2017

Cost-effective Big Data Mining in the Cloud: A Case Study with K-means

Qiang He
Swinburne University of Technology, qhe@swin.edu.au

Xiaodong Zhu
University of Shanghai for Science and Technology

Dongwei Li
University of Shanghai for Science and Technology

Shuliang Wang
University of Shanghai for Science and Technology

Jun Shen
University of Wollongong, jshen@uow.edu.au

See next page for additional authors

Publication Details

He, Q., Zhu, X., Li, D., Wang, S., Shen, J. & Yang, Y. (2017). Cost-effective Big Data Mining in the Cloud: A Case Study with K-means. IEEE 10th International Conference on Cloud Computing 2017 (pp. 74-81). United States: IEEE.

Cost-effective Big Data Mining in the Cloud: A Case Study with K-means

Abstract
Mining big data often requires tremendous computationalresources. This has become a major obstacle to broad applicationsof big data analytics. Cloud computing allows data scientists to access computationalresources on-demand for building their big data analytics solutions in the cloud.

Keywords
k-means, study, case, big, cloud:, cost-effective, mining, data

Publication Details
He, Q., Zhu, X., Li, D., Wang, S., Shen, J. & Yang, Y. (2017). Cost-effective Big Data Mining in the Cloud: A Case Study with K-means. IEEE 10th International Conference on Cloud Computing 2017 (pp. 74-81). United States: IEEE.

Authors
Qiang He, Xiaodong Zhu, Dongwei Li, Shuliang Wang, Jun Shen, and Yun Yang

Cost-effective Big Data Mining in the Cloud: A Case Study with K-means

Qiang He[1], Xiaodong Zhu[2], Dongwei Li[3,1], Shuliang Wang[3], Jun Shen[4], Yun Yang[1,5]
[1]Swinburne University of Technology, Melbourne, Australia
[2]University of Shanghai for Science & Technology, Shanghai, China
[3]Beijing Institute of Technology, Beijing, China
[4]University of Wollongong, Wollongong, Australia
[5]Anhui University, Hefei, China
qhe@swin.edu.au, zhuxd@usst.edu.cn, {dli, slwang}@bit.edu.cn, jshen@uow.edu.au, yyang@swin.edu.au

***Abstract*—Mining big data often requires tremendous computational resources. This has become a major obstacle to broad applications of big data analytics. Cloud computing allows data scientists to access computational resources on-demand for building their big data analytics solutions in the cloud. However, the monetary cost of mining big data in the cloud can still be unexpectedly high. For example, running 100 m4-xlarge Amazon EC2 instances for a month costs approximately $17,495.00. On this ground, it is a critical issue to analyze the cost effectiveness of big data mining in the cloud, i.e., how to achieve a sufficiently satisfactory result at the lowest possible computation cost. In certain big data mining scenarios, 100% accuracy is unnecessary. Instead, it is often more preferable to achieve a sufficient accuracy, e.g., 99%, at a much lower cost, e.g., 10%, than the cost of achieving the 100% accuracy. In this paper, we explore and demonstrate the cost effectiveness of big data mining with a case study using well known k-means. With the case study, we find that achieving 99% accuracy only needs 0.32%-46.17% computation cost of 100% accuracy. This finding lays the cornerstone for cost-effective big data mining in a variety of domains.**

Keywords-Cloud Computing; Data Mining; Cost-Effective; Big Data; K-Means

I. INTRODUCTION

The era of big data has arrived [1]. Ninety percent of the data in the world today were produced within the past two years and 2.5 quintillion bytes of new data are created every day [2]. For instance, about 6 billion new photos are reported every month by Facebook and 72 hours of video are uploaded to YouTube every minute [2]. This explosive growth of data has fueled big data mining in a wide range of sections, e.g., business [3], government [4], healthcare [5], etc.

