A Novel approach for k-means++ approximation using Hadoop

Prajakta chandgude, Ashwini Bhagwat, Mayuri Autade, Anjali Pansare

Computer Engineering, Late G.N. Sapkal College of Engineering, Nashik, India.

Abstract- k-means is one of the most used clustering algorithms due to its simplicity of understanding and efficiency. However, this algorithm is mostly sensitive to the chosen initial centers and thus a proper initialization is hard for obtaining an ideal solution. To overcome this problem, k-means++ one by one chooses the centers so as to achieve a optimal solution. Due to less scalability, k-means++ is not efficient as the size of data increases. To improve its scalability and efficiency, use MapReduce along with the k-means++ method which can reduce the number of MapReduce jobs by using only one MapReduce job to obtain k centers. In this the k-means++ initialization algorithm is run in the first phase called Mapper phase and secondly the weighted k-means++ initialization algorithm is run in the Reducer phase. As this new MapReduce k-means++ method replaces the instances among multiple machines with a single machine. As this iterations are going to perform on single machine it can reduce the communication and I/O costs significant.

To reduce the complex distance computation of the proposed method, system further propose a pruning strategy that can avoid a large number of redundant distance computations.

Index Terms- Clustering algorithms, k-means, k-means++, MapReduce, approximation, scalability

I. INTRODUCTION

K means ++ is used because it gives the optimal solution in optimistic time, but it works better for the data set with small size as the data set size increases it becomes inefficient to provide optimal solution in time.So to avoid this we are using Proposed MapReduce along with this..

In this all iterations are made on single machine due to which time required for I/O communication is reduced and also complexity is also reduced To generate k centers, the MapReduce implementation of k-means++ initialization needs k rounds and 2k MapReduce jobs. In addition, large amount of data need to be transferred between multiple machines. This paper develops a k-means++ initialization algorithm in the situation with a very large amount of data by virtue of MapReduce. The major research challenges addressed are: how to implement the k-means++ initialization algorithm with MapReduce efficiently. The main idea is to use only one MapReduce job, instead of 2k MapReduce jobs.And the important to use k-means++ with MapReduce is to improve clustering of data in Hadoop bigdata.

II. SYSTEM DESCRIPTION

2.1Randomized Dimensionality Reduction for *k*-Means Clustering

Dimensionality reduction encompasses the union of 2 approaches: 1) feature choice and 2) feature extraction. A feature selection-based algorithmic rule for k-means cluster selects alittle set of the input features and so applies k-means cluster on the chosen features. A feature extraction-based algorithmic rule for k-means cluster constructs a little set of recent artificial features and so applies k-means cluster on the created features. Despite the importance of k-means cluster also because the wealth of heuristic strategies addressing it, demonstrably correct feature choice strategies for k-means cluster aren't better-known. one disadvantage of this method is it uses K-means algorithmic rule that is requires to reiterate the algorithmic rule in every machine that improves time similarly as code complexness.

2.2The Global Kernel *k*-Means Clustering Algorithm

In this work we tend to propose the global kernel k-means algorithmic rule, a specified algorithmic rule for optimizing the cluster error in feature area that employs kernel k-means as an area search procedure. The rule works in an progressive fashion by finding all issues with one, M clusters, utilizing kernel k-means, so as to resolve the M-clustering problem. the concept behind the planned methodology is that a close to optimum answer with M clusters are often obtained by beginning with a close to best answer with M-1 clusters and initializing the M-th cluster suitably supported an area search. throughout the native search the M-th cluster is initialized many times (specifically N times wherever N is that the size of the dataset) and therefore the answer with all-time low cluster error is kept because the answer with M clusters. Since the best answer for the 1-clustering problem is understood, the on top of procedure are often applied iteratively to search out a close to best answer to the M cluster downside.

2.3Divide and conquer? k-means clustering of demand data allows rapid and accurate simulations of the British electricity system.

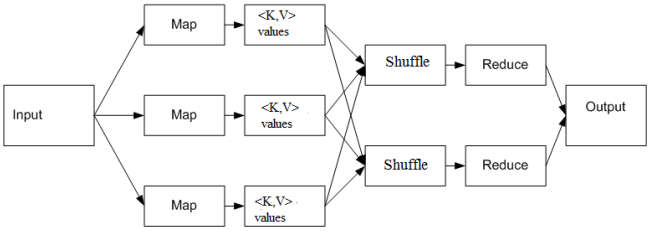
This paper presents some way of making a small range of "typical" daily profiles that may represent a year of operation, permitting a dispatch model to be run in no time and so repeatedly for random or alternative simulations. to try and do this we tend to apply a k-means cluster algorithmic rule, a technique that has been used wide in numerous disciplines to separate (often large) clustered datasets into variety of smaller teams. Membership for every cluster is determined employing a

similarity measure – during this case the euclidian distance of every information from the cluster mean. the target of the method is to assign each observation into a group in such some way that this distance is reduced. we have then investigated many ways that of exploitation the demand levels and hour-to-hour changes from each day inside every cluster to form a representative profile for that cluster. the foremost correct simulations (compared to those for a full year of data) were obtained once we took the means that of these hour-to-hour demand modifications with a similar sign because the median change inside the cluster to provide 23 hour-on-hour changes in demand, and set the remaining demand level so the entire demand across the profile equaled that inside all the cluster's observations. we discover that the clustered information provides a remarkably correct simulation for several key variables, including the common price of electricity, carbon emissions and generator revenues. alternative variables, that are a lot of passionate about extreme events, like generator start-ups (often in response to a short-lived spike in demand) are less accurately modelled, and then the strategy isn't applicable to each drawback that we would want to use it to. nevertheless, we discover that exploitation 10 clusters to represent a year of information provides a stimulating saving in pc time, and suggest that this methodology be thought of to permit the complexities of electricity dispatch to be described in studies that need several continual simulations of an electricity market.

