Parallelized K-Means Clustering Algorithm for Self Aware Mobile Ad-Hoc Networks

Likewin Thomas, Kiran Manjappa, Annappa B, G Ram Mohana Reddy
Centre for Wireless Sensor Network,
National Institute of Technology Karnataka, Surathkal
Mangalore, India

likewinthomas@gmail.com, kiranmanjappa@gmail.com, annappa@ieee.org, profgrmreddy@gmail.co

ABSTRACT
Providing Quality of Service (*QoS)* in Mobile Ad-hoc Network (MANET) in terms of *bandwidth, delay, jitter, throughput etc.,* is critical and challenging issue because of node mobility and the shared medium. The work in this paper predicts the best effective cluster while taking *QoS* parameters into account. The proposed work uses *K-Means* clustering algorithm for automatically discovering clusters from large data repositories. Further, iterative K-Means clustering algorithm is parallelized using Map-Reduce technique in order to improve the computational efficiency and thereby predicting the best effective cluster. Hence, parallel K-Means algorithm is explored for finding the best effective cluster containing the hops which lies in the best cluster with the best throughput in self aware MANET.

Categories and Subject Descriptors
B.2.1 **[Design Style]:** Parallel; B.4.1 **[Data communication Devices]:** Processors, Receivers, Transmitters [Master]; C.1.4 **[Parallel Architecture]:** Mobile Processors; C.2.1 **[Network Architecture]:** Packet Switching Network, Wireless Communication; D.1.3 **[Concurrent Programming]:** Distributed Programming, Parallel Programming; D.2.1 **[Requirement/ Specification]:** Tools: MPI; D.2.9 **[Management]**:Time Estimation; D.4.1 **[Process Management]:** Scheduling, Multiprocessing/ Multiprogramming/ Multitasking; D.4.4 **[Communication Management]:** Input/ Output, Message Sending, Network Communication; H.2.1 **[Database Application]:** Data Mining; H.3.3 **[Information Search and Retrieval]:** Clustering, Information Filtering, Query Formulation, Selection Process; H.3.7 **[Digital Library]:** Collections; H.5.2 **[User Interfaces]**: Natural Language; I.2.11 **[Distributed Artificial Intelligence]:** Intelligent Agents;

General Terms
Algorithm, Design, Experimentation, Management, Performance, Standardization, Verification, Reliability.

ICCCS'11, February 12–14, 2011, Rourkela, Odisha, India.

Keywords
K-Means, Map-Reduce, MANET, QoS.

1. INTRODUCTION
Mobile Ad-hoc networks have neither fixed infrastructure nor administrative support where as a conventional wireless network requires both fixed infrastructure and centralized administration for their operation [1]. Hence, providing Quality of Service (QoS) in MANET is critical and challenging issue. Traditional MANET routing protocols focused on finding a feasible route from a source to destination, without any consideration for optimizing the utilization of network resources or for supporting application specific QoS requirements [8-10].

To support QoS, the essential problem is to find a route with sufficient available resources, such as finding the lowest cost or most stable route that meets the QoS constraints. Further, the nodes of conventional Ad-hoc networks don't have the knowledge about its environment, i.e. context awareness. Networks which are capable of knowing the present context through online self monitoring and measurement are referred to as *Self Aware Networks* [8].

Self Awareness is achieved by *Software Agents where a*n agent can be defined as *"an autonomous entity capable of interacting with its environment and other agents".* These software agents are capable of performing sequence of activities such as *Perceiving* the observation which is present in the network, *Reasoning* about this information using existing knowledge, *Judging* the obtained information using existing knowledge and *Responding* to other *agent*s or to the external environment[7].

These software agents gives feedback about the current status of the network at every 't' seconds which can be stored in some place and can be mined to predict the best cluster so that resource rich path between the source and the sink can be predicted [7].

K-Means clustering algorithm is one of the popular clustering techniques used for minimizing the total distance between the group's members and its corresponding centroid, representative of the group by finding the best division of *n* entities in *k* groups. It mines the large databases in order to find the useful patterns between the nodes so that the effective best pair of nodes can be determined. The main purpose of clustering technique is to find the entities present in the large database belonging to the best, good or bad cluster.