Most data mining algorithms are exponential in computational complexity. In big data scenarios, it is not rare for the data mining process to take hours, even days, to complete. Thus, big data mining often requires tremendous computational resources. Many businesses and organizations cannot afford the costs of in-house IT infrastructure for big data mining, especially, small and medium sized businesses. Cloud computing is the perfect solution for them [6]. The "pay-as-you-go" model promoted by cloud computing enables flexible and on-demand access to virtually unlimited computational resources. This allows big data mining to be performed using only the computational resources necessary for the needed period of time. In fact, many businesses and organizations have already had their data saved in the cloud. For such businesses and organizations, it is a natural choice to perform data mining in the cloud [6, 7]. However, the monetary cost of utilizing the computational resources in the cloud (referred to as *computation cost*) for big data mining can be unexpectedly high if they are not managed properly. For example, running 100 m4-xlarge Amazon EC2 virtual machine (VM) instances costs $583.00 per day. Thus, the cost effectiveness in the cloud has become a major obstacle for broad applications of big data mining. On this ground, it is a critical issue to analyze the cost effectiveness of big data mining in the cloud, i.e., how to achieve a sufficiently satisfactory result at the lowest possible computation cost.

In many data mining scenarios, achieving the optimal result, e.g., 100% accuracy, is not necessary. Take marketing for example, where data mining is usually performed on a large number of consumers. A reasonable margin of inaccuracy is acceptable. For example, marketers do not need their consumers to be classified with a 100% accuracy. As long as they can obtain a general picture, they are able to make a decision. In fact, in some data mining scenarios, there will never be a 100% accuracy, e.g., weather forecasting and traffic jam prediction. It is possible to achieve high cost effectiveness by stopping the data mining process at a reasonable point in such scenarios because it is often more preferable to achieve a sufficient accuracy, e.g., 99% or 99.9%, at much lower costs, e.g., 10% or 20%, than the cost of achieving a 100% accuracy.

Cost-effective data mining allows big data analytics to be applied in a broader range of fields by more businesses and organizations, especially small and medium sized ones. However, it has not been well investigated by the research community. In this paper, we study k-means, one of the top 10 data mining algorithms [8], to explore and demonstrate the cost effectiveness of big data mining in the cloud.

The remainder of this paper is organized as follows. Section II discusses the related work. Section III introduces the methodology adopted in this study. Section IV presents and analyzes the experimental results. Section V further

discusses the findings of this study. Section VI analyzes the threats to the validity of our experiments. Finally, Section VII concludes this paper and discusses the future work.

II. Related Work

The pay-as-you-go model introduced and promoted by cloud computing has significantly changed the way that IT infrastructure and resources are provisioned and utilized. Since most major advantages offered by cloud computing are built around the flexibility of this cost model, cost effectiveness has attracted many researchers' attention as a core research problem in cloud computing.

There have been many studies on cost-effective computation in the cloud. Ostermann et al. analyzed the performance and cost effectiveness of Amazon's EC2 using micro-benchmarks and kernels [9]. Two similar studies, one conducted by Mehrotra et al. with NASA HPC workloads [10] and the other conducted by Iosup et al. with Many-Task Computing (MTC) workloads [11], both came to the same conclusion that the performance of public cloud services was not high enough for HPC applications. As cloud vendors continuously improved their cloud services over the recent years, more studies have been conducted on the performance as well as the cost effectiveness of public cloud services and achieved more satisfactory results. Berriman et al. studied the cost effectiveness of scientific computing applications in Amazon's EC2 [12] through a comparison between Amazon's EC2 and the Abe high-performance cluster at the National Center for Super Computing in the US. Their study showed that Amazon EC2 cloud offers better performance and value for processor- and memory-limited applications than for I/O-bound applications. Carlyle et al. conducted a similar study that compared the costs of high-performance computation in traditional HPC environments and in Amazon's EC2 environments, using Purdue University's HPC "community cluster" program [13]. Their study showed that an in-house cluster is more cost effective when the organization fulfils three criteria: 1) having sufficient demand that fully utilizes the cluster; 2) having an IT department capable of sustaining IT infrastructure; and 3) having cyber-enabled research as a priority. These constraints, in fact, confirm the flexibility and cost effectiveness of running computation-intensive applications in the commercial clouds. Deelman et al. analyzed the tradeoff between the cost of running computation-intensive and data-intensive applications and their performance in the cloud [14]. Their main finding was that running computation-intensive applications was more cost effective than running data-intensive applications in the cloud. Gupta et al. evaluated and analyzed the performance of HPC applications in cloud. Their experiments showed that current cloud services cannot substitute supercomputers but can effectively complement them [7]. Wang et al. proposed a stochastic multi-tenant framework for investigating the response time of cloud services as a stochastic metric with a general probability distribution [15]. Hwang et al. tested the performance of Amazon's cloud services with five benchmark applications, with a focus on the comparison between the scaling out and the scaling up strategies [16].

