An OpenMP Based Approach for Parallelization and Performance Evaluation of k-Means Algorithm

Ansari Abdullah[a], Quazi Mateenuddin H [b] and Zahid Ansari [c]

[a] Department of Computer Science and Engineering, Bearys Institute of Technology, Mangalore
[b] Faculty of Electronics and Communication Engineering, Indian Naval Acadamy, Ezhimala
[c] Department of Computer Science, P A College of Engineering, Mangalore
Email:[a]ansaridx99@gmail.com, [b]qmateen@rediffmail.com, [c]zahid_cs@pace.edu.in

Article History Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

Abstract: In today's digital world, the volume of data is drastically increasing due to the continuous flow of data from various heterogenous sources such as WWW, social media, environmental sensors, huge enterprise data warehouses, bioinformatic labs etc. to name a few. This results in creation of many high-volume datasets in various domains. Processing such large datasets is a tedious task, therefore they need to be categorized into smaller subsets using various supervised or unsupervised classification techniques. Clustering is the process of statistically analyzing and categorizing data objects with similarity, into substantially homogeneous groups, called data clusters. k-Means is the most common, simple and popular clustering technique, due to its ease of implementation, usability and wide range of applications. One of the issues associated with the k-Means algorithm is that it suffers from the scalability problem due to which, its performance degrades as the dataset sizes grow. In order to address this issue, we have presented an OpenMP based parallelized k-means algorithm which results in better computational cost as compared with its sequential counterpart. Computational performance results of both sequential and OpenMP based k-means algorithms are illustrated and compared.

Keywords: k-Means, OpenMP, Parallel Clustering

1. Introduction

Clustering is one of the common data mining operations that has many applications for data processing and categorization [1-3]. k-Means algorithm performs partitioning of the dataset objects into various clusters each of them represented by their centroids. [4-6]. In today's age of digitization there is a continuous flow of data from various heterogenous sources such as social media, WWW, environmental sensors, enterprise data warehouses, bioinformatic labs etc. This results in creation of a large number of high-volume datasets in various domains. Processing such large datasets is a tedious task, therefore they need to be categorized into smaller subsets using various supervised or unsupervised classification techniques [7-8].

When k-Means algorithm is applied to these massive datasets of sizes in gigabytes or terabytes, it suffers from the scalability problem due to which, its performance degrades as the dataset sizes grow. Many times, the traditional k-Means algorithms fail to execute in-core such high voluminous data or would result in extremely high computational time [9-10]. In order to speed up the k-Means execution on large datasets the parallel or distributed variant of k-Means must be used for processing voluminous datasets. Since now days most of the computational hardware are equipped with multiple cores, the performance of k-Means can be greatly improved by utilizing these cores and their associated memory units. [11-13]

In this study we have presented an OpenMP based parallelized k-means algorithm to improve the computational cost as compared with its sequential counterpart. One of the necessary requirements of this algorithms is that, the clustering result produced by it should match with that of its sequential counterpart.

After providing the introductions in section I, the remaining paper is organized as follows. In Section II, a review of the literature related to traditional sequential k-Means and its parallel OpenMP based counterpart are provided. In section III details of the proposed methodology are provided. Section IV describes the comparison of results of traditional and the proposed OpenMP based k-means. Finally, conclusions are drawn in section V.

2.Related Works

An extensive amount of work related to k-Means and various other clustering techniques has been reported in literature. In this section, a review of some of the selected work is presented. Clustering algorithms have been applied in wide range of domains including web mining, bioinformatics, image analysis, telecommunication, software modelling, business intelligence to name a few [14-28]. In order to prepare the massive datasets for clustering to be applied, it needs to be preprocessed. Several data preprocessing work have been reported, some of which can be found in [29-33].

Ansari et. al. has worked on various clustering techniques in the field of web usage clustering [34-35]. They have provided the comparative results of these techniques and performed the quantitative evaluation of their performance based on various performance measuring indices [36-37]. They have also utilized partition-based clustering algorithms for the clustering of web navigational access data [38] using k-Means, Fast global k-Means and k-Medoids methods [39-41]. They have also provided the comparison between these algorithms for cluster formation. When k-Means algorithm is integrated with soft computing techniques it become more robust against data imperfections, but it becomes computationally expensive. Fuzzy set-based k-Means algorithms have been extensively applied for data categorization [42-45] where each object may be associated with multiple categories with a different level of membership. Neural Network based k-Means algorithms add better more and robustness to k-Means but at the cost of high computational time [46-49]. Rough-set based k-Means algorithm also provides overlapping clusters but runs too slow [50]. Other soft computing techniques such as modified mountain clustering have also been used for data categorization [51-52].

To deal with the high voluminous data, several distributed data clustering approaches using Hadoop and MapReduce have been successfully applied. Tanvir et. al. has reported several works related to MapReduce based variant of k-Means algorithms for document clustering [53-54]. Many other improved variants of k-Means with the objective of enhancing their computational performance can be found in [55-56]. There are related works on OpenMP based parallel k-Means. Huang et al. illustrated performance of k-Means on multi-cores [57]. Nazir et. al have performed parallel partitioning using OpenMP to optimize the computational cost of k-Means [58].