K-Means algorithm is simple and effective, but it is computationally very expensive for large datasets because of its iterative nature. Hence, in order to improve the computational efficiency, in the present work, *K-Means* algorithm has been parallelized by using Map-Reduce technique for the formation of clusters and thereby determining to which cluster the pair of nodes belong to.

The parallelization paradigm used for optimizing K-Means is based on the *master-worker model* [5]. *Master* is the one who monitors the task and the *worker* is the one who does the task which has been assigned by the *master*. The *master* collects the information given by software agents and then distributes to different workers for the processing. Further, a study has been carried out by using 2, 3 and 4 machines with the combination of slow and fast processors in distributed environment and the results are analyzed.

Our main contributions are:

(i) Making self aware MANET using Software Agents, and
(ii) To find the resource rich hop in self aware MANET by using parallel K-Means algorithm and thereby improving the computational efficiency of K-Means algorithm.

The rest of the paper is organized as follows: Section 2 provides an overview of Map-Reduce technique and describes the K-Means algorithm. Results and Discussion are given in Section 3; Conclusion and Future Work are shown in Section 4.

2. *MAP-REDUCE* FOR DETERMINING CLUSTERS

Figure 1 explains the working principle of *Map-Reduce* model along with its utilization in determining clusters. *Map tasks* are referred to as group of independent tasks assigned to each worker for further processing by utilizing the information collected from software agents [3].

Then, the *master* will distribute the tasks among *workers* based on the information from software agents either in the round robin or in the serial fashion. Each *worker* will perform the *K-Means* algorithm on the information given and thereby determining the clusters as *Best, Good and Bad.*

In the *Reduce Phase* the workers will compare each entry given by the software agents against the cluster value [2] and thereby finding the hop belonging to the best, good or bad categories of clusters found during *Map-Phase*.

2.1 Master Slave Model of Parallelization: *Map-Reduce*

The parallelization paradigm that is used for optimization of resource scheduling is the *Master-Slave model*. This model is aimed at distributing the (objective function which in our term is called as task) evaluation of the individuals on several *slave* computing resources while a *master* resource executes the *optimization procedure [5]* by assigning the task to calculate the optimization of each processor. The *master–slave* model is shown in the Figure 2.

Figure. 1 Map-Reduce model.

Figure 2 Master-Slave Model

2.2 K-*Means* Clustering Algorithm

Clustering can be considered as the most important unsupervised learning in order to find a structure based on the collection of unlabeled data, in other words, clustering is the process of grouping similar data [11]. Most of the clustering algorithms in literature use some form of a distance measure of which the Euclidean being the most common distance measure. Care must be taken in choosing the distance measure that determines the correctness of the clustering to a large extent. *K-Means*, in this work, has been effectively used to determine the cluster with *best through-put*, *good through-put* and *bad through-put* based on the information collected from the software agents. Hence, during *Map-phase* of *K-Means* algorithm, clusters are identified based on the threshold value that has been set depending upon the user's requirement [6].

3. RESULTS AND DISCUSSIONS

3.1 Implementation

For parallelization, Message Passing Interface (MPI) is used (MPI is a standard for distributed memory, message passing and parallel/distributed computing). MPI is well known for its simplicity, modularity and portability that give the complete control over the parallelism to the programmer. And for the simulation of network, NS-2 simulator is used. For test environment, the network topology with 6 nodes (one source and 5 sinks) is taken as shown in Figure 3. We had the following objectives for our simulation:

1. To collect the information about the current status of different QoS parameters (Packet Delivery ratio, Throughput in Kbps, Delay in ms, Number of packets dropped, Number of packets sent and Number of packets received) through software agents which interacts with different layers of the OSI layer stack.

2. To gather the information at every 't' seconds and store the information in a file so that the master can use it for further processing.

The simulation was performed for ten minutes and the scenario consists of random movements of nodes within a 1000m x 1000m area with the specified maximum speed.

