Improved K-means clustering on Hadoop

Kaustubh Chaturbhuj
Dept. of Computer Science and Engineering, YCCE
Nagpur, 441110, India
kaustubhs.chaturbhuj@gmail.com

Gauri Chaudhary
Dept. of Computer Science and Engineering, YCCE
Nagpur, 441110, India
chaudhary_gauri@yahoo.com

Abstract *:-* Clustering is the portioning method in which we grouped similar attribute items. Recently data grows rapidly so data analysis using clustering getting difficult. K-means is traditional clustering method. K-means is easy to implement and scalable but it suffers from local minima and sensitive to initial cluster centroids. Particle swarm optimization is mimic behavior based clustering algorithm based on particle's velocity but it suffers from number of iterations. So we use PSO for finding initial cluster center and then use this centroids for K-means clustering which is running parallel on Hadoop. Hadoop is used for large database. We try to find global clusters in limited iterations.

Keywords: *Hadoop, K-means, PSO*

*****

1. INTRODUCTION

K-means [1] is traditional partitioning method for data clustering algorithm used in data mining and data retrieval. In recent years, data generating over internet is rapidly increasing not only in size but also in variety. However processing with Big data, K-means has some limitations. K-means suffers from two drawbacks-It converges to local minima from starting position and sensitive to initial cluster centers. This drawbacks affect the effectiveness of resulting clusters such as with different initial cluster center resulting cluster of K-means are also different. So that using K-means generating global clusters of large data is big challenge.

For overcomes the drawback of K-means, many methods have been proposed. Generally this methods focus on how to optimize initial cluster centers. In [2] Juby Mathew, R Vijayakumar proposed the parallelization of k-means using firefly clustering algorithm. Firefly algorithm inspired from flashing behavior of the fireflies. In [3] K. Arun Prabha and N. Karthikayini Visalakshi proposed K-means clustering using Particle swarm optimization algorithm i.e PSO algorithm. PSO is mimic behavior based clustering algorithm. In [4] Kunhui Lin, Xiang Li, Zhongnan Zhang and Jiahong Chen proposed A K-means clustering with Optimized Initial Center. Optimized initial center based on data dimensional density. In [5] S. Mohanavalli, S.M. Jaisakthi, and C. Aravindan proposed strategies for parallelizing K-means in which they proposed hybrid MPI and openMP. In [6], Kannan Govindarajan, Thamarai Selvi Somasundaram, Vivekanandan S Kumar , Kinshuk uses PSO for enhancing the clustering strategy of K-means. In [7], M. B. Al-Daoud, and S. A. Roberts proposes two methods to optimize the initial clusters. Methods are depends on the data distribution. First method distribute cluster centroids according to the density of data objects and then second one optimize the distribution of cluster centroids. Then the Gas algorithm and PSO algorithm has been gradually introduced into K-means. In [11], new method that combine k-means and PSO is introduced which enhance previous result. In [12], uses combinations of PSO and K-means to enhance the result. They consider three combinations such as a) K-means + PSO; b) PSO + K-means; c) K-means + PSO + K-means.

To make k-means effective and efficient in clustering of large dataset, this paper proposes an improved method called PSO+ parallel K-means based on HADOOP. In this method first we apply PSO algorithm over data and find initial centers for K-means. This method will help us for finding global clusters and HADOOP will process parallel hence improve the performance.

Rest of the paper is organized as, Section 2 gives an introduction about Hadoop. Section 3 gives details of PSO and K-means. Section 4 gives details of improved K-means algorithm. Section 5 describes experimental results. Finally, the paper is concluded in 6 section.

2. HADOOP

Apache Hadoop is open source framework. Hadoop is framework used for processing Big data. Written in java and build on 4 modules. It is framework that help for distributed processing of large databases across network of computers using simple programming modules. Hadoop adopt MapReduce programing model.

2.1 Hadoop working

MapReduce is a programming framework written in JAVA which is used to process large data in parallel mode, proposed by Google and implemented in Hadoop by Apache. Its programming mechanism is to divide large dataset into numbers of splits. Size of each split is generally same as that of HDFS data blocks, by default this size is 64 MB and this assigned to a Map. Intermediate results are obtained by parallel processing of Map tasks. Reduce task combine the results on assigned nodes and produce final result. [13]

2.2 Map and Reduce

Map Reduce is programming framework propose for processing of large dataset in parallel. Map Reduce dividing the single large task into numbers of tasks. In Map task Master Node divides a problem into number of independent chunks then problems that are assign to map tasks. Each map process it and produces key-value pairs [k1, v1]. Reduce step-nodes takes the output of maps and process it. The MapReduce program which is written in the functional style are automatically parallelized. Records are treated in isolation by task called as mapper. Output from mappers is then brought together into second set of tasks called Reducers. Many map tasks run parallel on a dataset.

