International Journal of Engineering and Techniques - Volume 2 Issue 3, May – June 2016

RESEARCH ARTICLE OPEN ACCESS

Multi-core Clustering of Categorical Data using Open MP

Payal More[1], Rohini Pandit[2], Supriya Makude[3], Harsh Nirban[4], Kailash Tambe[5]

[1,2,3,4,5](Department of Computer, MITCOE, Pune, India)

Abstract:

The processing power of computing devices has increased with number of available cores. This paper presents an approach towards clustering of categorical data on multi-core platform. K-modes algorithm is used for clustering of categorical data which uses simple dissimilarity measure for distance computation. The multi-core approach aims to achieve speedup in processing. Open Multi Processing (OpenMP) is used to achieve parallelism in k-modes algorithm. OpenMP is a shared memory API that uses thread approach using the fork-join model. The dataset used for experiment is Congressional Voting Dataset collected from UCI repository. The dataset contains votes of members in categorical format provided in CSV format. The experiment is performed for increased number of clusters and increasing size of dataset.

Keywords — **Raspberry Pi, MQ6 gas sensor, DS18B20 temperature sensor, Risk Management.**

INTRODUCTION

Data mining is defined Data mining is the process of extraction of non-trivial, previously unknown, implicit and potentially useful patterns from a massive amount of available data. Data mining is also called as knowledge discovery in databases (KDD), pattern analysis, data archeology, knowledge extraction, business intelligence. Data mining process comprises of four major steps -data collection, data preprocessing, data transformation and data visualization.

Clustering is the process of grouping the similar objects. The cluster is defined as a group of objects having similar features. Cluster analysis can be achieved by using various algorithms depending on the individual dataset and the required results. Clustering is an unsupervised learning approach, where class labels are not defined.

Parallel programming approach is useful to solve many complex and scientific computation problems efficiently. The multi-core processing achieves parallelism by speeding up the execution of algorithms. Parallelism can be achieved at various levels such as data level, instruction level, task level, process level, memory level. In data level parallelism the data is divided into the subsets and these subsets are assigned to the different threads. In instruction level parallelism an instruction or the set of instructions is assigned to the threads so as execute them in parallel. In this paper, we are using data level and instruction level parallelism.

Two ways for achieving parallelism are shared memory and message passing. The different computing environments require different programming paradigms depending on the problem type. OpenMP, MPI, CUDA, MapReduce are some approaches to achieve parallelism. MPI is a message passing interface which is used in distributed computing for communication between the nodes in the network. OpenMP is used on a single multi-core machine by sharing the memory between threads. CUDA is a model for multiprocessing specially designed for GPUs. MapReduce is a threaded framework where a master thread maps the tasks to slave threads and the results are reduced to a single value and then return to the master thread by the slave threads.

In our experiment we have used OpenMP as a model to achieve parallelism in k-modes clustering algorithm on multi-core platform.

CLUSTERING TECHNIQUE

Clustering can be defined as a way of grouping objects such that objects present in same group are similar. Algorithm design is an important step in the clustering process. Various algorithms are available for clustering of data. Selection of an algorithm is dependent on the input provided and output results required.

The algorithm we have selected for our experiment is k-modes algorithm. The input provided in our experiment is Congressional Voting data from UCI repository in categorical format. K-modes algorithm is advancement to K-means algorithm which uses simple dissimilarity measure for distance computation on categorical data. Also, it provides scope to achieve parallelism in distance computation of nodes as it is independent of other nodes. Thus, we have selected k-modes algorithm for our experiment

OPENMP

The OpenMP is standard for directive based parallel programming. It is a collection of APIs for developing shared memory parallel applications. OpenMP uses C, C++ and FORTRAN programming languages. OpenMP consists of three primary API components; these are compiler directives, runtime library routines and environmental variables. OpenMP is abbreviation for Open Multi-/processing.

The OpenMP system provides APIs supports for dynamic decisions to execute different parallel regions. OpenMP supports Fork-Join model for parallel execution. This model allows a process to create multiple execution thread for parallel regions. In the Fork-Join Model the fork () construct is used to create multiple threads in any parallel regions and join () is used to combine all created threads to the main thread called as master thread.

