

Linux Cluster for Parallel Computing To Implement Data Mining Algorithm, K-Means with MPI

Remya O

Assistant Professor
Department of IT/CS

Pillai HOC College of Arts Science and Commerce, Rasayani
remyaomanakuttan90@gmail.com

Nirali Verma

Assistant Professor
Department of IT/CS

Pillai HOC College of Arts Science and Commerce, Rasayani

Abstract-Data clustering is one of the most popular methods for exploratory analysis of data. It has a prevalent attention in many of the disciplines like data mining, pattern classification, image segmentation, bioinformatics, statistics etc. The most famous data mining clustering algorithm is K-means, so called of its simplicity, efficiency, easiness of implementation, effectiveness etc. Impressed with the advantage of K-means clustering algorithm we introduce the K-means with MPI called MK-means through this paper. The algorithm provide a wide area to solve how to compute the huge volume of data produced as a result of real world applications, which is the major challenging issue faced by the present world. For solving the computational issues a Linux parallel Cluster is setup on the basis of distributed sharing environment. The file sharing is performed using NFS (Network file system) and parallel computing using MPD (Multi-purpose Daemon). In addition, this paper introduce MPI (message passing interface) a programming model for message passing for communication among nodes that runs parallel programs on a distributed-memory system. It is a library of routines in C programs in open-source versions MPICH2 having performance, scalability and portability. Experimental analysis and implementation proved that MK-means provide efficiency and effectiveness in the field of parallel computing clustering algorithms. It is relatively stable, portable and less time consumption in processing huge volumes of data sets.

Index Terms: Clustering, K-means, MPD, NFS, MPI, MK-Means, MPICH2, Linux Cluster.

◆

1 INTRODUCTION

With the advance of information technology the real world applications produce a huge volume of data. Thus data analysis has become a serious issue which creates complication. Therefore to find a solution on the computation of huge volume of data we use parallel computing techniques using clusters. A computer cluster is a group of identical computers networked into a small local area network with programs installed together for parallel computing. Parallel computing is made possible by three criteria job scheduling, process managing, parallel communication. The Job scheduler

decides what jobs should be executed on which nodes and process manager is responsible for the start and termination of each process. A background process named MPD is responsible for both scheduling and managing. Parallel communication is done by a parallel library. The three set of programs are integrated into a single system called master node, which is the backbone of the cluster.

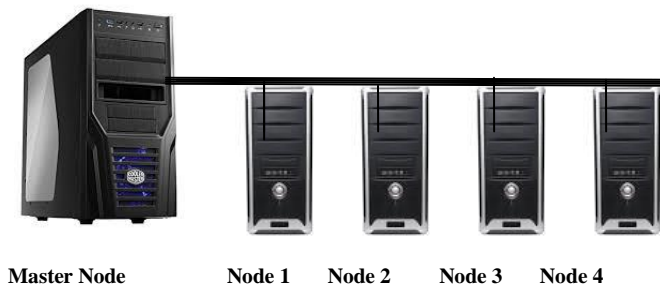


Fig1. Master and computational nodes

The cluster manages as a master node and several computational nodes. The paper proposes Linux based computers to form cluster for parallel computing in a distributed memory environment because it is cost effective, robust, virus free, high security and efficient. The benefit of the cluster is file sharing and parallel computing. File sharing is made possible by NFS which is installed and mounted in the master node. NFS is a distributed file system protocol that allows a user on a client node to access files in a computer network. Parallel computing is done by MPD; Multi-Purpose Daemon. Daemon is a Linux or UNIX program that runs in the background. It is a MPI library process management system for starting parallel jobs. MPD is a process manager has to be run on all the cluster nodes for gathering information about system, hardware and information exchange. MPI is a message-passing application programmer interface, together with protocol and semantic specifications for how its features must behave in any implementation (such as a message buffering and message delivery progress requirement) [1]. In this paper we propose to implement K-means data clustering algorithm in Linux cluster.