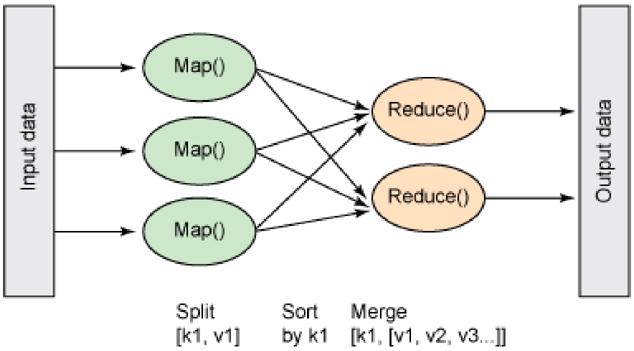

FIGURE 1: Map and Reduce [14]

3. CLUSTERING ALGORITHMS

A. K-means

K-means is traditional partitioning clustering algorithm. It is centroid based partitioning techniques uses centroid of a cluster. An Objects within cluster must have intracluster similarity and objects within different clusters have intercluster similarity. This method uses the centroid of a cluster. Centroid of cluster is its center point. Distance between object and centroid is calculated using Euclidean distance. Distance between object in cluster and object at the cluster center is squared and distance are summed. Time complexity of K-means is O (nkt) where n is no of objects k is no clusters and t is iterations. It converges to local minima from starting position and sensitive to initial cluster centers.

Algorithm: k-means. “The k-means clustering for partitioning method, where each cluster’s center is represented by the mean value of the objects in the cluster [1].

Input:
k : the number of clusters,
D: data set with n objects.
Output: set of k clusters.
Method:
(1) Arbitrarily select k objects from Data as the initial cluster centers;
(2) **repeat**
(3) (Re) assign each object to the cluster to the most similar object, based on the mean value of the objects in the cluster;
(4) calculate the mean value of the objects for each cluster;
(5) Until same results are occurred [1]

A. PSO

PSO is a mimic behavior based algorithm that the flock searching for food. Number of particles constitute a swarm. In the search space each particle moves for finding best position according to its own value as well as others. Each Personal best position i.e Pbest can be a Global best position i.e Gbest. Gbest position is the global optimal solution to the problem. This algorithm is velocity position search model. Each particle represents a position x_i (t) in n-dimension space with a velocity v_i (t). The Fitness function is set by user according to type of problem to be solved.

PSO steps-
Step 1: Initialize particles
Step 2: calculate fitness value for each particle
Step 3: if current fitness value better than p_{best} go to step 5
Step 4: keep previous p_{best} go to step 6
Step 5: Assign current fitness as new p_{best}
Step 6: Assign best particle p_{best} value to gbest
Step 7: Calculate velocity for each particle
Step 8: Use each particle’s velocity value to update its data values
Step 9: If target or max position reached go to Step 11
Step 10: go to step 2
Step 11: exit

4. Improved K-means

In this paper, we proposed method of parallel clustering based on Hadoop. In this method we use PSO and k-means clustering algorithms. In starting PSO algorithm is apply to get the initial cluster centroids roughly. Then results of PSO is use in parallel k-means clustering running on Hadoop.

A. PSO + k-means

In the PSO, current location of individual particles i.e x_i (t) = (c1, c2…, ck) denotes K cluster centroids. If the sample space is m-dimension, then each particle's position is K×m dimension. And velocity becomes K×m dimension. When PSO reaches the condition defined by user or maximum iteration, the gbest particle is consider as initial cluster centers. PSO consider initial particle i.e current position of particle for initializing process. PSO cannot be apply at data node of Hadoop since we have globally optimal centroids. If we apply PSO at data node it gives locally optimal centroid because at data node data is in split (data block) format. So we apply PSO on data set roughly before pass to Hadoop hence we get centroids for overall dataset rather than data block.