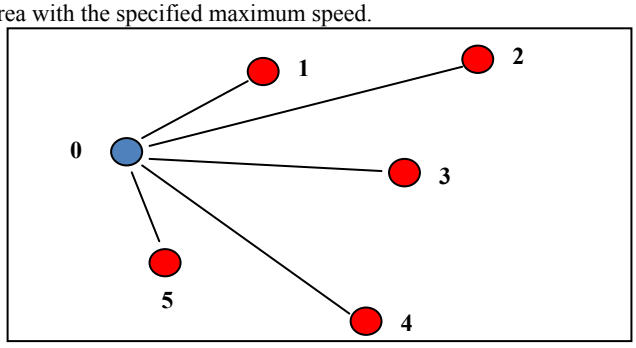

Figure 3. Network Topology

3.2 Simulation Results

Results corresponding to the information obtained by the software agents between the pair of nodes '0' and '1' are shown in Table 1. Information corresponding to other pairs obtained from software agents is not given since results were almost similar to Table 1. Here we are looking for 3 clusters namely Best, Good and Bad and the corresponding results obtained by *Map-phase* are shown in Table 2. Once the clustering is done the master reassigns the reduce phase to all workers where the worker will compare the each entries in Table 1 with the cluster average value to decide where it falls i.e. Best, Good or Bad cluster. Results of *Reduce phase* are shown in Table 3.

Table 1 Results obtained from Software Agents hops o to 1

Packet delivery ratio	*Throughput* in Kbps	*Delay* in ms	*Packet lost*	Packet Sent	Packet Received
98	1000.89	65	12	668	650
95	998.56	56	10	670	655
97	1054.93	65	6	640	629
98	980.89	75	24	658	650
94	1000.89	35	20	629	640
93	1001.01	55	22	668	650
90	968.89	85	06	635	630
93	997.89	35	14	647	650
85	890.09	55	23	640	630

Table 2 Clusters obtained using K-Means for hops (01)

	Packet delivery ratio	*Throughput* in Kbps	*Delay* in ms	*Packet lost*	Packet Sent	Packet Received
Cluster0 (Best)	97	1054.93	65	6	640	629
Cluster1 (Good)	95	989.355	70.16	14.16	661.1	647.5
Cluster2 (Bad)	90.66	962.9	41.66	19	638	640

Table 3 Reduce phase of K-Means for node '0' to All Nodes

Paths	Cluster 0 (Best)	Cluster 1 (Good)	Cluster 2 (Bad)	Remarks
0 To 1	3	5	2	GOOD
0 TO 2	3	2	5	BAD
0 TO 3	5	1	4	BEST
0 TO 4	8	1	1	BEST
0 TO 5	6	3	1	BEST

Now from above Table 3we were able to observe that 3 pairs of hops to be in best cluster ('0' to '3', '0'to '4' and '0' to '5'), now the algorithm will check the strength of each pair, hence node pair '0' and '4' is recommended.

Figure 4 shows the reading obtained when the algorithm was run for paths from source '0' to all the nodes in the cluster using 2 machines. Here we can observe even all the hops generate same size of information but the time taken for estimating the cluster is varied and it is due to the communication overhead of the network.

Figure 5 and Figure 6 shows the results obtained when the algorithm is run on 3 machines, here we have conducted 2 trails, for the purpose of experiment we took one slow running machine, in terms of processor speed, for parallelization. Trail 1 is with slow machine (X Processor) and trail 2 is without the X Processor hence we are able to notice the change in the result time because of slow machine.

Figure 7 was obtained when the algorithm was run on 4 machines and here also we are able to notice the change in the time line due to that slow running processor; here the Z Processor is the slow running one, hence we are able to see the change in time taken by that processor to estimate the efficiency of the path between '0' to '3'.