Data clustering is a method of exploratory data analysis which is a common unsupervised learning technique prevalent in many disciplines like data mining, image segmentation, bioinformatics, pattern recognition and statistics [2]. Clustering means the process of grouping similar objects into subsets which have a specific meaning in the context of a problem. Clustering and classification are learning methods which are different in data mining context. Clustering is unsupervised learning technique where the objects are grouped based on their features or properties whereas Classification is supervised learning technique where predefined labels assigned to instances according to various properties. Clustering

algorithms are classified as Exclusive Clustering, Overlapping Clustering, Hierarchical Clustering and Probabilistic Clustering. K-means is a well-known exclusive clustering algorithm in which data are grouped in an exclusive manner. Exclusive clustering means if certain data points belongs to a definite cluster then it could not be included in another cluster. K-mean is ranked as second top clustering algorithm in data mining by ICDM conference in 2006 for its efficiency, simplicity, effectiveness and easiness of implementation [3]. Compared to other algorithms the main advantage of K-means is efficient handling of large dataset, simplicity in implementation and based on a solid theoretical foundation greedy optimization of Voronoi partition [4]. MPI is a message passing interface which is proposed as a standard programming model for message passing in parallel computing environment. It is used to pass messages between processes on same computers and different computers using functions. MPI is an application programmer interface with protocol, semantics for how its features must behave in any implementation [1]. MPI interface provide virtual topology, synchronization and communication between a set of process in language independent way. In MPI communication is made possible using API library which consist of hundreds of function interfaces for point-to-point communication, collective operations, process groups, communication domains, process topologies etc.

The contributions in our paper are:-

- A simple Linux cluster for parallel computing using NFS, MPD, MPI.
- PK-Means, a Parallel K-means clustering algorithm for data analysis with MPI using divide and conquer methodology.
- A merging algorithm is implemented in Master node to merge clustered values and centroid values to obtain the final result.
- After implementation experimental result analysis mention that PK-Means efficient, accurate, portable, stable, relatively less execution or processing time, improved performance for large data set in clustering.

2. RELATED WORK

The related works include MPI and K-Means clustering algorithm.

2.1 MPI Functions (Message Passing Interface)

MPI is a standard popular for message passing in parallel computing. It is used to pass messages between processes in same computer and also different computers in a distributed environment. MPI is a library having variety of functions for parallel computing. The functions of MPI are

TABLE 1
MPI FUNCTIONS

MPI_Init	Start the MPI environment (Initialization)
MPI_Finalize	End the execution (Termination)
MPI_Comm_size	Access to the number of processes
MPI_Send	Send
MPI_Recv	Receive
MPI_Comm_rank	Access to the identification number of a process
MPI_Gather	Gather elements from many process to single process
MPI_Scatter	Send set of data to different processes

2.2 K-Means Clustering Algorithm

K-Means is an iterative clustering algorithm that partition the input data set into K pre defined distinct exclusive clusters. Exclusive means the data point belongs to only one group. The objective of K-means is to partition N data points into K clusters where K is always less than N. The intercluster data points are similar as possible and intracluster data points are dissimilar as possible.

How K-Means works?

- Specify the number of clusters K.
- Initialize centroids with some random and approximate data point value.
- Using Euclidean distance formula find the distance between the centroid on each cluster and

data point.

- Compare the distance and group the data point to the cluster having minimum distance.
- Calculate the new centroid of the cluster by average of data point and old centroid.
- Repeat steps 3, 4 and 5 iteratively until entities can no longer change groups.

Jain and Dubes [5] describe K-Means algorithm as follows:

1. Select an initial partition with K clusters; repeat steps 2 and 3 until cluster memberships stabilize.
2. Generate a new partition by assigning each pattern to its closest cluster center.
3. Compute new cluster centers.

The method of clustering analysis changed to an extent with the advance of technology. The variations of K-Means emerged are Fuzzy C-Means Clustering algorithms (FCM), Accelerated k-means, Constrained K-Means Clustering.

