Big Data Clustering Using Genetic Algorithm On Hadoop Mapreduce

Nivranshu Hans, Sana Mahajan, SN Omkar

Abstract: Cluster analysis is used to classify similar objects under same group. It is one of the most important data mining methods. However, it fails to perform well for big data due to huge time complexity. For such scenarios parallelization is a better approach. Mapreduce is a popular programming model which enables parallel processing in a distributed environment. But, most of the clustering algorithms are not "naturally parallelizable" for instance Genetic Algorithms. This is so, due to the sequential nature of Genetic Algorithms. This paper introduces a technique to parallelize GA based clustering by extending hadoop mapreduce. An analysis of proposed approach to evaluate performance gains with respect to a sequential algorithm is presented. The analysis is based on a real life large data set.

Index Terms: Big Data, Clustering, Davies-Bouldin Index, Distributed processing, Hadoop MapReduce, Heuristics, Parallel Genetic Algorithm.

1 INTRODUCTION

Clustering [1] is a popular technique used for classifying data set into groups. Data points under particular group share similar features. It is widely used for pattern recognition, data mining etc. Many techniques have been devised for cluster analysis [12], [13] such as K-means, fuzzy c means etc. However most of the conventional techniques either compromises speed of execution for clustering accuracy or produce poor results. For instance, some clustering algorithms stuck at local optima. To achieve globally optimal solution, it requires iterating over all possible clustering. As the number of iterations is exponential in data size, for large data sets most of such techniques would fail. To tackle this we make a shift to the heuristics. Heuristics employs practical methodology to obtain near optimal solutions. Under heuristics we compromise the accuracy to achieve considerable speed ups. Instead of achieving an accurate result heuristic aims at achieving a satisfactory near optimal solution to speed up the process. Genetic algorithm [2], [3] is one such technique. It mimics the Darwinian's principal of "Survival of the fittest" to find the optimal solution in search space. However, Genetic Algorithms fail to keep up with big-data due to huge time complexity. Big data is a term used to address data sets of large sizes. Such data sets are beyond the possibility to manage and process within tolerable elapsed time. For such a scenario parallelization is a better approach.

Hadoop Mapreduce [4] is a parallel programming technique build on the frameworks of Google app engine mapreduce. It is used for processing large data in a distributed environment. It is highly scalable and can be build using commodity hardware. Hadoop mapreduce splits the input data into particular sized chunks and processes these chunks simultaneously over the cluster. It thus reduces the time complexity for solving the problem by distributing the processing among the cluster nodes. In this paper we propose a technique to implement clustering using genetic algorithm in a parallel fashion using hadoop mapreduce. To do so we extend the coarse grained parallel model of genetic algorithms and perform a two phase clustering on the data-set. This two phase clustering approach is realized by exploiting the hadoop mapreduce architecture. The rest of the paper is organized as follows in section 2 we give an overview of genetic algorithms. Section 3 explains the mapreduce model and discusses the hadoop mapreduce. Section 4 presents the technique we devised to parallelize genetic algorithm based clustering by extending hadoop mapreduce. Section 5 and 6 describe the criteria we used for deploying parallelized GA on hadoop mapreduce as well as the results of experimentation.

2 GENETIC ALGORITHMS

Genetic Algorithm is a nature inspired heuristic approach used for solving search based and optimization problems. It belongs to a class of evolutionary algorithms [10], [11]. In GAs we evolve a population of candidate solutions towards an optimal solution. GA simulates nature based techniques of crossover, mutation, selection and inheritance to get to an optimal solution. Under GA we implement the law of survival of the fittest to optimize the candidate solutions The technique of GA progresses in the following manner:

1. Initial population of candidate solutions is created
2. Each individual from the population is assigned a fitness value using appropriate fitness function
3. Parents are selected by evaluating the fitness
4. Offspring are created using reproduction operators i.e. crossover ,mutation and selection on parents
5. New population is created by selecting offspring based on fitness evaluation
6. Steps 3,4,5 are repeated until a termination condition is met

- *Nivranshu Hans is currently pursuing bachelor degree program in computer science engineering at National Institute of Technology Srinagar, India. E-mail: nivranshu36@gmail.com*
- *Sana Mahajan is currently pursuing bachelor degree program in computer science engineering at National Institute of Technology Srinagar, India. E-mail: sanamahajan09@gmail.com*
- *Dr.S N Omkar is currently the chief research scientist at Aerospace department of Indian Institute of Science. E-mail: omkar@aero.iisc.ernet.in*

3 MAPREDUCE PARADIGM

Mapreduce [5] programming paradigm involves distributed processing of large data over the cluster. Under this paradigm the input data is spitted according to the block size. The data split is performed by the input format. These splits are assigned a specific key by the record reader and thus a key, value pair is generated. Key, value pairs are then subjected to a two phase processing. This two phase processing comprises of a map phase and a reduce phase. The architecture of a basic mapreduce paradigm is depicted in figure 1. The map phase is composed of a mapper or a map routine (). Map phase is executed in the mapper of each node. The reduce phase is composed of a reducer or a reduce routine().After receiving the mapped results reducer performs the summary operations to generate final result.