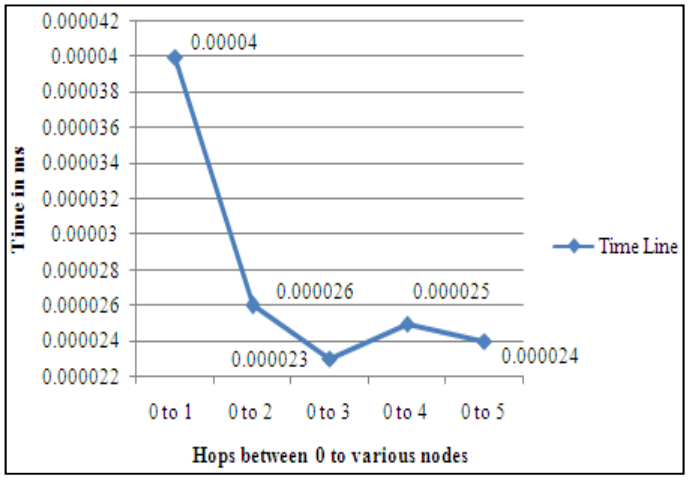

Figure 4 Result of the work on 2 machines

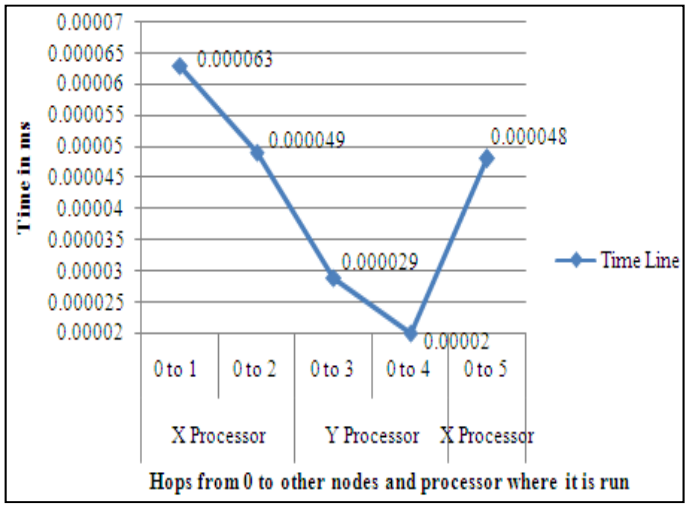

Figure 5 Result of the work on 3 machines Trail 1

Figure 8 show the speed achieved by implementing the said algorithm in more than one processor hence parallelizing the *K-Means* algorithm has generated efficient result with an efficient speed.