Existing work indicates the fast-growing popularity of running computation-intensive applications in the cloud and offers a general picture about the cost effectiveness of big data mining in the cloud through a comparison between the cloud environment and a traditional cluster environment. In this study, we take a look at the issue of cost effectiveness from a different and important perspective – achieving a satisfactory accuracy at a relatively small proportion of the total cost of achieving a 100% accuracy by stopping the data mining process at some point before its completion

III. Methodology

In this section, we discuss the methodology adopted in our case study, including the data mining technique, the data sets, the accuracy calculation method, the cost model and the study procedure.

A. K-means

There is a wide range of data mining techniques that can be adopted for the exploration and demonstration of the cost effectiveness of big data mining in the cloud. Wu et al. systematically discussed the top 10 data mining techniques according to their influence in the research community in [8]. K-means is ranked second to C4.5. It is a simple data clustering algorithm that has been studied intensively and applied widely in both academia and industry. Furthermore, k-means converges – it approaches the final (and optimal) result iteratively. This allows us to calculate and demonstrate the accuracy of the intermediate clustering result, as well as the incurred cost, at each iteration of the clustering process.

The clustering problem is to partition a given data set D into a number of clusters so that the total Euclidean distance between each data point and its closest center is minimized. Solving this problem exactly is NP-hard [17]. In our case study, we employ the local search solution proposed by Lloyd [18]. It is by far the most popular clustering algorithm used in scientific and industrial applications [19]. It follows a simple process:

1. Choose k arbitrary centers $C=\{c_1, c_2, \ldots, c_k\}$;
2. For each $i \in \{1, \ldots, k\}$, set cluster C_i as the set of data points in D that are closer to c_i than to c_j for all $j \neq i$;
3. For each $i \in \{1, \ldots, k\}$, set c_i as the center of C_i:

$$c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \ ;$$

4. Repeat Steps 2 and 3 until C no longer changes, i.e., the process stabilizes.

During this process, the total Euclidean distance between each data point and its closest center is monotonically decreasing one iteration after another. This ensures that no cluster assignment is repeated during the process. Thus, this process will always terminate. Given n data points in D, there are only a total of k^n possible cluster assignments. The time complexity of Lloyd's k-means algorithm is $O(nkdi)$, where n is the number of d-dimensional data points in D, k is the number of clusters and i is the number of iterations needed for the algorithm to complete.

The efficiency of Lloyd's k-means algorithm is sensitive to the k initial centers arbitrarily selected. Inappropriate

initial centers result in excessive iterations and computation time, especially in big data scenarios. There have been many pieces of work on how to select k appropriate initial centers [20-22]. Unfortunately, there is no simple and universally good solution to this problem [23]. In this study, the impact of the randomness in initial center selection on the efficiency of the algorithm must be limited by ensuring the consistency in the selection of initial centers across the experiments. Thus, we employ a method similar to the initial center selection used by Erisoglu et al. [24] that follows the principal of maximum Euclidean distance between initial centers. This method replaces Step 1 in Lloyd's k-means algorithm with a simple process that spreads out the initial centers:

1. Choose k centers $C = \{c_1, c_2, \ldots, c_k\}$;
   a. Select the data point with the maximum Euclidean distance from the origin as the first center c_1.
   b. Calculate the Euclidean distance between each data point and all selected m centers:

$$d(c_i) = \sum_{r=1}^{m} \sum_{j=1}^{p} (x_{r,j} - x_{i,j})^2 \quad i = 1, 2, \ldots, n \quad (1)$$

   where $x_{i,j}$ is d_i's coordinate on the j^{th} (of all p) axis.
   c. Select the data point with the highest $d(c_i)$ as the next center.

Repeat Steps 1.*b* and 1.*c* until a total of k initial centers are selected.

B. Accuracy Evaluation

Accuracy is an important measurement for evaluating the effectiveness of k-means [25]. Lloyd's k-means algorithm employed in this study is a heuristic algorithm and thus does not guarantee an optimal solution to the clustering problem. In order to demonstrate the gradual increase in the accuracy of the clustering result iteration by iteration, we use the final partition achieved by Lloyd's k-means algorithm as the reference partition, noted by P_f. Through the comparison between the partition achieved at each iteration of Lloyd's k-means algorithm, we can demonstrate how the accuracy of the r - 1 intermediate partition, $P_1, \ldots, P_{r-1}$, increases. Here, the *accuracy* is measured by the similarity between $P_1, \ldots, P_{f-1}$ and P_f. In this study, we employ the Rand index proposed in [26] to measure the similarity between $P_1, \ldots, P_{r-1}$ and P_f. The Rand index has been widely used to calculate the accuracy of partitions from a mathematical standpoint [25]. The Rand index measures the similarity between two partitions, P_1 and P_2 of the same data set D. Each partition is viewed as a collection of $n \times (n - 1) / 2$ pairwise decisions, where n is the size of D. For each pair of data points d_i and d_j in D, a partition either assigns them to the same cluster or to different clusters. Thus, the similarity between P_1 and P_2 is calculated as:

$$Rand(P_1, P_2) = \frac{a + b}{n \times (n-1) / 2} \quad (2)$$

where a is the number of decisions where d_i is in the same cluster as d_j in P_1 and in P_2, b is the number of decisions where d_i and d_j are placed in different clusters in both P_1 and P_2. Take P_1 and P_2 in Figure 1 for example. There are two clusters each in P_1: $\{d_1, d_2, d_3\}$ and $\{d_4, d_5, d_6, d_7, d_8\}$, and in P_2: $\{d_1, d_2, d_3, d_4\}$ and $\{d_5, d_6, d_7, d_8\}$. Data points d_1 and d_2 are assigned to a same cluster in both P_1 and P_2. Thus, pair (d_1, d_2) belongs to a. There are 9 such pairs in total, namely, (d_1, d_2), (d_1, d_3), (d_2, d_3), (d_5, d_6), (d_5, d_7), (d_5, d_8), (d_6, d_7), (d_6, d_8) and (d_7, d_8). Thus, a = 9. Data points d_1 and d_5 are assigned to two different clusters in both P_1 and P_2. Thus, pair (d_1, d_5) belongs to b. There are 12 such pairs, namely, (d_1, d_5), (d_1, d_6), (d_1, d_7), (d_1, d_8), (d_2, d_5), (d_2, d_6), (d_2, d_7), (d_2, d_8), (d_3, d_5), (d_3, d_6), (d_3, d_7), (d_3, d_8). Thus, b = 12. Given a = 9, b = 12 and n = 8, we can measure the similarity between P_1 and P_2 by: $Rand(P_1, P_2) = (9 + 12) / (8 \times (8 - 1) / 2) = 0.75$. Suppose P_2 is the final partition achieved by Lloyd's k-means algorithm, P_1 achieves a 75% accuracy. At the final (r^{th}) iteration of Lloyd's k-means process, $P_r = P_f$. Thus, $Rand(P_r, P_f) = 1.0$, which indicates that the process completes with a 100% accuracy.