MAP PHASE:

- The mapper receives the key-value pairs generated by the record reader
- The mapper performs the distributed algorithm to process the key-value pairs and generates the mapping results in form of intermediate key-value pairs
- The intermediate key-value pairs are then passed on to the reducer

REDUCE PHASE:

- The mapped results of the mapper are shuffled
- The shuffled results are then passed on to the appropriate reducer for further processing
- Combined output of all the reducers serves as the final result

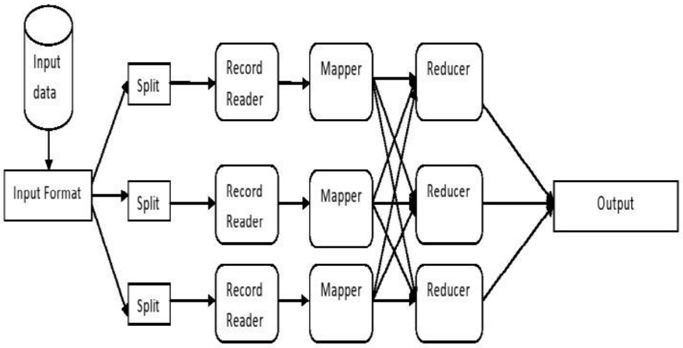

FIGURE 1

3.1 HADOOP MAPREDUCE

Hadoop mapreduce is a programming model which uses the mapreduce paradigm for processing. It is inspired by the Google app engine mapreduce. It allows for huge scalability by using commodity hardware. Mapreduce uses HDFS [6] (hadoop distributed file system) which is another component of hadoop framework for storing and retrieval of data. The processing time is reduced by splitting the data set into blocks depending upon the block size. The block size is usually 64mb or 128mb. This split data is then processed parallely over the cluster nodes. Mapreduce thus provides a distributed approach to solve complex and lengthy problems

4 PARALLEL GENETIC ALGORITHMS

In the following sections we discuss some strategies commonly used for parallelizing GA [8], [9]. Then, we propose a customized approach to implement Clustering based parallel GA on hadoop mapreduce.

Parallel implementations

Parallel implementation of GA is realized using two commonly used models as:

- Coarse-grained parallel GA
- Fine-grained parallel GA

Under first model each node is given a population split to process. The individuals are then migrated to other node after map phase. Migration is used to synchronize the solution set. In the second model each individual is given to a separate node usually for fitness evaluation. Neighboring nodes communicate with each other for selection and remaining operations.

4.1 CUSTOMIZED PARALLEL IMPLEMENTATION FOR CLUSTERING USING HADOOP MAPREDUCE

In this sub section we propose the format of GA we used for clustering based problems. Along with this we discuss our customized approach to exploit Coarse-grained parallel GA model. This approach successfully implements GA based clustering on hadoop mapreduce. Crux of this approach lies in performing a two phased clustering in mapper and then, in the reducer.To begin, the input data set is split according to the block size by the input format. Each split is given to a mapper to perform the First phase clustering. The first phase mapping results of each mapper are passed on to a single reducer to perform the Second phase mapper. We thus, are using multiple mappers and a single reducer to implement our clustering based parallel GA. The architecture of proposed model is depicted in Figure 2.

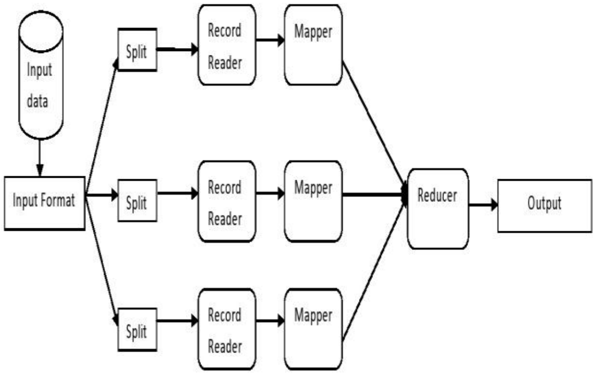

FIGURE 2

FIRST PHASE CLUSTERING:

- Population initialization:
  After receiving the input split each mapper forms the initial population of individuals. Each individual is a chromosome of size N. Every segment of the chromosome is a centroid. Centroids are randomly selected data points from the received data split. For every data point in each chromosome clustering is performed. For this data point in the received data set assigned to the cluster of the closest centroid.
- Fitness evaluation
- For evaluating fitness we are computing the Davies-Bouldin [7] index of each individual.
  Davies-Bouldin index is the ratio of inter cluster scatter to the intra cluster separation.