The OpenMP implementations for Java programming are JOMP, JaMP, pyjama and omp4j [17].

The JOMP is used to implements the OpenMP API for Java.

JaMP is an OpenMP implementation used in Jackal DSM that distributes Java parallel processes onto a cluster.

omp4j is an open-source OpenMP preprocessor for Java.

For our experiment we have used omp4j. The omp4j is user friendly and easy to use.

MULTI-CORE CLUSTERING MODEL

The model for multi-core implementation of OpenMP is an implementation of multithreading approach. The model for multi-core implementation of clustering algorithm on a voting database uses the concept of thread in a multiprocessing environment. The OpenMP framework has a mapping of one thread per core. The model is shown below:

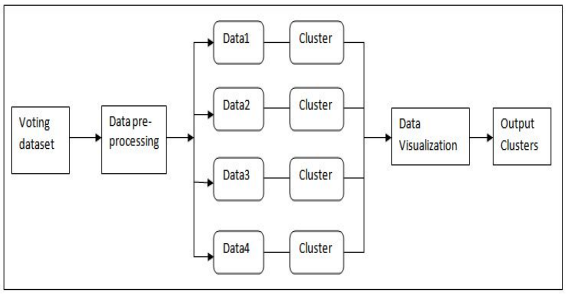

Fig.1. Multi-core Clustering Model

The input is voting database of members in a house. The dataset is collected from UCI repository. The model uses data level parallelism. The dataset is pre-processed and the divided into chunks. The data chunks are provided to each core for further processing. The clustering algorithm selected here is k-modes clustering. K-modes clustering need a similarity measure for clustering of data. The similarity measure selected is Hamming Distance measure. On each data chunk the clustering operation is performed simultaneously. The result from different core is provided for data visualization. The output is cluster of members with similar votes.

A. Dataset

The dataset for this model is Congressional dataset-Voting database. The database is available on UCI repository [1].The Congressional Dataset [1] is available on UCI repository. The dataset consists of votes of individuals in the house on different issues. The set contains of categorical values in CSV format.

Attributes in dataset [1]:

1. Class Name: 2 (Democrat, Republican)
2. Handicapped Infants: 2 (Y, N)
3. Water Project Cost Sharing: 2 (Y, N)
4. Adoption of Budget Resolution: 2 (Y, N)
5. Physician Fee Freeze: 2 (Y, N)
6. El Salvador Aid: 2 (Y, N)
7. Religious Groups In School: 2 (Y, N)
8. Anti-satellite Test Ban: 2 (Y, N)
9. Aid to Nicaraguan Contras: 2 (Y, N)
10. Max missile: 2 (Y, N)
11. Immigration: 2 (Y, N)
12. Sinful Corporation Cutback: 2 (Y, N)
13. Education Spending: 2 (Y, N)
14. Superfund Right to Sue: 2 (Y, N)
15. Crime: 2 (Y, N)
16. Duty Free Exports: 2 (Y, N)
17. Export Administration Act South Africa: 2 (Y, N)

B. Data Processing

The missing value present in the congressional dataset need to be handled effectively. The missing value in the dataset is denoted as "?". It is important to recognize the "?" in this dataset does not mean that the value of the attribute is unknown. It means simply, that the value is "yes" or "no".

C. Distance Computation

The similarity measure studied [2] are Cosine Similarity, Chi-Square, Overlap. For clustering on categorical data the clusters are formed on the similarity of values for the attributes. Thus similarity measure plays an important role. The simple the similarity measure faster it runs Distance measure needs to be selected taking into account the time complexity of algorithm. The k-modes showed better performance than hierarchical clustering for mushroom, vote, and cancer dataset. K-mode is implemented using WEKA while hierarchical using Java platform [2].