In this OpenMP based implementation of k-means algorithm, those snippets of code are parallelized which most expensive computationally such as distance calculation, choosing the cluster etc. This selective parallelization gives good performance and doesn't add much overhead. And for solving the problems of false sharing, OpenMP's 'schedule' clause is used to schedule the iteration between the threads.

3.Methodology

Let us first review the sequential implementation of k-means algorithm for a better understanding the methodology on OpenMP based k-Means.

Sequential k-Means: Sequential k-Means clustering algorithm is described in Algorithm 1. The initial centroid C_i can be found for the range values n_1 and n_2 with k clusters as:

$$C_i = ((n_2 - n_1) / k) * (i+1) \quad \text{for } i < k \quad (1)$$

The Euclidean distance in two dimensions between two points $p = (p_1, p_2)$ and $q = (q_1, q_2)$ is given by:

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}. \quad (2)$$

The new cluster centroid C(i) can be found by,

for i=0 to k-1

$$\text{sum}_i = \sum_{j=0}^{count(i)} d(j,i)$$

$$C_i = \text{sum}_i / \text{count}_i \quad (3)$$

The advantage of this algorithm is that here we take the initial cluster centroid with the help of the range of the data items. Hence the performance and the cluster quality will be increased

Algorithm 1: Sequential K-means

```
Input: $D = \{d_1, d_2, \ldots, d_n\}$, set of $n$ data items, $k$ number of desired clusters.
Output: $k$ clusters.
Steps:
1. Initialize the k centroids using (1).
2. Perform distance calculation between cluster centers and data objects using (2).
3. Associate each object $d$i to the nearest cluster with minimum distance.
4. Repeat
    a. Calculate new cluster centroid using (3).
    b. Perform distance calculation between cluster centers and data objects using (2).
    c. Associate each object $d$i to the nearest cluster
```

Algorithm 2: K-Means Using OpenMP

```
Input: $D = \{d_1, d_2, \ldots, d_n\}$, set of $n$ data items, $k$ number of desired clusters
Output: $k$ clusters.
Steps:
1 Master thread initializes the k centroids using (1).
2 Childs threads calculate the distance between each data items and each cluster using (2) in parallel.
3 Child threads associates each $d$i to the closest cluster with minimum distance between them in parallel.
4 Repeat
    a. Master thread calculates new cluster centroid using (3).
    b. Child threads perform distance calculation
```

OpenMP Based Implementation of k-Means: Parallel K-means clustering algorithm using OpenMP is described in Algorithm 2. It enables the cluster analysis in shared memory system for very large datasets. In this implementation we use the number of threads equal to hardware threads because that gives the better efficiency and the problems with false sharing is also avoided with the help of the schedule clause of the for directive of OpenMP.

4.Experimental Results

Artificially generated synthetic datasets are used for the experimentation purpose. Data objects are randomly generated in each synthetic dataset. To observe the influence of the number of dataset size on the computational performance, datasets with 1000, 10000, 20000, 30000 and 50000 2-dimensional were created for different values of *k* ranging from 2 to12. Multiple runs providing execution time of serial and OpenMP k-means clustering were set, based on the two ways:

1. Varying data size, keeping k (number of cluster) constant.
2. Varying k (number of cluster), keeping data size constant.

***Varying data size keeping k constant*:** Observing the change in execution time keeping k the number of clusters constant from k=2,4,6,8,10,12 and varying dataset from 1000, 10000, 20000, 30000 and 50000. Table 1,3,5,7,9,11 shows the execution time of Serial vs. OpenMP code where k=2, 4, 6, 8, 10, 12. Table 2, 4, 6, 8, 10, 12 shows the Speedup for Serial vs. OpenMP code where k=2, 4, 6, 8, 10. 12. Fig. 1-6 illustrate the graph of computational time of Sequential vs. OpenMP implementation where k=2, 4, 6, 8, 10, 12.

Table I Execution Time (ms) of Serial vs OpenMP when k=2

Dataset	Serial	OpenMP
1000	7	10
10000	50	40
20000	60	60
30000	130	100
50000	200	150

Table II Execution Time (Ms) Of Serial Vs Openmp When K=3

Dataset	Speedup (OpenMP)
1000	0.7
10000	1.25

20000	1
30000	1.3
50000	1.333

Table III Execution Time (ms) of Serial vs OpenMP when k=4

Dataset	Serial	OpenMP
1000	7	10
10000	50	40
20000	110	80
30000	150	110
50000	290	190

Table IV Speedup for Serial vs. OpenMP code for k=4

Dataset	Speedup (OpenMP)	
1000	0.7	
10000	1.25	
20000	1.375	
30000	1.364	
50000	1.526	

Table V Execution Time (ms) of Serial vs OpenMP for k=6

Dataset	Serial	OpenMP
1000	10	20
10000	100	70
20000	410	250
30000	780	460
50000	1170	690

