IJCSE International Journal of Computer Sciences and Engineering Open Access
Review Paper Volume-4, Issue-8 E-ISSN: 2347-2693

A Modified MapReduce-K-Means Clustering Based Load Distribution Method for Wireless Sensor Network in Mobile Cloud computing

Enakshmi Nandi[1],* Debabrata Sarddar[2]

[1,2] Department of Computer Science and Engineerinmg,University of Kalyani, West Bengal ,India

Available online at: www.ijcseonline.org

Received: 16/Jul/2016 Revised: 23/Jul/2016 Accepted: 18/Aug/2016 Published: 31/Aug/2016

Abstract. With the rising popularity of Cloud computing and Mobile computing, different field, such as individuals, enterprises and research centers have started outsourcing their IT and computational requirement to on-demand cloud services. Many research works have been done on the load balanced issue for wireless sensor network or WSN in Mobile Cloud Computing. For supplying fast, reliable, secure data, collected by WSN to users through cloud is an important issue in Mobile Cloud Computing. Main limitation of WSN is it has limited energy resources, thus maximizing the lifetime of the sensor node; it is required to distribute the energy dissipated throughout the wireless sensor network, which is a critical problem. There are several existing clustering based algorithm in case of load distribution such as, K-means clustering, random clustering etc. In order to minimize maintenance and maximize overall system performance distribution of load to all sensor nodes not only cluster head node is preferable job. Here we present MapReduce based K-means clustering for load distribution equally to all sensor nodes for avoiding overload condition and optimized energy consumption in WSN. MapReduced based clustering method is desirable due to its scalability and fault tolerant property. In our approach, it reads data from and writes the output back to the sensor nodes once. We also show that our proposed algorithm gives better performance in comparison with other existing load distribution approach for WSN by optimizing overload condition from cluster head node [CH] in case of wireless sensor network.

Keywords: Mobile Cloud Computing, Wireless Sensor Network, load distribution, Clustering, MapReduce, K-means.

1. Introduction

Mobile Cloud Computing (MCC) is the combination of cloud computing, mobile computing and wireless networks to provide rich computational resources to mobile users, network operators, and cloud computing providers.[1][2] The main objective of MCC is to activate execution of rich mobile applications on a plethora of mobile devices, with a rich clients experience. [3] MCC serves business opportunities for mobile network operators as well as cloud providers. [4] MCC gives better services through the channel of internet regardless of heterogeneous environment and platforms based on pay-as-you –use principle to large mobile users. There are several difficulties which have to faces in Mobile Cloud Computing, such as, Mobile Computation Offloading, Seamless Connectivity, Long WAN Latency, Mobility Management, Context-Processing, Energy Constraint, Vendor/data Lock-in, Security and Privacy etc.

In recent years, one emerging revolution is that, computers are getting too small and cheap so that single-purpose computers with embedded sensors are nearly practical from economical and theoretical aspect.**WSN** is a popular trend in this case. In WSN, there are various disadvantages; to overcome difficulties of wireless sensor networks different steps are required. The respective limitations are limited energy resources, varying energy consumption based on location, high cost of transmission, and limited processing capabilities[. All of these specifications of wireless sensor networks are totally opposites of their wired network counterparts, in which energy consumption is not any factor, transmission cost is within budget, and the network nodes have enough of processing capabilities. For maximizing the lifetime of the sensor nodes, it is required to distribute the energy dissipated throughout the wireless sensor network in order to minimize maintenance and maximize overall system performance [12].The usual topology of wireless sensor networks consist many network nodes dispersed throughout a particular physical area. An ad-hoc wireless sensor network is an autonomous system of sensor nodes in which all nodes behave as routers connected by wireless links.

Some typical applications for wireless sensors involve the collection of vast amounts of sensor data. Even, the sensors can cover mobile device functionality, such as providing input to mobile computer applications and receiving commands from mobile computers. They communicate with the user's environment with great facility such as beacons or sources of information throughout the physical space [9]

2. Load Distribution

Load distribution is the process of distributing the load among multiple resources in a system. Hence load required to be distributed over the resources in cloud-based architecture in such a way that each resource remains approximately equally loaded at any point of time. The main aim of load distribution is, to promote availability of Cloud resources and increase performance of the system. [10].The advantages of load distribution are discussed below:

a. Cost effectiveness: For receiving total improvement in system performance at sensible cost effectiveness is required.

b. Scalability and flexibility: The implemented algorithm in

Cloud Computing system the cloud computing system may change in size or topology. So the algorithm must be scalable and flexible to support these changes easily.

c. Priority: prioritization of the resources required before assigning better service to the important or high prioritized jobs despite of equal service provision for entire jobs irrespective with their origin. There are several load distributive algorithm present such as Active monitoring, RoundRobin, Throttled etc.