3. Proposed work

3.1 How to build a Linux cluster

Step1:

The paper proposes the installation of Linux distribution on each computer in the cluster. During the installation process, assign hostnames and unique IP addresses for each node.

One node is designated as the master node with all the other nodes used as computational slaves.

IP Address and Host naming

- 192.168.212.80 -11cpu0208L (Master Node)
- 192.168.212.83 - 11cpu0211L
- 192.168.212.81- 11cpu0209L
- 192.168.212.82-11cpu0210L

Create identical user accounts on each node. In our case, we create the user 'mpi' on each node in the cluster. Create the identical user accounts during installation, or use the adduser command as root.

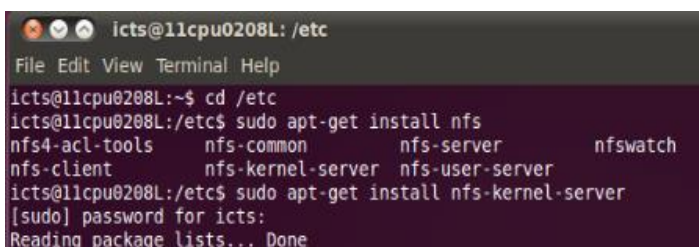
- adduser mpi

Step 2:

NFS Installation

NFS allows us to create a folder on the master node which is shared with all the other nodes in the cluster. This folder is used to store programs and data. For Installing NFS a user should run the following command in the master node's terminal:

- apt-get install nfs-kernel-server

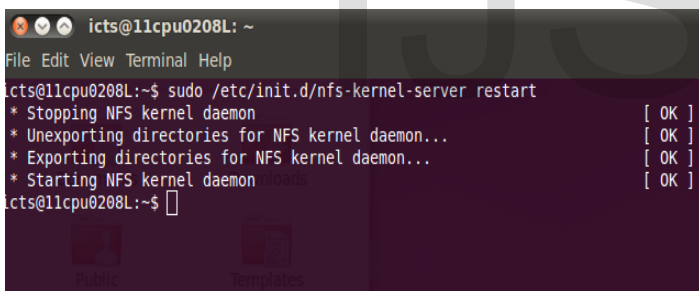


```
icts@11cpu0208L: /etc
File Edit View Terminal Help
icts@11cpu0208L:~$ cd /etc
icts@11cpu0208L:/etc$ sudo apt-get install nfs
nfs4-acl-tools      nfs-common          nfs-server          nfswatch
nfs-client          nfs-kernel-server  nfs-user-server
icts@11cpu0208L:/etc$ sudo apt-get install nfs-kernel-server
[sudo] password for icts:
Reading package lists... Done
```

Fig 2. NFS Installation

After installing nfs-kernel-server, the server should be started for making it in running mode.

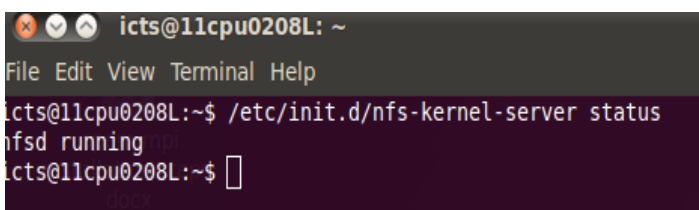
- nfs-kernel-server restart



```
icts@11cpu0208L: ~
File Edit View Terminal Help
icts@11cpu0208L:~$ sudo /etc/init.d/nfs-kernel-server restart
* Stopping NFS kernel daemon [ OK ]
* Unexporting directories for NFS kernel daemon... [ OK ]
* Exporting directories for NFS kernel daemon... [ OK ]
* Starting NFS kernel daemon [ OK ]
icts@11cpu0208L:~$
```

Fig 3. Restart NFS

The nfs-kernel status can be checked using the following command



```
icts@11cpu0208L: ~
File Edit View Terminal Help
icts@11cpu0208L:~$ /etc/init.d/nfs-kernel-server status
nfsd running
icts@11cpu0208L:~$
```

Fig 4. Checking NFS status

Step 3:

Sharing Master Folder

Make a folder in all nodes, for sharing data and programs.

mkdir /mpi

And then share the contents of this folder located on the master node to all the other nodes. In order to do this first edit the /etc/exports file on the master node. The /etc/exports file controls the exporting of files to remote hosts and specifies options.