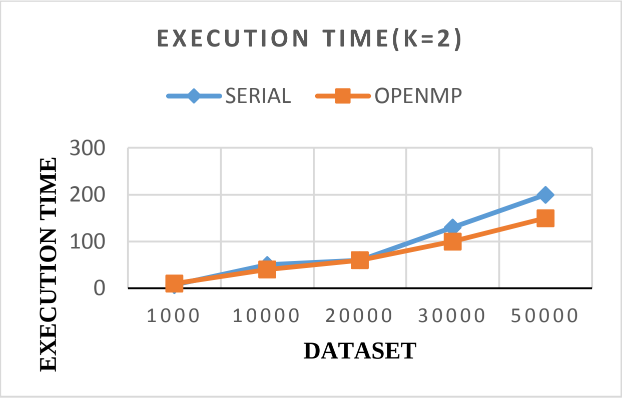

Fig. 1 Execution Time (ms) of Serial vs OpenMP for k=2

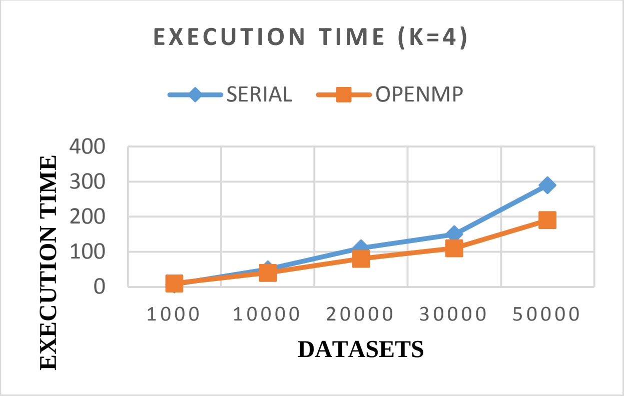

Fig. 2 Execution Time (ms) of Serial vs OpenMP for k=4

Table VI Speedup for Serial vs OpenMP code for k=6

Dataset	Speedup (OpenMP)
1000	0.50
10000	1.429
20000	1.640
30000	1.696
50000	1.696

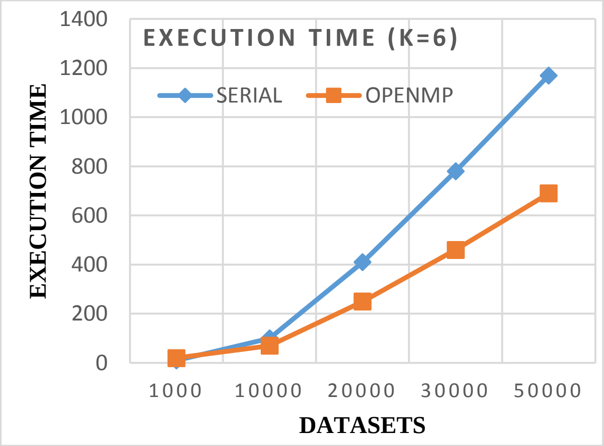

Fig. 3 Execution Time (ms) of Serial vs OpenMP for k=6

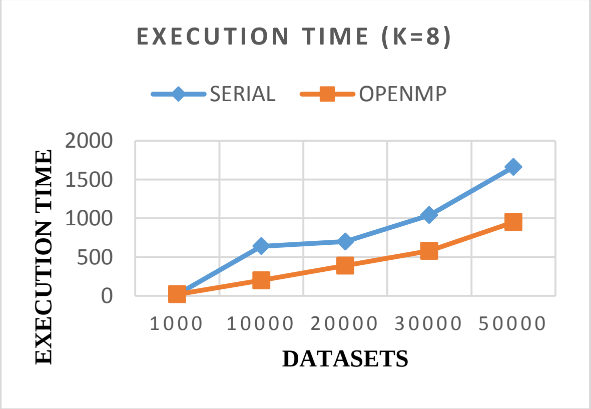

Fig. 4 Execution Time (ms) of Serial vs OpenMP for k=8

Table VII Execution Time (ms) of Serial vs OpenMP for k=8

Dataset	**Serial**	**OpenMP**
1000	20	20
10000	640	200
20000	700	390
30000	1040	580
50000	1660	950

Table VIII Speedup for Serial vs OpenMP for k=8

Dataset	**Speedup (OpenMP)**
1000	1
10000	3.2
20000	1.795
30000	1.793
50000	1.747

Table IX Execution Time (ms) of Serial vs OpenMP for k=10

Dataset	**Serial**	**OpenMP**
1000	10	10
10000	210	130
20000	350	210
30000	730	420
50000	1890	1070

Table X Speedup for Serial vs OpenMP for k=10

Dataset	**Speedup (OpenMP)**
1000	1.00
10000	1.615
20000	1.667
30000	1.738
50000	3.405

Table XI Execution Time (ms) of Serial vs OpenMP for k=12

Dataset	**Serial**	**OpenMP**
1000	10	10
10000	220	130
20000	950	530
30000	990	570
50000	1720	950