3. Clustering

lustering is an important area which finds application in a variety of fields including data mining, pattern recognition, image processing, chemistry, and more Given a set of patterns, the goal of clustering is to partition the input patterns into groups, known as clusters, such that similarity between patterns of a definite cluster is maximized, whereas similarity between patterns of various clusters is minimized. There are various algorithms have been developed for clustering in research area, such as K-means is one of them. In this era growth in data generation from several sources is a popular issue, even the amount of digital data will be doubled in each year ,till then gaining knowledge hidden in this large amount of digital data is a great problem [8]In this case Clustering is a powerful unsupervised learning data mining method for this solution.. K-Means is very popular, simple and robust to implement. There are some barriers of K-Means method that is users has to mention the number of clusters before start the algorithm, which creates resolution problem. The generated clusters are too sensitive to selection of initial seeds [9].Many efforts have been required to implement K-Means on different machines, some of which are based also Mapreduce.

In our paper we implement new MapReduce based K-Means Clustering algorithm for load distribution among several sensor nodes. This method already has been developed [8], but here just used this technique for load distribution among multiple sensor nodes in WSN. The main advantage of this method that it is faster and find out the number of clusters dynamically and it does not expect the user to mention about the number of clusters to be generated.

4. MapReduce method

MapReduce is activated by the map and reduce operations in functional languages, like as Lisp. This model solves computation problems through two functions called map and reduce function. All generated computational problems formulated with this method, can be parallelized automatically. At fist MapReduce model permits clients to write map/reduce components with functional-style code, next these components are compiled as a dataflow graph to explicitly definite their parallelism. Then the MapReduce runtime system schedules these components to distributed resources for execution purpose, at that time even managing multiple problems such as, parallelization, network communication, and fault tolerance. Here map function considers a key/value pair as input and creates a list of key/value pairs as output. The type of output key and value can be separate from input:

map:: $(key_1, value_1) \Rightarrow list\ (key_2, value_2)$ …… [1]

Reduce function consider a key and associated value list as input and creates a list of new values as output:

reduce :: $(key_2, list\ (value_2)) \Rightarrow list(value_3)$

……[2]

A MapReduce application is accomplished in a parallel way through two phases. In the first phase, all map operations can be executed without dependently from each other. In second phase, each reduce operation may rely on the outputs developed by any number of map operations [10]. Here entire reduce operations can be executed similar to map operations in independent way.

5. Overview of K-Means algorithm

K-means is very popular partitioned based Clustering algorithm. It selects initial cluster centers, that come randomly, and then assigns data points iteratively to them till convergence occurred. Then each point is included to the nearest center, in each iteration an optimized data has been choosing. The evaluation of this method has shown following algorithm [8]

.

Step1. Choose K center from the given data set randomly.

Step2.Calculate distance between each data point to the centers chooses using Euclidean distance formula.

Step3. Next to assign every point to its nearest center.

Step4. Calculate new center position

$v_i = [\ 1/C_i \sum x_i\ ]$[where i = 1to C_i Step5.Repeat step 3 and 4 till no change has occurred.

Step6. Stop the process.

6. Proposed Algorithm

At first initial data is divided into small blocks and distributed over individual sensor nodes. Each node performs in parallel way to process data points allotted to it. MapReduce is parallel programming with two important phases, known as map() and reduce().Here output of one map() task is not dependent to the output of other map tasks () [12].Every map() creates k clusters and k centroids. Here centroids are merged in reduce() phase on the basis of dynamically calculated threshold value. At the last stage data points in specific cluster are mapped to merged clusters created to acquire final output value. K-Means algorithm is run on every node in map() phase in parallel manner. The output value of the map() phase is set of clusters. In our method we minutely choose the final number of clusters to avoid the problem of over resolution.

The respective threshold value, which helps to determine output of merged output value of map() phase in reduce() phase is as follows

$T = 1/\ n^2\ \sum_i \sum_j d_{ij}$ for i= j…….. [3][Here i=1 to n and j=1to n,n= total no.of cluster centers created by each map() phase.]

If the distance between two or more cluster centers is less than the threshold value, then merged to become a single cluster. Here we take average distance between two cluster

centers as threshold value. The respective algorithm for map and reduce phase are mentioned below.

Algorithm for map (key, value)

Step 1: Choose k random centers[k] from D_n
Step 2: Set, centroid = center
Step 3: for i= 1 to n do
for j = 1 to k
Distance[j] = calculate Distance (Data[i],centroid[j]) endfor
Set, minDistance = min{Distance[j]}
ThenAdd D[i] to centroid[j] with minDiatance endfor

Step 4: Set,NewCentroid[i] = [∑(data ∈ clusteri)/ number of elements in clusteri]
Step 5: if (centroid! = NewCentroid)
Centroid = NewCentroid then goto step 2 endif
Step 6: Stop

*Here input has taken as data set and initial k and output consider the set of cluster of nodes.

Algorithm for reduce (key, value)

Step 1: for i to Dn_1
for each j to Dn_1 if i= j then
add i to merge[i] continue
endif
if distance(i,j)< = T then add j to merge [i] remove j from Dn_1
endif endfor
Centroid = $\sum merge_i$ / $merge_i$ then, remove i from Dn_1
endfor
Step 2: Stop

* Here consider set of centroids developed by map() and output has taken the final set of cluster nodes.