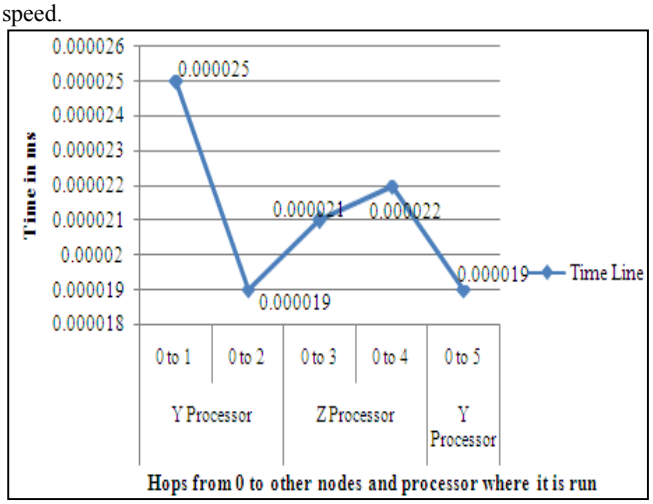

Figure 6 Result of the work on 3 machines Trail 2

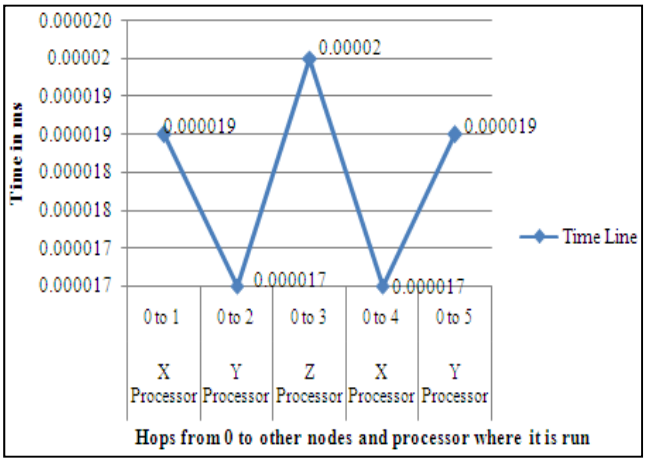

Figure 7 Result of the work on 4 machines

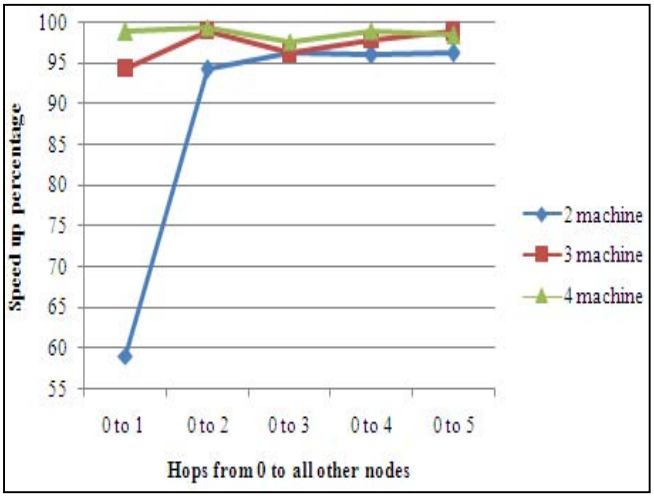

Figure 8 Speed-Up of the work

Hence from the above readings which were obtained when the algorithm was run on 2 to 4 machines, we are able to notice that the time taken for the estimation of cluster to which the pair of nodes belong to will decrease drastically as number of workers working with the master is increased.

4. CONCLUSION AND FUTURE WORK

This paper proposes an application of Map Reduce technique for parallelizing the K-Means algorithm for finding the best suitable cluster for a given pair of nodes in a Self Aware MANET. Also this paper attempts to add self awareness in MANET through software agents by interacting with the layers of the protocol stack in order to find the status of different QoS parameters. Simulation is done using single hop to identify in which cluster it falls i.e. Best, Good or Bad based on the feedback information from the software agents and results are encouraging.

To support QoS, we need to find a suitable path with sufficient available resources that meets the QoS constraints and the performance of parallel K-means algorithm is to be compared with other traditional routing protocols; will be considered for the future work.

5. REFERENCES

[1] Chakrabarti.S and Mishra. A , *QoS issues in ad hoc wireless networks* , IEEE Communication Magazine, Volume 39, Issue 2 , Feb. 2001.

[2] G.Manimaran and C.Siva Ram Murthy, Member, IEEE An Efficient Dynamic Scheduling Algorithm For Multiprocessor Real-Time Systems *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS,* VOL. 9, NO. 3, MARCH 1998 PP 312-319.

[3] Jeffrey Dean and Sanjay Ghemawat *Map-Reduce: Simplified Data Processing on Large Clusters* 0018-9162/95/ D OSDI IEEE 2004.

[4] Christopher H. Nevison *Parallel computing in the Undergraduate Curriculum* Colgate University 0018-9162/95/ D 1995 IEEE December 1995.

[5] Meira Jr., W.; Zaki, M. *Fundamentals of Data Mining Algorithms.*

[6] Hartigan, J.A. (1975) Clustering Algorithms, New York: John Wiley & Sons, Inc.

[7] Anna T. Lawniczaka, Bruno N. Di Stefanob Computational intelligence based architecture for cognitive agents *International Conference on Computational Science, ICCS* 2010 1877-0509 c 2010 Published by Elsevier Ltd A.T. Lawniczak, B.N. Di Stefano / Procedia Computer Science 1 (2010) 2227–2235.

[8] Erol Gelenbe, *Steps towards Self Aware Networks*, ACM Communications Magazine, No.7, 2009.

[9] R. Asokan , *A Review of Quality of Service (QoS) Routing Protocols for Mobile Ad Hoc Networks*, ICWCSC 2010X, IEEE, 2010.

[10] Lei Chen, Wendi B. Heinzelman , *A survey of Routing Protocols that Wupport QoS in Mobile Ad Hoc Networks* ,IEEE Network, November/December, 2007.

[11] David Pettinger, Giuseppe Di Fatta ,*Scalability of efficient Parallel K-Means*, e-Science 2009 Workshops, IEEE , 2009.