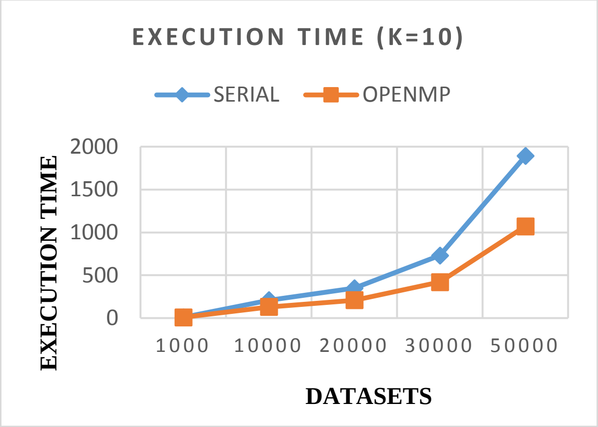

Fig. 5 Execution Time (ms) of Serial vs OpenMP for k=10

Table XII Speedup for Serial vs OpenMP for k=12

Dataset	Speedup (OpenMP)
1000	1
10000	1.692
20000	1.792
30000	1.737
50000	1.811

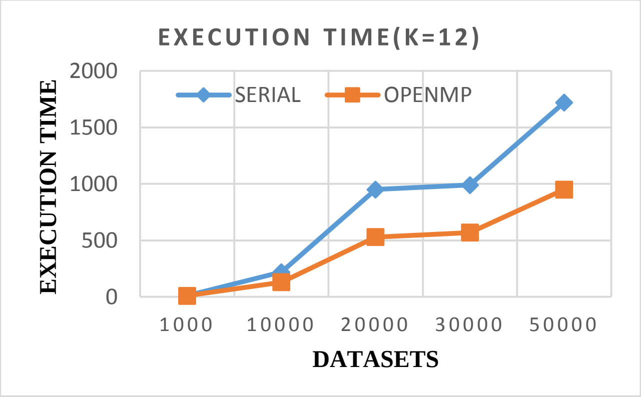

Fig. 6 Execution Time (ms) of Serial vs OpenMP for k=12

From Fig. 1-6 we can see that when k=2, 4, 6, 8, 10, 12 and the data size between 1000-50000 there are variations in execution time. For data size 1000, serial k-means has better execution time but OpenMP based k-Means provides better performance for all data sizes > 1000. This indicates that OpenMP based k-Means results in better execution time.

Varying k keeping data size constant: Observing the change in execution time keeping dataset constant from 1000, 10000, 20000, 30000 and 50000 and varying the k from k = 2 to12. Tables 13, 15, 17, 19, 21 shows the execution time of Serial vs. OpenMP code where dataset= 1000, 10000, 20000, 30000 and 50000. Table 14, 16, 18, 20, 22 shows the Speedup for Serial vs. OpenMP code where dataset= 1000, 10000, 20000, 30000 and 50000. Fig. 7-11 illustrate graph of execution time of Sequential vs. OpenMP for data size = 1000, 10000, 20000, 30000 and 50000.