7. Results Analysis

In the following figure1 we show that final number of cluster nodes generated for allotment of different task by using our approach and in figure 2 we also show that our method gives better lifetime of sensor nodes compared with other existing Clustering algorithm. According to figure2 we say that proposed method lived 80% of its lifetime with 100% utility. Due to space constraints we are not able to include other experimental results obtained, but we did present the valuable information and results related to our algorithm, compared with other algorithm in case of equal load distribution in all cluster sensor nodes and give a better solution in this matter.

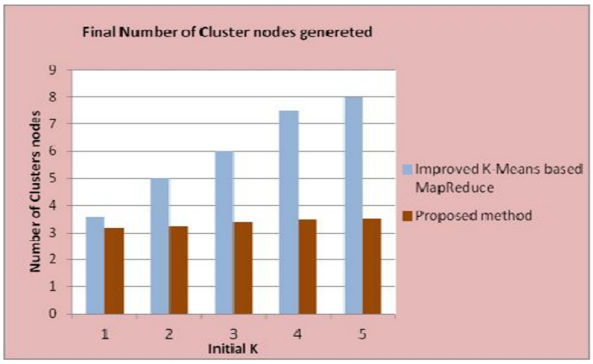

Fig 1: Final no. of Cluster nodes generated

Improved K-Means Proposed method Random Clustering Static Clustering

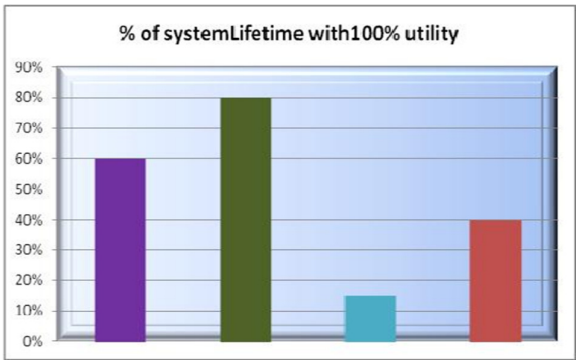

Fig: 2 Compared Sensor lifetime with other method

8. Conclusion and Future work

Through our experiments results we say that our method performed well in case of gaining its goal to distribute load throughout all sensor nodes and avoid overload condition and increase the life time of sensor nodes with great impact. Our proposed approach provides novel method for parallel load distribution and dynamically determines the number of cluster nodes required for multiple task assignment to serve fast services to mobile users. In future we would like to implement another algorithm based on this approach for handling large scale of data with proper allotment of load with respective cluster nodes, without degrading the quality of performance and apply it to real world data sets.

Acknowledgments

We want to covey our gratitude to CSIR and our institution for their great support for our research work.

References

[1]. Abolfazli, Saeid; Sanaei, Zohreh; Ahmed, Ejaz; Gani, Abdullah; Buyya, Rajkumar (1 July 2013). "Cloud-Based Augmentation for Mobile Devices: Motivation, Taxonomies, and Open Challenges". IEEE Communications Surveys & Tutorials **99** (pp): 1–32. *doi:10.1109/SURV.2013.070813.00285*.

[2]. Fangming Liu, Peng Shu, Hai Jin, Linjie Ding, Jie Yu, Di Niu, Bo Li, "Gearing Resource-Poor Mobile Devices with Powerful Clouds: Architecture, Challenges and Applications";, *IEEE Wireless Communications Magazine*, Special Issue on Mobile Cloud Computing, vol. 20, no. 3, pp.14-22, June, 2013.

[3]. Abolfazli, Saeid; Sanaei, Zohreh; Gani, Abdullah; Xia, Feng; Yang, Laurence T. (1 September 2013). "Rich Mobile Applications: Genesis, taxonomy, and open issues". Journal of Network and Computer Applications. *doi*:*10.1016/j.jnca.2013.09.009*.

[4]. A survey of mobile cloud computing: architecture, applications, and approaches, Hoang T. Dinh, Chonho Lee, Dusit Niyato andPing Wang , "Wiley online Library", inWireless communication and Mobile Computing.

[5]. Raicu, Ioan, et al. "Local load balancing for globally efficient routing in wireless sensor networks." International Journal of Distributed Sensor Networks 1.2 (2005): 163-185.

[6]. Coulouris G, Dollimore J., and Kindberg T., *Distributed Systems,Concept and Design.* AddisonWesley, 2001, pp. 116–119.

[7]. Raicu I., Richter O., Schwiebert L., and Zeadally S., "Using Wireless Sensor Networks to Narrow the Gap between Low-Level Information and Context-Awareness," ISCA Seventeenth International Conference on Computers and their Applications, 2002.

[8]. A novel MapReduced based K-Means Clustering" by Ankita Sinha and Prasanta K.Jana IEEE Senior Member, Department of Computer Science and Engineering, Indian School of Mines, Dhanbad.

[9]. Anchalia, Prajesh P. "Improved MapReduce k-Means Clustering Algorithm with Combiner." Computer Modelling and Simulation (UKSim), 2014 UKSim-AMSS 16th International Conference on. IEEE, 2014.

[10]. Jin, Chao, and Rajkumar Buyya. "Mapreduce programming model for. net-based cloud computing." Euro-Par 2009 Parallel Processing. Springer Berlin Heidelberg, 2009. 417-428.