/mpi *(rw,no_subtree_check,async) >> /etc/exports

The * denotes ip address of the machine, 192.168.212.83

Granting permission of a parent directory to all lower level directories

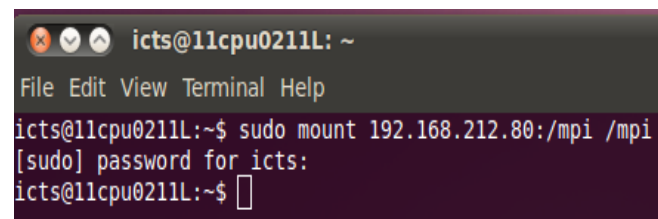
chmod +r 777 /mpi

Note than we store out data and programs only in master node and other nodes will access them with NFS.

Step 4:

Mounting in master nodes

- Mount 192.168.212.80:/mpi /mpi



```
icts@11cpu0211L: ~
File Edit View Terminal Help
icts@11cpu0211L:~$ sudo mount 192.168.212.80:/mpi /mpi
[sudo] password for icts:
icts@11cpu0211L:~$
```

Fig 5. Mount Master node

Step 5:

Installing MPICH2

- apt-get install mpich2

Step 6:

Defining a user for running MPI programs
Switch to 'mpi' user

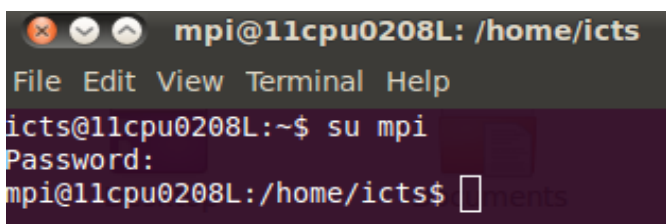


Fig 6. Switch to MPI User

Step 7:
Setting up MPD

Parallel programming is implemented using MPD (Multi purpose Daemon). Mpd is a default external process manager that comes with MPICH2. MPD provides both fast startup of parallel jobs and a flexible run-time environment. MPD hosts are created in mpiu's home directory with nodes names as mpi.

To test MPD run the following commands. The output should be the current hostname actively participating in the cluster.

- mpd &
- mpdboot -n 2
- mpdtrace

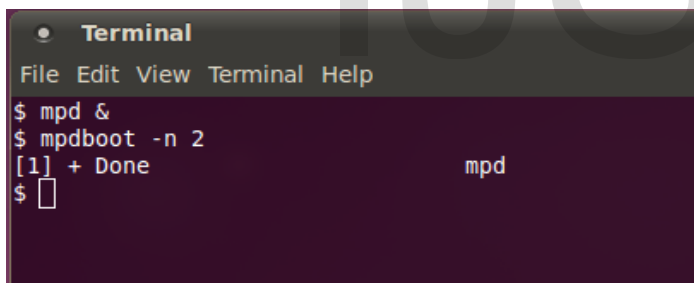


Fig 7. Run MPI

If we need more processors to actively participate in the cluster boot them using the command mpdboot.

- mpdtrace command will display the name of processors actively participating in the cluster

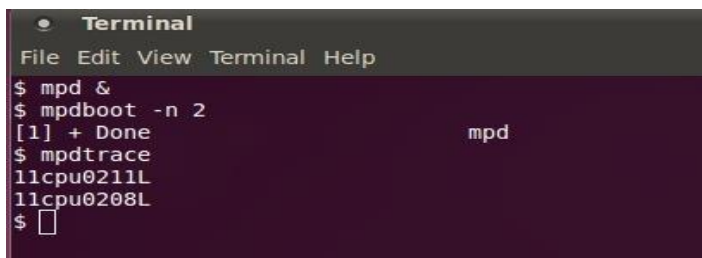


Fig 8. Trace the Nodes in Cluster

Finally the Linux Cluster is build.