Table XIII Execution Time (ms) of Serial vs. OpenMP when dataset=1000

k	Serial	OpenMP
2	7	10
4	7	10
6	10	20
8	20	20
10	10	10
12	10	10

Table XIV Speedup for CPU vs. OpenMP for dataset=1000

k	Speedup (OpenMP)
2	0.7
4	0.7
6	0.5
8	1
10	1
12	1

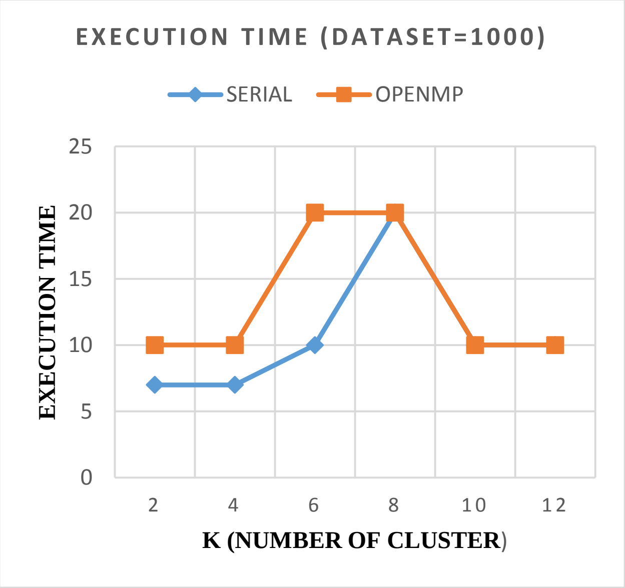

Fig. 7 Execution Time (ms) of Serial vs. OpenMP for dataset=1000

Table XV Execution Time (ms) of Serial vs. OpenMP for dataset=10000

k	Serial	OpenMP
2	50	40
4	50	40
6	100	70
8	640	200
10	210	130
12	220	130

Table XVI Speedup for CPU vs. OpenMP for dataset=10000

k	**Speedup (OpenMP)**
2	1.25
4	1.25
6	1.42
8	3.2
10	1.62
12	1.69

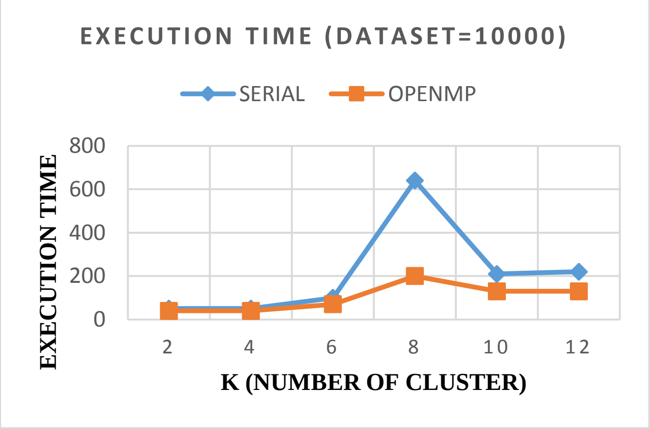

Fig. 8 Execution Time (ms) of Serial vs. OpenMP for dataset=10000

Table XVII Execution Time (ms) of Serial vs. OpenMP for dataset=20000

k	**Serial**	**OpenMP**
2	60	60
4	110	80
6	410	250
8	700	390
10	350	210
12	950	530

Table XVIII Speedup for CPU vs. OpenMP for dataset=20000

k	**Speedup (OpenMP)**
2	1
4	1.38
6	1.64
8	1.79
10	1.67
12	1.79

Table XIX Execution Time (ms) of Serial vs. OpenMP for dataset=30000

k	Serial	OpenMP
2	130	100
4	150	110
6	780	460
8	1040	580
10	730	420
12	990	570

Table XX Speedup for CPU vs. OpenMP for dataset=30000

k	Speedup (OpenMP)
2	1.30
4	1.36
6	1.70
8	1.79
10	1.74
12	1.74

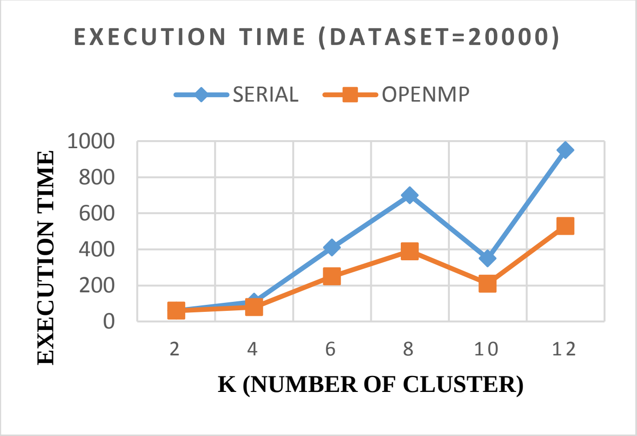

Fig. 9 Execution Time (ms) of Serial vs. OpenMP for dataset=20000

Table XXI Execution Time (ms) of Serial vs. OpenMP for dataset=50000

k	Serial	OpenMP
2	200	150
4	290	190
6	1170	690
8	1660	950
10	1890	1070
12	1720	950

Table XXI Speedup for CPU vs. OpenMP for dataset=50000

k	Speedup (OpenMP)
2	1.33
4	1.53
6	1.70
8	1.75
10	1.77
12	1.81

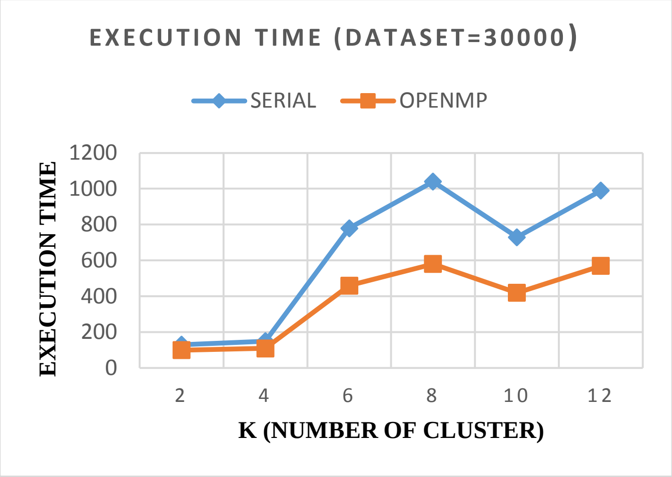

Figure 10: Execution Time (ms) of Serial vs. OpenMP for dataset=30000

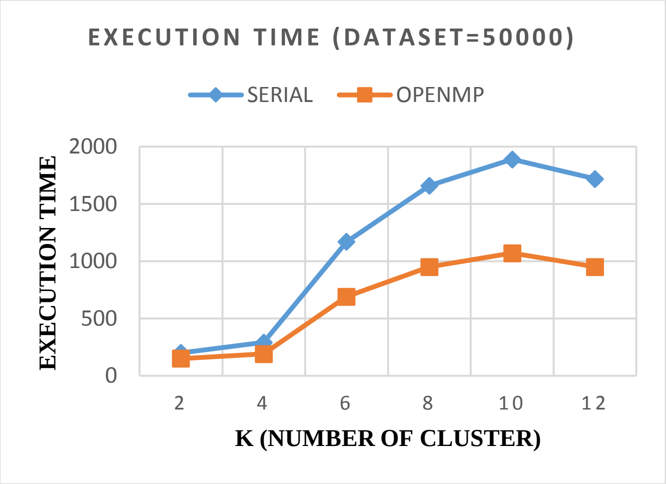

Figure 11: Execution Time (ms) of Serial vs. OpenMP for dataset=50000

From Fig. 7-11, we see that for data size 1000, the performance of OpenMP and serial k-means are comparable. But when dataset kept constant between 1000 to 50000 varying k from 2 to 12, we obtain better execution time in OpenMP k-means clustering code compared to serial k-means clustering code. This shows that OpenMP k-means results in better execution time.