The distance measure selected for k-modes is the Hamming distance measure which is a simple dissimilarity measure. The d(X, Y) gives the distance between a node X and node Y whereas x_j, y_j are values for attribute j for X and Y respectively and d(X, Y) can be computed as [10]:

$$d(X,Y) = \sum_{j=1}^{m} \partial(x_j, y_j) \qquad (1)$$

Where,

$$\partial(x_j, y_j) = 0 \quad \text{If } x_j = y_j$$
$$\partial(x_j, y_j) = 1 \quad \text{If } x_j \neq y_j$$

Where, $\partial(x_j, y_j)$ is variable indicating number of attributes mismatched between X and Y. The lower value of distance measure indicates the two nodes are similar and can be placed in same cluster. The distance computation of one node does not interfere with other, it is independent. In parallel, k-modes

this can be done parallel using threads available on the cores. The distance function is assigned to threads to work independently.

K-MODES CLUSTERING ALGORITHM

The K-modes is an advance for k-means algorithm. It uses dissimilarity measure or hamming distance for categorical data objects. To cluster a categorical data set X into k clusters, [16] the k-modes clustering process consists of the following steps:

Step 1: Randomly select k unique objects as the initial cluster centers (modes).

Step 2: Calculate the distances between each object and the cluster mode;

Assign the object to the cluster whose center has the shortest distance to the object;

Repeat until all objects are assigned.

Step 3: Select a new mode for each cluster and compare it with the previous mode.

If different, go back to Step 2; otherwise, stop

PARALLEL K-MODES ALGORITHM

The parallel k-modes algorithm makes use of omp4j directives supported to achieve parallelism. In k-modes the distance computation of one data point is independent of other data point. Thus, it allows parallel implementation. The data level parallelism is achieved in our algorithm. The algorithm is as follows:

Step 1:

//omp parallel for

Randomly select k unique objects as the initial cluster centers (modes).

Step 2:

//omp parallel for

Calculate the distances between each object and the cluster mode;

//omp critical

Assign the object to the cluster whose center has the shortest distance to the object;

Repeat until all objects are assigned.

Step 3:

//omp parallel for

Select a new mode for each cluster and compare it with the previous mode.

If different, go back to Step 2; otherwise, stop

RESULTS AND ANALYSIS

The performance measures for multi-core implementation of clustering algorithm on categorical data are speedup. Speedup is defined as ratio of implementation of serial to parallel processors. Multi-core implementation aims at increasing the efficiency of an algorithm. We have performed experiment to analyze the trend in execution time values for serial and parallel algorithms. The tables below provide the values for serial and parallel execution over number of instances and number of clusters respectively.

Table 1: Execution Time Comparison for k=10

Sr.No.	Dataset Instances	Serial Time(in ms)	Parallel Time(in ms)
1	1000	861	767
2	2000	1001	767
3	3000	955	876
4	4000	1376	1111
5	5000	1236	1048
6	6000	1142	1111
7	7000	1204	955
8	8000	1314	1298
9	9000	1485	1392
10	10000	1828	1579

The above table provides the values for execution time for serial implementation and parallel implementation of k-modes algorithm on different instances of dataset with k value 10. The table gives us detail that the execution time difference between serial and parallel varies with the number of instances in the dataset.

Table 2: Execution Time Comparison for Dataset with 10000 instances

Sr.No.	Number of Clusters	Serial Time(in ms)	Parallel Time(in ms)
1	10	1828	1579
2	20	1033	845
3	30	1860	1813
4	40	2140	1766
5	50	1968	1907
6	60	2016	1984
7	70	1968	1973
8	80	2483	1922
9	90	2749	2000
10	100	2172	2047

The above table provides the values for execution time for serial implementation and parallel implementation of k-modes algorithm on different k values on dataset with 10000 instances. The values in the table provide us detail, speedup is achieved even with varied number of clusters over dataset with 10000 instances.

Thus the results in the table concludes that the parallel k-modes algorithm provides speedup and also achieves scalability for increased number of clusters and increased number of dataset instances.