3.2 Implementation of PK-means algorithm with MPI

In this paper we propose the implementation of parallel K-means algorithm using MPI known as PK-means.

3.2.1 K-means Algorithm

In K-means clustering algorithm the main idea is to partition S, data points to K clusters. Each cluster has a centre point known as Centroid, these centroids are randomly selected. The total distance between the group's members and its corresponding centroid representative of the group, is minimized.

The goal is to partition the n data points into k sets $S_i, i=1, 2... k$ to minimize the within-cluster sum of squares (WCSS), defined as:

$$D = \sum_{n=1}^N \sum_{k=1}^K |x_n - c_k|^2 \quad (1)$$

Equation for Euclidean distance

$$C = \frac{1}{N_k} \sum_{n \in c_k} x_n \quad (2)$$

Equation for Average of Centroids.

$|x_i^j - c_j|$ is the distance between the data point and the cluster's centroid.

The K-Means is an efficient computational technique and most popular representative-based clustering algorithm. Fig. 9 is the pseudo code of the K-Means algorithm.

K-Mean algorithm

Input: n data points, k clusters

Output: K centroids

1. Read n objects frm the file.
2. Randomly select k, points as the initial cluster centroids, where $C (1 \leq k \leq K)$.
3. Calculate D in equation (1) with each cluster

- centroid
4. Compare the distance; Assign the object n ($1 \leq n \leq N$) to closest cluster.
 5. Determine new centroid for cluster using equation (2)
 6. Repeat 3 to 5 until entries N , has no change
 7. Output K centroids.

Fig 9. Algorithm for K-means.

3.2.2 PK- means Algorithm with MPI

In this paper we propose K-means algorithm based on MPI with parallel computing known as PK-means. Parallel K-means is simple and portable. The algorithm starts with function, `MPI_Init()`. It is the first function called in MPI program which is used for initialization. This function is the first executable statement of MPI to start the parallel computing environment. `MPI_Finalize` function is the last statement of MPI program to end the execution of the program.

PK- means Algorithm with MPI

Input: Number of data points N , Number of clusters K

Output: K centroids, Count of data points in each cluster and data points in each cluster, N_k

1. `MPI_Init`, Starting of Parallel computing algorithm PK-means.
2. Read N data points from the file
3. Partition N data points among the processes.
4. Consider each process has N data points and install the steps 5 to 11 for each process.
5. Randomly select K points as the initial cluster centroids.
6. Calculate D in equation (1)
7. Compare the distance; Assign the object n ($1 \leq n \leq N$) to closest cluster having minimum distance.
8. Increment count_k for each cluster data point.
9. Find the new centroid of each cluster using equation (2)
10. Repeat stem 6 to 9 till the value of N has no change.
11. Create the cluster ID for each data point.
12. Generate new cluster centroids accordingly.

13. Generate a final centroid set, Centroid using a Merge function.
14. Output the clustering results: K centroids, Count and data points in each cluster.
15. `MPI_Finalize()`, To finish the parallel computing algorithm PK-means

Fig 10. PK- means Algorithm.

3.2.3 Merge function

Merge Function Algorithm

Input: n centroid sets from Centroid_1 to Centroid_n

Output: A Centroid set of K centroids

1. Initialize the centroid set, Centroid as empty.
2. If the set is empty for all, then exit and the final centroid set is Centroid. Else go to next step.
3. Find the vector of centroids (c_1 to c_k) with minimum distance in equation (2) and add c into Centroid and delete c_i from Centroid. Repeat from step 2.
4. Output K centroids.