The inter cluster scatter of a cluster C_i is computed as

$$S_i = \frac{1}{T_i}\sum_{j=1}^{T_i}\left\|X_j - A_i\right\|_p \quad (1)$$

Here, Ai is the centroid point, X_j is the cluster point, T_i is the cluster size, p is 2 as we are calculating the Euclidian distance.
The intra cluster separation of two centroids A_i and A_j is computed as

$$M_{i,j} = \left(\left(\sum_{k=1}^{n}\left|a_{k,i} - a_{k,j}\right|^p\right)^{\frac{1}{p}}\right) \quad (2)$$

Here, k is the number of dimension of the data point and value of p is 2. Now the Davies-Bouldin index is

$$DB = \frac{1}{N}\sum_{i=1}^{N} D_i \quad (3)$$

Where Di:

$$Di = \max_{j:i\neq j}\left\{\frac{S_i+S_j}{M_{i,j}}\right\} \quad (4)$$

- **Mating & Selection:**
  For mating we are using cross-over and mutation techniques. For cross-over we are using arithmetic cross-over with 0.7% probability. This generates one offspring from two parents. The centroid of the offspring is the arithmetic average of the corresponding centroid of parents. For mutation swap mutation is applied with 0.02%. Under swap mutation we take 9's compliment of the data points. The offspring from older population are selected to populate a new population. For selection we are using Tournament selection procedure. Under tournament selection the individual is selected by performing a tournament based on fitness evaluation among several individuals chosen at random from the population.
- **Termination:**
  A new population as generated replaces the older population. This population would again form a newer population using mating and selection procedure. This whole procedure would be repeated again and again until the termination condition is met. Under the proposed approach this is achieved by completing the specified number of iterations. The fittest individual of the final population of each mapper is passed on as the result to the reducer. The reducer then performs Second phase clustering on the mapping results of all mapper.

SECOND PHASE CLUSTERING:

- Reducer forms a new chromosome by joining the chromosomes received from each mapper
- This newly created chromosome is analyzed. Those centroids for which intra cluster separation is less than the threshold, their respective clusters are merged. For two clusters the threshold is computed as sum of 20% the intra cluster separation and maximum of the largest distance of a cluster point from centroid among the two clusters. Centroid of this newly created cluster is the arithmetic mean of centroids of original clusters.
  The threshold computation:

  $$T = \left(0.2 \times M_{i,j}\right) + max\left(D_i, D_j\right)$$

  Here T is the threshold, $M_{i,j}$ is the intra cluster separation of the clusters C_i and C_j, $D_i and\ D_j$ are the distance of farthestest points of the clusters C_i and C_j from their respective centroids
- Above stated process is repeated until all centroids of the chromosome have an inter cluster separation greater than threshold value.
- The final chromosome contains location of centroid of optimal clusters.

5 EVALUATION CRITERIA & RESULTS

We compared the performance of the proposed algorithm with a sequential algorithm. Both the algorithms were executed for a total of 500 iterations with cross-over probability of 6% and mutation probability of 0.25% .The proposed algorithm was executed on a multi-node cluster with a total of 5 nodes each running hadoop v1.2.1 on an ubuntu 13.0 under vmware virtual machine with an allotted RAM of 2 GB, hard disk of 250 GB and two allotted processing cores. Hardware configuration of the cluster is shown in Table 1. The sequential algorithm was executed on a single node with configuration shown in table 2. To evaluate performance we measured the accuracy achieved and total execution time. Execution time was measured using system clock. The data set used for this experiment [] represents the differential coordinates of Europe map. It consists of 169308 instances and 2 dimensions.

HARDWARE CONFIGURATIONS (Table 1)

nodes	CPU	RAM	Hard Disk
Node 1	Intel core i3-370m	4GB DDR 3	640 GB
Node 2	Intel core i3-370m	4GB DDR 3	640 GB
Node 3	Intel core i7-2630qm	6GB DDR 3	640 GB
Node 4	Intel core i5-3230m	4GB DDR 3	1 TB
Node 5	Intel core i5-4200u	4GB DDR 3	1TB

HARDWARE SPECIFICATIONS (Table 2)

CPU	Intel core i3-370m
RAM	4GB DDR 3
Hard Disk	640 GB

Figure 3 and 4 shows the result on total execution time and accuracy achieved for the proposed algorithm and a sequential algorithm. The total execution time is highly reduced by using parallel genetic algorithm. A speed up of 80% was observed for PGA with a clustering accuracy of 92%. This shows that proposed algorithm considerably speeds up the clustering process for big datasets without compromising the accuracy to large extent.