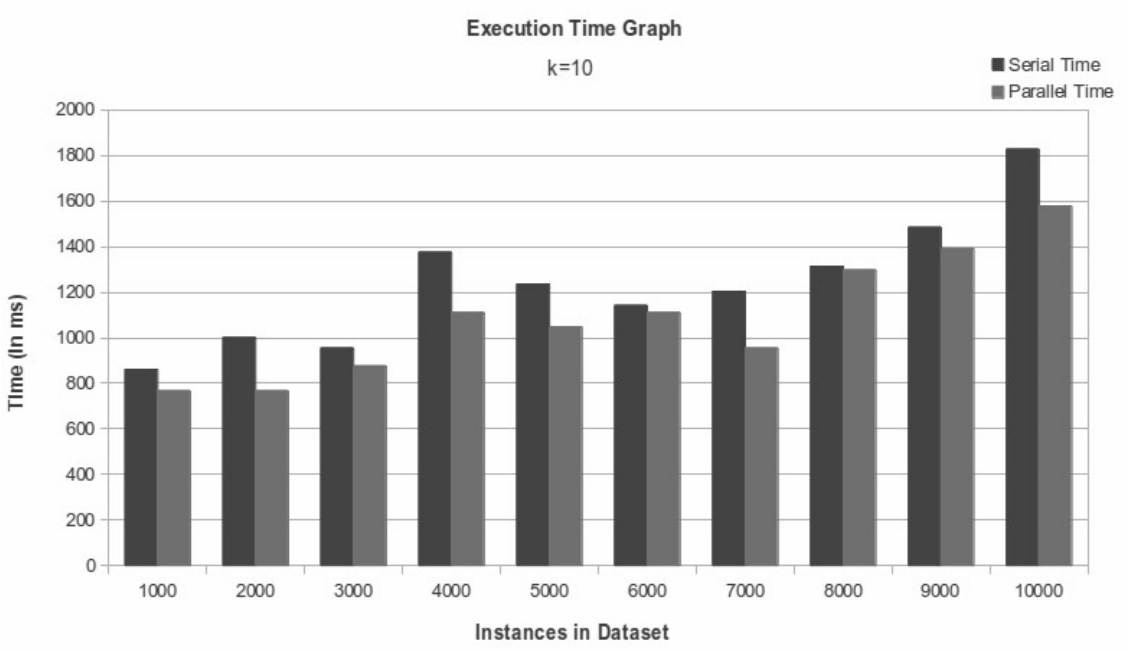

Fig.2. Execution Time graph for k=10

The above is a graph for values in Table 1. The execution time values are provided for k=10 over dataset instances varying from 1000 to 10000.