The procedure of Particle swarm optimization and parallel k-means is as follows:

Step 1: initially we run PSO on Dataset. Select K samples randomly from Dataset as the initial particle i.e $x_i(t)$. Set a random velocity $v_i(t)$ of particle and set p_i (t) = x_i (t). Number of iteration is same as number of particles.

Step 2: set ω according to condition and update it when the number of iteration increases.

Step 3: for individual particle, calculate the distance between all the samples and centroid of cluster represented by x_i (t). And then allot nearest cluster to the nearest particle.

Step 4: For individual particle, calculate new centers of the clusters then according to it update the position of the particle finally calculate fitness.

Step 5: Calculate fitness for each particle and compare with P_{best}. If current position is better global position, then update its P_{best} and personal best fitness.

Step 6: For individual particle, compare Pbest with the G_{best}. If the Pbest is better than previous then update G_{best} and global best fitness.

Step 7: update each particles velocity and position.

Step 8: go to Step2 until it reaches iteration equal to number of particles or global best position keeps unchanged for F times.

Step 9: Gbest position is consider as initial cluster centers.

B. Parallel K-means

In K-means clustering algorithm, the lengthy part is the calculations of distance between single object and all the centers of clusters. If there is given database with n-dimension samples and we required k clusters, then we have to calculate k×n distance in each loop. Consider the maximum iteration number t, then k×n×t calculations needed to complete the clustering. Serially calculation of Euclidian distance is time consuming so Map Reduce model, split this task into number of tasks and process parallel on multiple data node. If single node can complete x map tasks and the Hadoop clusters have m nodes, the total calculations required for one Map operation will be k×n×t /m×x. Hence the Euclidian distance calculations must be process in parallel on hadoop cluster. Center updated in last iteration are needed in next iteration, therefore this operation must be executed in serial manner. So the iterations are execute in serial manner and distance calculation in parallel. In each iteration. In the first iteration, get the initial centroids from the particle swarm optimization algorithm. Then execute the MapReduce program to calculate Euclidian distance, the assignment of sample objects and the update of new centers. Continue executing this iteration until it reaches the convergence criterion or reaches the maximum iteration number.

Database stored in HDFS (Hadoop Distributed File System) and centroids from last iteration of PSO are pass to Map function. First dataset is split and stored in lines. This pass to Mapper as series of <key, value> pairs where key is the line number and value for content of the line. At each Mapper, compute Euclidian distance between individual object and all the cluster centers. Then find nearest centroid of cluster to the object and assign to it. Iteration is going on till all sample objects in split is processed. Output of Mapper is in the form of list of <key, value> pairs here key represent id of the nearest center and value for sample object. The result of Mapper sorted in order and partitioned by key and then send to Reducers. It improve the efficiency of the Reducer.

The intermediate result i.e <key, value> pairs, with the same key are pass to the same Reducer. Key represent cluster id, and value for list of samples assigned to cluster. Reducer sum all the samples and samples with the same key value are collect. The new centers obtained by seeking mean value of the addition. The output <key, value> pairs, key represent cluster id, and value for k-means Cluster The result will be in the form of “clusterId” & “cluster center” & members .The result stored into Hadoop Distributed File System.

5. EXPERIMENTAL RESULTS

We implement PSO on simple java platform and find initial cluster centroids. We use VMware workstation for installing two windows on same system. Both having 1 GB ram. PSO generate centroids according to initial particles and numbers of cluster required. Hadoop use this as input and process it on two datanodes and generate K-means clustering output in sorted order since Map Reduce arrange them in sorted order. Here we use Medicare provider data set from us govt. web site. Clustering distribution is shown in figure2. Clustering output of first two hundred values in figure 3.