5.Conclusion

In this study, OpenMP based parallelization of k-Means algorithm is attempted with objective reducing the computational cost of k-Means on large datasets without sacrificing the accuracy. From the experimental results, it has been observed that the proposed OpenMP based parallel version of the k-Means produces exactly, the same results as with the Serial algorithm with much lower computational almost inversely proportional to the number of cores used.

Although the experimental results presented in this study is based on artificially generated synthetic data, OpenMP based parallel version of k-Means can very well be applied on real world huge datasets such as web access logs, bioinformatics sequences, high dimensional images etc.

6.Acknowledgement

This study is funded by GoK-VGST-CISEE scheme (GRD-No-461). We are also thankful to Saikiran Hegde, Yusuf Ansar, Shaswath and Prajan Shetty for their support in this work.

References

[1] Zahid Ansari, Asif Afzal, Tanvir Habib Sardar, "Data Categorization Using Hadoop MapReduce-Based Parallel K-Means Clustering", Journal of the Institutions of Engineers, Springer Link. ISSN: 2250-2106 (Print) 2250-2114 (Online), vol. 100, no. 2, pp. 95-103, April 2019. DOI:10.1007/s40031-019-00388-x

[2] Ahamed Shafeeq and Zahid Ansari, "Empirical Analysis of K-Means, Fuzzy C-Means and Particle Swarm Optimization for Data Clustering", Journal of Advanced Research and Dynamical Control Systems (JARDCS) ISSN:1943-023X, Vol. 11, Special Issue 03, pp. 1743-1748, 2019.

[3] Zahid Ansari, Quazi Mateenuddin H. and Ansari Abdullah, "Performance Analysis of Medical Data Classification Using Traditional and Soft Computing Techniques", International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, vol. 8, no. 2, pp. 990-995, July 2019. DOI: 10.35940/ijrte. B1185.0782S319

[4] Mohammed Zakir Bellary, Aziz Mustafa and Zahid Ansari, "Segmentation of Pathological MR Images for Discovery of Useful Patterns Using Various Clustering Techniques", International Journal of Emerging Technologies and Innovative Research, ISSN-2349-5162, vol. 6, no. 5, pp. 451-459, 2019

[5] Zahid Ansari, Syed Abdul Sattar and A.Vinaya Babu, "A Fuzzy Neural Network Based Framework to Discover User Access Patterns from Web Log Data", Advances in Data Analysis and Classification (ADAC), Springer Berlin Heidelberg, ISSN: 1862-5347, vol. 11, no. 3, pp. 519-546, September 2017. doi: 10.1007/s11634-015-0228-4

[6] Zahid Ansari, Ahmed Rimaz Faizabadi, "Fuzzy c-Least Medians Clustering for Discovery of Web Access Patterns from Web User Sessions Data", Intelligent Data Analysis, An International Journal, IOPress, ISSN: 1088-467X, vol. 21, no. 3, pp. 553-575, May 2017. doi: 10.3233/IDA-150489

[7] Tanvir Habib Sardar, Zahid Ansari, "An Analysis of Distributed Document Clustering using MapReduce based K-Means Algorithm", Journal of the Institutions of Engineers, Springer Link. (Scopus Indexed Journal) ISSN: 2250-2106 (Print) 2250-2114 (Online), vol. xxx, no. x, pp. xx-xxx, xxx 2020. DOI:10.1007/s40031-019-00388-x (In Press)

[8] Zahid Ansari, M.F. Azeem, A. Vinaya Babu and Waseem Ahmed. "A Fuzzy Approach for Feature Evaluation and Dimensionality Reduction to Improve the Quality of Web Usage Mining Results". International Journal on Advanced Science Engineering and Information Technology (IJASEIT), ISSN: 2088-5334, vol. 2 no. 6, pp. 67-73. 2012

[9] Zahid Ahmed Ansari, "Web User Session Cluster Discovery Based on k-Means and k-Medoids Techniques", International Journal of Computer Science & Engineering Technology (IJCSET).