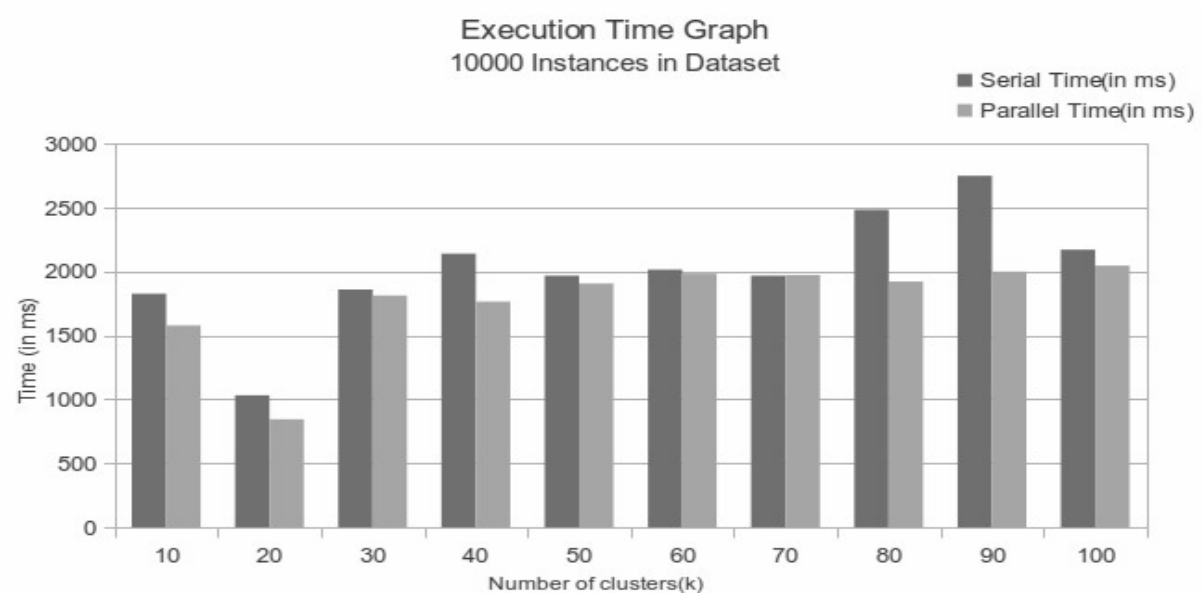

Fig.3. Execution graph for 10000 instances

The above is a graph for values in Table 2. The execution time values are provided for k=10 over dataset instances varying in range 1000 to 10000.

Conclusions

The k-modes algorithm inherits parallelism in distance computation. Also, categorical data is processed by k-modes. Thus, selection of k-modes for clustering categorical data on multi-core platform is done. The execution time results listed in above tables prove that speedup is achieved in k-modes algorithm with the use of OpenMP framework. Further, it can be tuned to work with CUDA framework and MapReduce framework. The OpenMP framework can be used in future to allow parallelism in other data mining algorithms.

REFERENCES

[1] Congressional Database, UCI repository.

[2] Kavitha Karun A, Elizabeth Isaac, *"Cogitative Analysis on K-Means Clustering Algorithm and its Variants"*, International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 4, April 2013

[3] Shraddha Masih, Sanjay Tanwani, *"Data Mining Techniques in Parallel and Distributed Environment- A comprehensive Survey"*, International Journal of Emerging Technology and Advanced Engineering(ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 4, Issue 3, March 2014)

[4] Joshua Zhexue Huang, *"Clustering Categorical Data with k-Modes"*, The University of Hong Kong, Hong Kong

[5] Rishi Syal, Dr V. Vijaya Kumar, *"Innovative Modified K-Mode Clustering Algorithm"*, International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 Vol. 2, Issue 4, July-August 2012, pp.390-398.

[6] Poonam M. Bhagat, Prasad S. Halgaonkar, Vijay M. Wadhai, *"Review of Clustering Algorithm for Categorical Data"*, International Journal of Engineering and Advanced Technology (IJEAT).

[7] RekhanshRao, Kapil Kumar Nagwanshi, and SipiDube, "*Multicore Processing for Clustering Algorithms"*, International Journal of Computer Technology & Applications,Vol 3 (2), 555-560

[8] Zhexue Huang, *"Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values"*, Data Mining and Knowledge Discovery 2, 283–304 (1998)

[9] Tirumala Rao , E.V. Prasad' and N.B.Venkateswarlu, *"A Scalable k-means Clustering Algorithm on Multi-Core Architecture"*, International Conference on Methods and Models in Computer Science, 2009.

[10] S. Anitha Elavarasi and J. Akilandeswari, *"Survey on clustering algorithm and similarity measure for categorical data"*, Ictact journal on soft computing, January 2014, volume: 04, issue: 02

[11] Dinesh. B. Kulkarni, Vivek N. Waghma, *"Convex Hull Using K-Means Clustering in Hybrid(MPI/Open MP) Environment"*, International Conference on Computational Intelligence and Communication, 2010.

[12] Li X. and Fang Z., *"Parallel clustering algorithms, Parallel Computing"*, 1989, 11(3): pp275-290

[13] Kantabutra S. and Couch A., *"Parallel k-means clustering algorithm on NOWs"*, NECTEC Technical Journal, 2000, 1(6): pp. 243-248.

[14] Stoffel Kilian and Belkoniene Abdelkader, *"Parallel k/h-Means Clustering for Large Data Sets"*, Euro-Par'99, LNCS 1685, 1999, pp. 1451-1454.

[15] Periklis Andristos, *"Data Clustering Techniques"*, March 2007.

[16] Joshua Zhexue Huang, *"Clustering Categorical Data with k-Modes"*, IGI Global, 2009.

[17] Petr Behlohlavek, "OpenMP for Java", Bachelor Thesis, Charles University in Prague, May 2015.