Fig 11: Merge Function.

The Merge function is defined to find the vector of centroids with the minimum distance. Euclidean distance is used to find the distance in equation (2).

4.EXPERIMENTS

4.1 Experimental Environment

The proposed algorithm will be simulated on a cluster of 4 machines with the configuration. All the nodes were connected by a 100 Mbps Ethernet switch with UBUNTU 12.04 operating system. The hardware platform of the PCs used in cluster is Processor: Intel® Core™ i5 -3330 CPU @ 3.00GHZ 3.20GHZ, RAM: 8.00GB, Hard drive: 500GB. The software platform of the proposed system is MPICH2 to implement MPI and MPD, Java development platform for network environment LAN, NFS and a C++ compiler is required to compile and link MPI programs written in C++.

4.2 Data Sets

The breast cancer and diabetes data set selected from the UCI Machine learning Dataset Repository [6] is used as the Experimental dataset. For basic testing we use the dataset of Height and Weight of a set of people for clustering analysis.

4.3 Result and Analysis

In the experiments execution speed and clustering time are evaluated. According the number of clusters the execution time will be reduced. As the clusters are increased the sum of squared distance will be less.

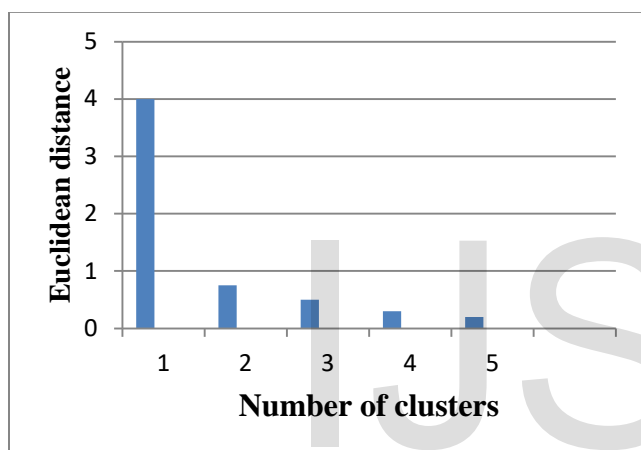


Fig12: Graph plotted between clusters and Euclidean distance between data points and cluster centroids.

5. CONCLUSIONS AND FUTURE WORK

In this paper we propose a Parallel K-means algorithm PK-means based on MPI and parallel computing using MPD. The platform for our work is provided by MPICH2 in Ubuntu 12.04. The result and analysis says that PK-means is more efficient, accurate, stable, portable, secure and minimum execution time.

In future we are planning to validate more datasets using PK-means and to port the concept to other platforms.

6. ACKNOWLEDGMENT

We express our sincere gratitude to Dr. Remya Rajesh for her valuable comments throughout the research work.

7. REFERENCES

- [1] W. Gropp, E. Lusk, Nathan Doss, Anthony Skjellum, "A High-Performance, Portable Implementation of the MPI Message Passing Interface Standard," Vol. 22, no. 6, pp. 789-828, September 1996.
- [2] Anil Jain, "Data Clustering: 50 years beyond K-means," Pattern Recognition Letters, Vol 31. No. 8, pp 651- 666, 2010.
- [3] X. Wu, Vipin Kumar, Ross, J. Ghosh, Q. Yang, H. Motoda, G. Mclachlan, A. Ng, B. Liu, P. Yu, Z. H. Zhou, M. Steinbach, David J. Hand and Dan Steinberg, "Top 10 algorithms in Data Mining," Knowledge and Information Systems, vol.14, pp. 1-37, 2008.
- [4] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," Berkeley: University of California, pp281-297, 1967.
- [5] A. Jain, R. Dubes, "Algorithms for Clustering Data" Prentice Hall, US, 1988.
- [6] C. Blake, E. Keogh, C. Merz, "UCI Repository of machine learning databases," Department of Information and Computer Science, University of California, 1998.