[10] Tanvir Habib Sardar, Zahid Ansari, "Partition based clustering of large datasets using MapReduce framework: An analysis of recent themes and directions", Future Computing and Informatics Journal, Elsevier. ISSN 2314-7288 vol. 3, no. 2, pp. 247-261, December 2018. DOI: 10.1016/j.fcij.2018.06.002

[11] Asif Afzal, Zahid Ansari, Ahmed Rimaz Faizabadi, Ramis M.K, "Parallelization Strategies for Computational Fluid Dynamics Software: State of the Art Review", Archives of Computational Methods in Engineering (ARCO), Springer Netherlands, ISSN: 1134-3060, vol. 24, no. 2, pp. 337-363, April 2017, doi:10.1007/s11831-016-9165-4

[12] Tanvir Habib Sardar, Ahmed Rimaz Faizabadi, Zahid Ansari, "An Analysis of Data Processing using MapReduce Paradigm on the Hadoop Framework", Special Issue in International Journal of Emerging Research

in Management and Technology, ISSN: 2278-9359, vol. 6 no. 5, pp. 922-927, May 2017.

[13] Zahid Ansari, Asif Afzal, Moomin Muhiuddeen, Sudarshan Nayak, "Literature Survey for the Comparative Study of Various High-Performance Computing Techniques", International Journal of Computer Trends& Technology (IJCTT), ISSN: 2231-2803. vol.27, no. 2, pp. 74-80, September 201.

[14] Zahid Ansari and Syed, A.S. "Discovery of web usage patterns using fuzzy mountain clustering", Int. J. Business Intelligence and Data Mining, Inderscience Publications, ISSN: 1743-8195, vol. 11, no. 1, pp.1-18, May 2016.

[15] Amjad Khan and Zahid Ansari, "Soft Computing based Medical Image Mining: A Survey", International Journal of Computer Trends & Technology (IJCTT), ISSN: 2231-2803, vol. 27, no. 2, pp. 76-79, September 2015.

[16] Sameema and Zahid Ansari, "Performing Automatic Clustering Using Active Clusters Approach with Particle Swarm Optimization", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), ISSN: 2277-128X, vol. 5 no. 3, pp.198-204. March 2015

[17] Jovita Vanisequeira and Zahid Ansari, (March 2015) "Analysis on Improved Pruning in Apriori Algorithm", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), Vol. 5 Issue 3, pp. 892-902, (ISSN: 2277-128X).

[18] Mohammed Tajuddin, Zahid Ansari, Syed Ab. Sattar, "A Study for the Discovery of Web Usage Patterns Using Soft Computing Based Data Clustering Techniques", International Journal of Data Mining Techniques and Applications (IJDMTA), ISSN: 2278-2419, vol. 4, no. 2, pp. 418-424. December 2014.

[19] Kadijath Tahira, Jovita Vanisequeira, Sameema and Zahid Ansari, "Performance Evaluation of Neural Network Based Pattern Classification", International Journal of Emerging Technologies and Applications in Engineering, Technology and Sciences (IJ-ETA-ETS), ISSN: 0974-3588, Special Issue, pp. 306-312, Dec. 2014.

[20] Shajeeah. M, Safana. N, Fathimath Rajeela. K.A., Kadeejath Sajida, Zahid Ansari, "A Framework to Discover Association Rules Using Frequent Pattern Mining", International Journal of Data Mining Techniques and Applications (IJDMTA), ISSN: 2278-2419, vol 4, no. 2, pp. 425-430, December 2014

[21] Waseem Ahmed, Zahid Ansari, Johannes Herrmann, and Matin Abdullah. "A Looking-Out Portal (lop) Approach to Enhance Qualitative Aspects of Bandwidth Utilization in Academic networks", International Journal of Networking and Virtual Organizations (IJNVO), ISSN:1741-5225, Vol. 9, No. 4, pp. 317-330, 2011, doi:10.1504/IJNVO.2011.043802.

[22] Kevin. J. Dsouza and Zahid. Ansari, "Experimental Exploration of Support Vector Machine for Cancer Cell Classification", IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bangalore, India, Nov. 1-3, 2017, pp. 29-34 doi: 10.1109/CCEM.2017.15

[23] Kevin. J. Dsouza and Zahid. Ansari, "A Novel Data Mining Approach for Multi Variant Text Classification", IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bangalore, India, Nov. 25-27, 2015, pp. 68-73 doi: 10.1109/CCEM.2015.11

[24] Mohammed Tajuddin, Syed Hussain, Zahid Ansari, "Efficient Document Clustering: A Survey", in RIST Second International Conference on Advancements in Engineering and Management, (ICAEM-2013), Hyderabad, India. Feb.27-28. 2013

[25] S. Reddy, R. Venkatesh, Zahid Ansari, "A Relational Approach to Model Transformation Using QVT Relations", at TECS, Tata Research Development and Design Centre, Pune, 2006, India. http://www.iist.unu.edu/ vs/wiki-files/QVT-TRDCC.pdf

[26] Zahid Ansari, Tanvir Sardar, Moksud Mallik and Naveen Chandavarkar, "Data Mining in Soft Computing Framework: A Survey", Proceedings of the National Conference on Multimedia and Information Security (NCMIS-14), Mangalore, India. February 20-21. 2014

[27] Kevin. J. Dsouza and Zahid. Ansari, "Big Data Science in building medical Data Classifier using Naïve Bayes Model", IEEE 7th International Conference on Cloud Computing in Emerging Markets (CCEM), Bangalore, India, Nov. 23-24, 2018.