III. PROPOSED SYSTEM

MapReduce could be a programming model and an associated implementation for process and generating massive information sets. Users specify a map operation that processes a key/value combine to get a collection of intermediate key/value pairs, and a scale back perform that merges all intermediate values related to similar intermediate key. several world tasks are representable during this model, as shown within the paper. Programs written during this operational format are automatically parallelized and executed on an outsized cluster of goods machines. The run-time system takes care of explaination of partitioning the input file, planning the program's execution across a collection of machines, handling machine failures, and managing the specified inter-machine communication. this permits programmers with none expertise with parallel and distributed systems to simply utilize the resources of an outsized distributed system. Our implementation of MapReduce runs on an outsized cluster of goods machines and is very scalable: a typical MapReduce computation processes several terabytes of information on thousands of machines. Programmers and also the system simple to use: many MapReduce programs are enforced and upwards of 1 thousand MapReduce jobs are executed on Google's clusters on a daily basis.

3.1 K-means++ ARCHITECTURE



In propose system, we are combining the initialization algorithms of MapReduce and k-means to form MapReduce K-means++. A good clustering result satisfies the condition that the distance between arbitrary two clusters should be as far as possible. Intuitively, it is a wise choice to choose the initial centers that are far away from each other in the beginning. k-means++ initialization algorithm follows this idea, but the farthest point is not always chosen to be a center. k-means++ is extremely simple and runs very fast in practice. Actually, except that the first center is chosen uniformly and randomly from the data points, each subsequent center is chosen from the remaining data points with the probability proportional to its squared distance from the existing cluster center that is closest to the point.

Map-

The data is first sent to the K-means++ algorithm for initialization then given to multiple mappers as shown in the figure above. These Mappers then convert input data into key and value pairs and then these key value pair data is then it send to the **shuffle** function which sorts data accordingly.

Reducer-

Then this sorted data is given to reducer it gives the final result.

IV. CONCLUSION

In this paper we discussed the basic concept of K-means and Mapreduce.Architecture represent the flow of data processing.

ACKNOWLEDGMENT

Our special thanks to Prof.S.B.Wagh for continuous inspiration and valuable guidance in throughout our dissertation work.

REFERENCES

[1] E. Chandra and V. P. Anuradha, "A survery on clustering algorithms for data in spatial database management systems," Comput. Appl., vol. 24, no. 9, pp. 19–26, 2011.

[2] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng, "A model-based approach to attributed graph clustering," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2012, pp. 505–516.

[3] H.-P. Kriegel, P. Kr€ oger, and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering," ACM Trans. Knowl. Discovery Data, vol. 3, no. 1, pp. 1–58, 2009.

[4] D. Moise, D. Shestakov, G. Gudmundsson, and L. Amsaleg "Indexing and searching 100 m images with map-reduce," in Proc. 3rd ACM Conf. Int. Conf. Multimedia Retrieval, 2013, pp. 17–24.

[5] J. F. Lu, J. B. Tang, Z. M. Tang, and J. Y. Yang, "Hierarchical initialization approach for k-means clustering," Pattern Recog. Lett.vol. 29, no. 6, pp. 787–795, 2008.

[6] S. Yamada, T. Onoda, and M. Sakai, "Careful seeding method based on independent components analysis for k-means clustering," J. Emerging Technol. Web Intell., vol. 4, no. 1, pp. 51–59,2012.

[7] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms, 2007, pp. 1027–1035.

[8] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassivitskii, "Scalable k-means++," Proc. VLDB Endowment, vol. 5,no. 7, pp. 622–633, 2012.

[9] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in Proc. 6th Conf. Symp. Opearting Syst. Des.Implementation—Vol. 6, 2004, pp. 10–10.

[10] W. Zhao, H. Ma, and Q. He, "Parallel k-means clustering based on mapReduce," in Proc. 1st Int. Conf. Cloud Comput., 2009, pp. 674–679.

[11] S. Papadimitriou and J. Sun, "Disco: Distributed co-clustering with map-reduce: A case study towards petabyte-scale end-to-end mining," in Proc. 8th IEEE Int. Conf. Data Mining, 2008,pp. 512–521.

[12] A. Ene, S. Im, and B. Moseley, "Fast clustering using mapReduce," in Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2011, pp. 681–689.

[13] R. L. F. Cordeiro, C. Traina, Junior, A. J. M. Traina, J. L# opez, U.Kang, and C. Faloutsos, "Clustering very large multi-dimensional data sets with mapReduce," in Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2011, pp. 690–698.

AUTHORS

First Author – Prajakta chandgude, Computer Engineering, Late G.N. Sapkal College of Engineering, Nashik, India., prajaktachandgude1@gmail.com
Second Author – Ashwini Bhagwat, Computer Engineering, Late G.N. Sapkal College of Engineering, Nashik, India., ashubhagwat00@gmail.com
Third Author – Mayuri Autade, Computer Engineering, Late G.N. Sapkal College of Engineering, Nashik, India., mayuautade00@gmail.com
Fourth Author – Anjali Pansare, Computer Engineering, Late G.N. Sapkal College of Engineering, Nashik, India., anjalipansare555@gmail.com