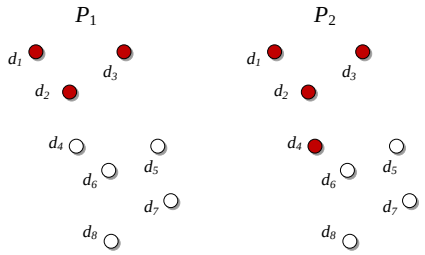

Figure 1. Partitions P_1 and P_2.

C. Cost Model

In our case study, we also measure the computation cost incurred during the clustering process until its completion. Different cloud vendors offer various cost models to accommodate users' various needs. Take Amazon for example, it offers following three EC2 cost models:

- *On-demand*. This model allows users to pay by the hour without long-term commitments or upfront payments.
- *Spot* instances. This model allows users to bid on spare EC2 resources.
- Reserved *instances*. This model allows users to pay with a long-term commitment (1-3 years).

The on-demand cost model is a basic and flexible cost model that is available with various cloud vendors, including Microsoft, Google, etc. Thus, in this study, we employ the on-demand cost model for measuring the computation cost incurred during the k-means process:

$$ComputationCost = UnitPrice \times ComputationTime \quad (3)$$

The computation time is how long the k-means process has taken, which can be easily measured. However, the unit price can vary significantly, depending on the computational resource employed to run the algorithm. Take Amazon's EC2 for example, there are six major categories of EC2 VM instances: Linux, RHEL (Red Hat Enterprise Linux), SLES (SUSE Linux Enterprise Server), Windows, Windows with SQL Standard and Windows with SQL Web. In each of those categories, there are various types of EC2 VM instances available at different unit prices. In the Linux category alone, there are 45 EC2 VM instances of five types, i.e., General Purpose, Compute Optimized, GPU Instances,

Memory Optimized and Storage Optimized. The unit prices of those EC2 VM instances range from $0.0065 to $16.006 per hour. Furthermore, those prices vary at Amazon's data centers in Amazon's 12 different regions around the globe. For example, an x1.32xlarge EC2 VM instance costs $19.341 per hour in Amazon's Singapore region but only $13.338 per hour in its North Virginia region.

For the purposes of simplicity and generality, we use the computation time as an indicator of the computation cost. The reasons are twofold: 1) given a specific cloud resource instance, e.g., a specific Amazon EC2 VM instance, the computation cost and the computation time are positively correlated - longer computation time incurs higher computation cost; 2) given a specific data mining algorithm and the same input, a higher-performance cloud resource instance usually requires less computation time to complete because a more powerful computational resource instance is priced higher than a less powerful one.

Other costs may occur in order to run the k-means algorithm. For example, the data set to be partitioned needs to be stored in the cloud or transferred to the cloud in advance. However, the cost incurred by data storage and data transfer are independent of the k-means process. Thus, in this study, we focus only on the cost incurred by the computation of the k-means process and isolate it from the other costs.

D. Case Study Procedure

Our case study procedure consists of several steps:

1. *Data set preparation*. The data sets to be partitioned in the experiments are prepared.
2. *Data set partitioning*. The data sets are partitioned using Lloyd's k-means algorithm introduced in Section III.A with different *k* in different experiments. During the process of partitioning a data set, the intermediate partition and the computation time taken at each iteration of the algorithm are recorded.
3. *Accuracy calculation*. For each set of experiments, the similarity between the intermediate partitions and the final partition are calculated by using Formula (2) in Section III.B to obtain the accuracy of the intermediate partitions.
4. *Accuracy-time comparison*. For each set of experiments, the accuracy of the intermediate partitions obtained by the algorithm at every iteration is illustrated against the computation time taken by the algorithm by the end of every iteration based on the on-demand model discussed in Section III.C.
5. *Analysis and discussion*. The comparison results are analyzed and discussed.