[28] Amjad Khan and Zahid Ansari, "Comparative Study of Data Mining in Telecommunications - A Survey", International Journal of Emerging Technologies and Applications in Engineering, Technology and Sciences

(IJ-ETA-ETS), ISSN:0974-3588., Special Issue, vol 7, no. 1, pp. 269-276, Jan - June 2014.

[29] Kadijath Tahira and Zahid Ansari, "Advanced Data Preprocessing and Soft Computing Based Web Usage Pattern Discovery", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), ISSN: 2277-128X, vol. 5, no. 3, pp. 785-795, March 2015.

[30] Ishrath Rayeesa, Kadijath Thahira and Zahid Ansari, "Preprocessing Methodologies for the Discovery of Web Access Patterns from the Raw Web Log Data", International Journal of Emerging Technologies and Applications in Engineering, Technology and Sciences (IJ-ETA-ETS), ISSN:0974-3588, Special Issue, vol 7, no. 1, pp. 269-276, Jan - June 2014.

[31] Tanvir Sardar and Zahid Ansari, "Detection and Confirmation of Web Robot Requests for Cleaning the Voluminous Web Log Data", Proceedings of the IEEE International Conference on Impact of E-Technology on US (IC-IMPETUS), Bangalore, India. January 10-11. 2014 pp. 13-19

[32] Tanvir Sardar and Zahid Ansari, "A Methodology for Detecting Web Robot Requests from Voluminous Web Log File", Proceedings of the International Conference on Emerging Trends in Engineering (ICETE-2014), Mangalore, India. May 15-17. 2014.

[33] Zahid Ansari, M. F. Azeem, A. V. Babu and W. Ahmed. "Preprocessing User's Web Navigational Data to Discover Usage Patterns", in Proceedings of The Seventh International Conference on Computing and Information Technology, Bangkok, Thailand, pp. 184-189. May 2011.

[34] Zahid Ansari, Waseem Ahmed, M.F. Azeem and A. Vinaya Babu. "Discovery of Web Usage Profiles Using Various Clustering Techniques", International Journal of Computer Information Systems, ISSN: 2229 5208, vol. 1, no. 3, pp. 18-27, July 2011.

[35] Zahid Ansari, A. Vinaya Babu, Waseem Ahmed and M. F. Azeem, "A Comparative Study of Mining Web Usage Patterns Using Variants of k-Means Clustering Algorithm", International Journal of Computer Science and Information Technologies, ISSN: 0975-9646, vol. 2 no. 4, pp. 1407-1413. July 2011.

[36] Zahid Ansari, "Discovery of Web User Session Clusters Using DBSCAN and Leader Clustering Techniques", International Journal of Research in Applied Science & Engineering Technology (iJRASET), ISSN: 2321-9653 vol 2, no. 12, pp. 209-207. December 2014.

[37] Zahid Ansari, A. Vinaya Babu, M.F. Azeem and Waseem Ahmed. "Quantitative Evaluation of Performance and Validity Indices for Clustering the Web Navigational Sessions", World of Computer Science and Information Technology Journal (WCSIT), ISSN: 2221-0741, vol. 1, no. 5, pp. 217-226, June 2011.

[38] Zahid Ansari, Mohammed Tajuddin, Syed Ab. Sattar, "Discovery of Web User Session Clusters Using Partitioning Based Clustering Techniques", International Journal of Computer Technology and Applications (IJCTA), ISSN: 2229-6093, vol 5, no. 6, pp. 2049-2056, Nov-Dec 2014.

[39] Zahid Ansari, M. Tajuddin and S. A. Sattar, "Web Usage Mining Using k-Means and k-Medoids Clustering Techniques", In Proceedings of the International Conference on Communications, Signal Processing, Computing and Information Technologies (ICCSPCIT 2014), ISBN 978-93-83038-27-5, Hyderabad, India. Dec 26-27, 2014, pp. 234-239.

[40] Zahid Ansari, "Web User Session Cluster Discovery Based on k-Means and k-Medoids Techniques", International Journal of Computer Science & Engineering Technology (IJCSET), ISSN:2229-3345, vol 5, no. 12, pp. 1105-1113, December 2014.

[41] Zahid Ansari and Amjad Khan, "Fast Global k-Means Method to Discover User Session Clusters from Web Log Data", International Journal of Computer Engineering and Applications (IJCEA), ISSN:2321-3469, vol. 8 no. 3, pp. 26-35. Dec.2014

[42] Zahid. Ansari and Mateenuddin Quazi, "A Fuzzy Approach for Discovery of Web Usage Patterns from Web Log Data", In 2017 International Conference on Science, Technology, Engineering and Management (ICSTEM 2017), New York, USA, pp. 60-66, Dec. 15-16, 2017,

[43] Zahid Ansari, A. Vinaya Babu, Waseem Ahmed and Mohammad Fazle Azeem, "A Fuzzy Set Theoretic Approach to Discover User Sessions from Web Navigational Data", in International Conference on IEEE Recent Advances in Intelligent Computational Systems, Trivandrum Sep. 22-24, 2011.