Computing and Informatics. 2016;35(1):222-40.

[11] Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: a gravitational search algorithm. Information sciences. 2009;179(13):2232-48.

[12] Dorigo M, Birattari M. Ant colony optimization. Encyclopedia of machine learning: Springer; 2011. p. 36-9.

[13] Razavi SH, Ebadati EOM, Asadi S, Kaur H. An efficient grouping genetic algorithm for data clustering and big data analysis. Computational Intelligence for Big Data Analysis: Springer; 2015. p. 119-42.

[14] Abraham A, Das S, Roy S. Swarm intelligence algorithms for data clustering. Soft computing for knowledge discovery and data mining: Springer; 2008. p. 279-313.

[15] Cui X, Potok TE, Palathingal P, editors. Document clustering using particle swarm optimization. Swarm Intelligence Symposium, 2005 SIS 2005 Proceedings 2005 IEEE; 2005: IEEE.

[16] Omran MG, Salman A, Engelbrecht AP. Dynamic clustering using particle swarm optimization with application in image segmentation. Pattern Analysis and Applications. 2006;8(4):332.

[17] Zhang C, Ouyang D, Ning J. An artificial bee colony approach for clustering. Expert Systems with Applications. 2010;37(7):4761-7.

[18] Karaboga D, Basturk B. On the performance of artificial bee colony (ABC) algorithm. Applied soft computing. 2008;8(1):687-97.

[19] Krishnasamy G, Kulkarni AJ, Paramesran R. A hybrid approach for data clustering based on modified cohort intelligence and K-means. Expert Systems with Applications. 2014;41(13):6009-16.

[20] Kulkarni AJ, Durugkar IP, Kumar M, editors. Cohort intelligence: a self-supervised learning behavior. Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on; 2013: IEEE.

[21] Lu Y, Cao B, Rego C, Glover F. A Tabu search based clustering algorithm and its parallel implementation on Spark. Applied Soft Computing. 2018;63:97-109.

[22] Starczewski A, Krzyżak A, editors. Performance evaluation of the silhouette index. International Conference on Artificial Intelligence and Soft Computing; 2015: Springer.

[23] Davies DL, Bouldin DW. A cluster separation measure. IEEE transactions on pattern analysis and machine intelligence. 1979(2):224-7.

[24] Pakhira MK, Bandyopadhyay S, Maulik U. Validity index for crisp and fuzzy clusters. Pattern recognition. 2004;37(3):487-501.

[25] Caliński T, Harabasz J. A dendrite method for cluster analysis. Communications in Statistics-theory and Methods. 1974;3(1):1-27.

[26] School of Computing University of Eastern Finland. clustering basic benchmarks 2015 [Available from: https://cs.joensuu.fi/sipu/datasets/.

[27] Rand WM. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association. 1971;66(336):846-50.