IV. Experiments

In this section, we present and discuss the experiment platform, the data sets and the corresponding experiments conducted in our case study.

A. Platform

The k-means algorithm was implemented on MATLAB R2014b. All experiments were conducted on a machine with a 3.40 GHz Intel Core i5 processor and 8GB memory. The operating system is 64-bit Windows 7 enterprise.

B. Data Sets

In our experiments, we used two data sets:

- *Gaussian data set*. This data set was synthetically generated based on the Gaussian distribution, following a design for data generation similar to [27] and [28] - two widely acknowledged pieces of work on k-means. First, 72 central points were randomly generated. Then, based on each central point, 13,889 2-dimensional data points were generated according to a Gaussian distribution with standard deviation σ = 0.05 along each coordinate. In total, there were 1,000,008 2-dimensional data points in this 32Mb data set.
- *Road Network data set*. This is a public data set[1] provided by the Center for Machine Learning and Intelligent Systems at the University of California, Irvine [29]. This data set contains the longitude, latitude and altitude information about a road network covering a region of 185 × 135 km^2 in North Jutland, Denmark. There are 434,874 3-dimensional data points in this 20Mb data set.

C. Experiments on Gaussian Data Set

In this set of experiments, for illustration, we ran Lloyd's k-means algorithm to partition the Gaussian data set with k = 2, 4, 8, 16, 32 and 64. The same Gaussian data set was used across the experiments for a fair comparison. According to the method for initial center selection presented in Section III.A, a total of 64 centers are selected, i.e., $C = \{c_1, c_2, \ldots, c_{64}\}$. In the first experiment with k = 2, k-means was run to partition the Gaussian date set into two clusters using c_1 and c_2 as the initial centers. $\{c_1, c_2, c_3, c_4\}$ were selected as the initial centers in the second experiment with k = 4, $\{c_1, c_2, \ldots, c_8\}$ in the third experiment, etc. In this way, we further limit the impact caused by the randomness in the initial center selection on the partitioning of the data sets across different experiments in this set.

Figure 2 demonstrates the increases in the accuracy of the intermediate partitions over time (in seconds) in this set of experiments. Each marker on the lines denotes the intermediate partition at an iteration. As discussed in Section III.A, the increase in k will increase the time complexity of the k-means algorithm. As a result, the algorithm needs to take more iterations and more time to complete. Figure 2 confirms this. More importantly, Figure 2 shows that the k-means algorithm first takes a relatively small number of iterations to reach a high accuracy, and then spends a large number of iterations to converge to the accuracy of 1.0, i.e., 100% accuracy. We refer to this phenomenon as *long tail*. In this set of experiments, we have also run the k-means algorithm with other k and observed similar long tail phenomena. The long tail phenomenon indicates that the k-means algorithm consumed most of the computation time in the middle and

[1] https://archive.ics.uci.edu/ml/machine-learning-databases/00246/

late stages. Take the experiment with k = 64 for example. The k-means algorithm takes three iterations (2.27 seconds) to reach an accuracy of 0.99, i.e., 99%, then proceeds to take 84 more iterations (33.20 seconds) to complete.

Table I summarizes the computation time taken by the k-means algorithm to reach above the accuracy of 0.99, 0.999 and 0.9999 in this set of experiments. It shows that, as k increases, the k-means algorithm tends to spend more time to converge from a 0.99 accuracy to a 1.0 accuracy. With k = 32, the algorithm spends 75.23% (100% - 24.77%) of its computation time for the accuracy to reach from 0.99 to 1.0, while it spends 93.59% (100% - 6.41%) to do the same with k = 64. This finding indicates that, if the user running the k-means algorithm in the cloud does not need a 100% accuracy, they can stop at an early stage with a satisfactory accuracy and save a huge proportion of the monetary cost of getting a 1.0 accuracy. The saving can be considerably large in big data mining scenarios.