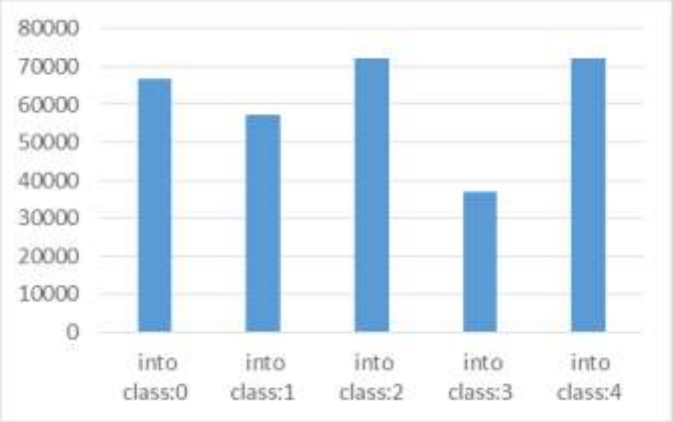

FIGURE 2: K-means Clustering distribution

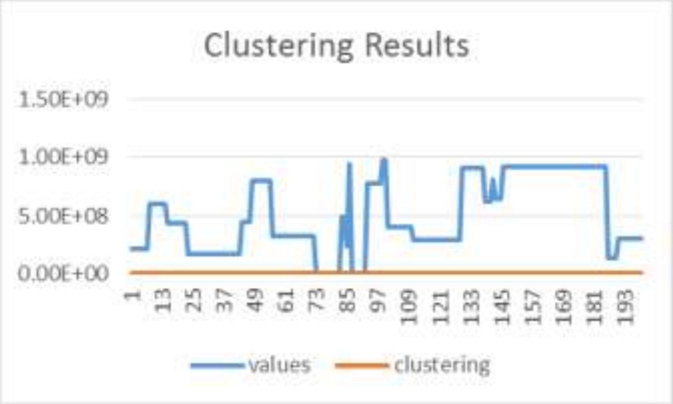

FIGURE 3: Clustering Output

Conclusion

Rapidly generating Big data is difficult to handle with traditional data mining techniques. Traditional k-means clustering is sensitive to some issues. We overcome this issues using parallel k-means clustering with Particle Swarm Optimization (PSO) and generate globally optimal clusters. Hadoop and MapReduce help us process large data parallel hence time required for clustering is reduced. Multi nodes (data nodes) help us to parallel processing and hence increase the scalability of method.

6. REFERENCES

[1] J. Han, M. Kamber, J. pei, “Data Mining concepts and techniques,” third edition.

[2] J. Mathew, R Vijayakumar, “Scalable Parallel Clustering Approach for Large Data Using Parallel K Means and Firey Algorithms”, 978-1-4799-5958-7, *IEEE*, 2014.

[3] K. Arun Prabha and N. Karthikayini Visalakshi, “Improved Particle Swarm Optimization based K-Means Clustering”, *International Conference on Intel-ligent Computing Applications, IEEE*, 2014.

[4] K. Lin, X. Li, Z. Zhang, J. Chen ,”A K-means Clustering with Optimized Initial Center Based on Hadoop Platform”, *The 9th International Conference on Computer Science and Education, IEEE*, 2014.

[5] S. Mohanavalli, S.M. Jaisakthi, and C. Aravindan, “Strategies for Parallelizing K-Means Data Clustering Algorithm”, Springer-Verlag Berlin Heidelberg, 2011.

[6] K. Govindarajan, T. Selvi Somasundaram, V. S Kumar, Kinshuk, “Continuous Clustering in Big Data Learning Analytics,” *2013 IEEE Fifth International Conference on Technology for Education.*

[7] M. B. Al-Daoud, and S. A. Roberts, “New methods for the initialization of cluster,” *Pattern Recognition Letters, vol.17, no. 5, pp. 451-455*, May 1996.

[8] B. A. Shboul, and S. H. Myaeng, “Initializing K-Means using Genetic Algorithms,” World Academy of Science, Engineering and Technology 54, 2009.

[9] K. Krishna and M. Narasimha Murty, “Genetic K-Means algorithm,” *IEEE Transactions on Systems, Man and Cybernetics—part b: cybernetics, vol. 29, no. 3*, June 1999.

[10] X. J. Sun, and X. Y. Liu, “New genetic K-means clustering algorithm based on meliorated initial center,” *Computer Engineering and Applications, vol. 44, no. 23, pp. 166-168*, 2008.

[11] A. Ahmadyfard, and H. Modares, “Combining PSO and k-means to enhance data clustering,” *International Symposium on Telecommunications*, 2008.

[12] J. J. Li, X. Yang*, Y. M. Lu, and S. T. Wu, “Survey of particle swarm clustering algorithms,” *Application Research of Computers, vol. 26, no.12*, Dec. 2009.

[13] J. Wang, D. Yuan and M. Jiang, “Parallel K-PSO Based on MapReduce,” IEEE, 2012.

[14] IBM documentation on MapReduce, “https://www-01.ibm.com/software/data/puredata/”.

[15] Apache documentation on Hadoop.