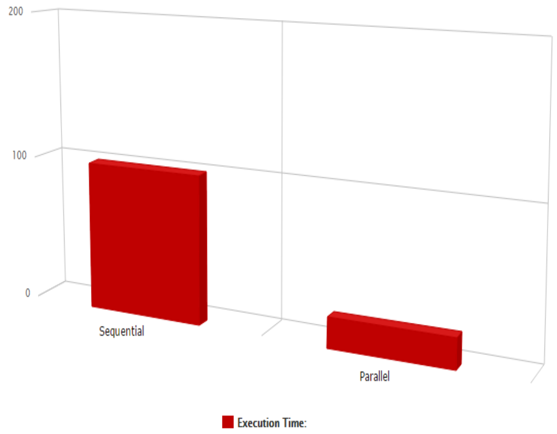

FIGURE 3

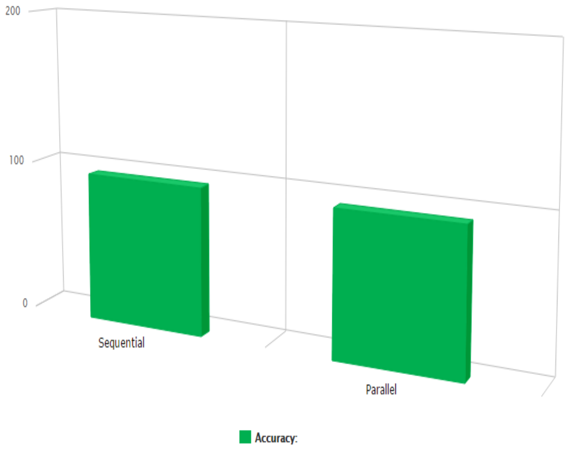

FIGURE 4

6 CONCLUSION

This Paper Introduces a Novel Technique to Parallelize GA based clustering. For this, we have Customized Hadoop Mapreduce by Implementing a Dual Phase Clustering. The speed up based on Evaluation are presented. In Future, we Hope to Improve Upon the Accuracy and Enhance the Speed Gains

ACKNOWLEDGMENT

This work was supported by Indian Institute of Science Bangalore. We thank Mr. Sukanta Roy for his guidance and support.

REFERENCES

[1] Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31, no. 3 (1999): 264-323.

[2] Bandyopadhyay, Sanghamitra, and Ujjwal Maulik. "Genetic clustering for automatic evolution of clusters and application to image classification." Pattern Recognition 35, no. 6 (2002): 1197-1208.

[3] Schaffer, J. David. "Multiple objective optimization with vector evaluated genetic algorithms." In Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh, PA, USA, July 1985, pp. 93-100. 1985.

[4] White, Tom. Hadoop: the definitive guide: the definitive guide. " O'Reilly Media, Inc.", 2009.

[5] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." Communications of the ACM 51, no. 1 (2008): 107-113.

[6] Mackey, Grant, Saba Sehrish, and Jun Wang. "Improving metadata management for small files in HDFS." In Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on, pp. 1-4. IEEE, 2009.

[7] Davies, David L., and Donald W. Bouldin. "A cluster separation measure."Pattern Analysis and Machine Intelligence, IEEE Transactions on 2 (1979): 224-227.

[8] Jin, Chao, Christian Vecchiola, and Rajkumar Buyya. "Mrpga: an extension of mapreduce for parallelizing genetic algorithms." In eScience, 2008. eScience'08. IEEE Fourth International Conference on, pp. 214-221. IEEE, 2008.

[9] Di Geronimo, Linda, Filomena Ferrucci, Alfonso Murolo, and Federica Sarro. "A parallel genetic algorithm based on hadoop mapreduce for the automatic generation of junit test suites." In Software Testing, Verification and Validation (ICST), 2012 IEEE Fifth International Conference on, pp. 785-793. IEEE, 2012.

[10] Zitzler, Eckart, and Lothar Thiele. "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach." evolutionary computation, IEEE transactions on 3, no. 4 (1999): 257-271.

[11] Zitzler, Eckart, and Lothar Thiele. "Multiobjective optimization using evolutionary algorithms—a comparative case study." In Parallel problem solving from nature—PPSN V, pp. 292-301. Springer Berlin Heidelberg, 1998.

[12] Senthilnath, J., S. N. Omkar, and V. Mani. "Clustering using firefly algorithm: performance study." Swarm and Evolutionary Computation 1, no. 3 (2011): 164-171.

[13] Kanungo, Tapas, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. "An efficient k-means clustering algorithm: Analysis and implementation." Pattern Analysis and Machine Intelligence, IEEE Transactions on 24, no. 7 (2002): 881-892.