D. *Experiments on Road Network Data Set*

In this set of experiments, the k-means algorithm was run to partition the Road Network data set with k=2, 4, 8, 16, 32 and 64. The initial centers for the k-means algorithm were selected in the same way as in the experiments on the Gaussian data set. Figure 3 demonstrates the results. It shows that the long tail phenomenon also exists during the partitioning of the Road Network data set. This confirms our findings in the experiments on the Gaussian data set.

Table II summarizes the computation time taken by the k-means algorithm to reach the accuracy of 0.99, 0.999 and 0.9999 in this set of experiments. On average, the computation

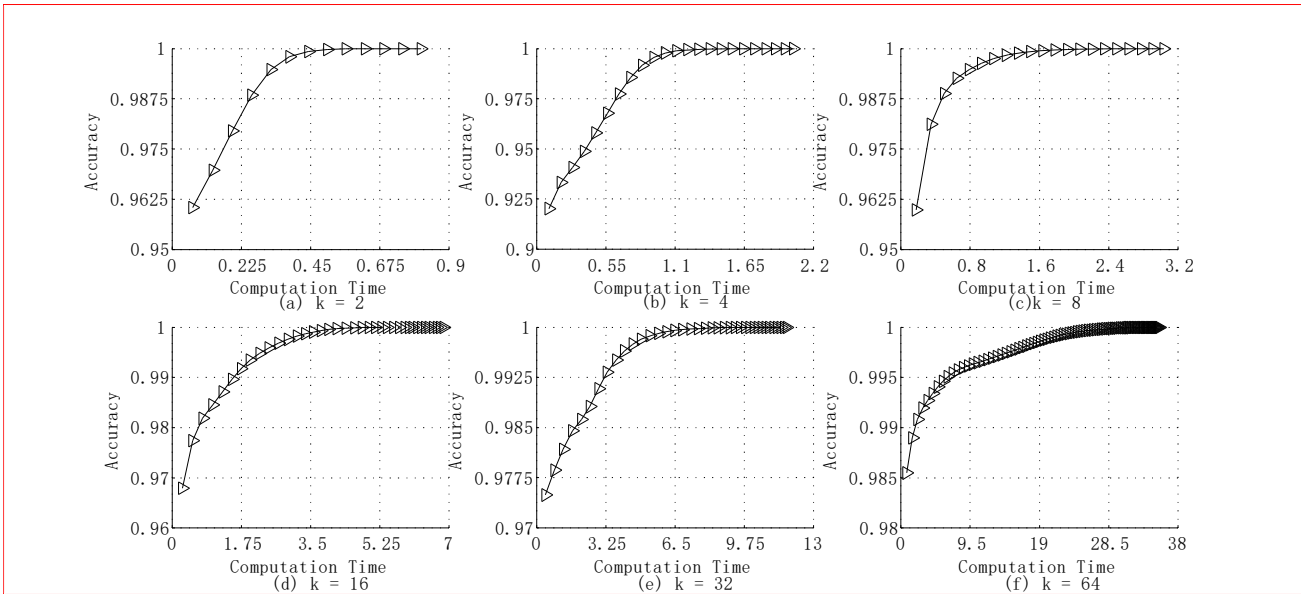

Figure 2. Accuracy and computation time of k-means on Gaussian data set.

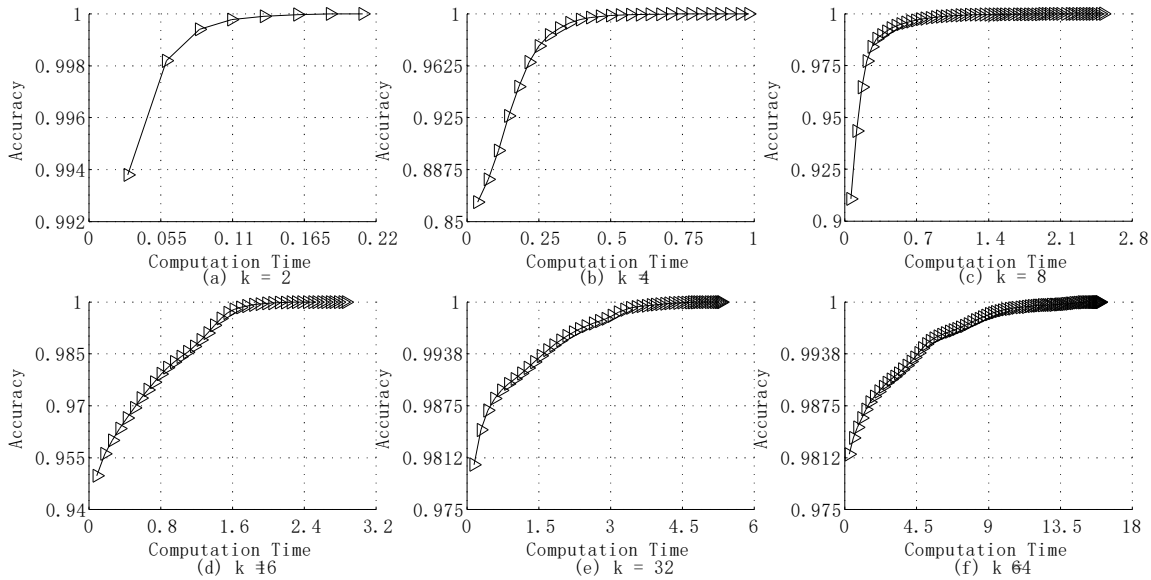

Figure 3. Accuracy and computation time of k-means on Road Network data set.

TABLE I. PERCENTAGE OF COMPUTATION TIME FOR K-MEANS ON GAUSSIAN DATA SET

k	Accuracy		
	≥ 0.99	≥ 0.999	≥ 0.9999
2	39.60%	54.74%	69.76%
4	41.18%	58.79%	71.79%
8	21.21%	49.26%	72.71%
16	25.39%	51.95%	69.86%
32	24.77%	48.75%	70.06%
64	6.41%	59.04%	82.38%
Average	26.43%	59.04%	72.76%

TABLE II. PERCENTAGE OF COMPUTATION TIME FOR K-MEANS ON ROAD NETWORK DATA SET

k	Accuracy		
	≥ 0.99	≥ 0.999	≥ 0.9999
2	14.14%	40.35%	76.25%
4	36.52%	57.94%	82.97%
8	16.08%	40.69%	80.55%
16	46.17%	63.96%	80.61%
32	16.26%	61.86%	84.26%
64	16.50%	62.94%	91.14%
Average	24.28%	54.62%	82.63%

time needed for the k-means algorithm to reach those thresholds are similar to what Table I presents, 24.28% vs 26.43% for ≥ 0.99, 54.62% vs 53.76% for ≥ 0.999 and 82.63% vs 72.76% for ≥ 0.9999. This again indicates that an early stop point with a satisfactory accuracy instead of a 100% accuracy offers a highly cost-effectively clustering solution.

V. DISCUSSION

The experimental results presented in Sections IV.C and IV.D show that, as *k* increases, the k-means takes more iterations (and thus more time) to complete in general. As a result, the long tail phenomenon becomes more significant. Take Figure 2 for example. The algorithm takes a total of 13, 23, 21, 38, 38 and 87 iterations to complete with *k* = 2, 4, 8, 16, 32 and 64 respectively. The long tail phenomenon is also relevant to the scale of the scenario. The data points to be partitioned in the first set of experiments are much more than those in the second set of experiments. Accordingly, the increase in the number of iterations and the computation time with the increase in *k* is more significant in the first set of experiments. Specifically, in the first set of experiments, as *k* increases from 2 to 64, the number of iterations and the total computation time taken by the algorithm increase from 13 to 87 and 0.81 to 35.47 seconds respectively. In the second set of experiments, these increases are from 8 to 88 and 0.21 to 16.01 seconds respectively. This observation indicates that in a real-world big data mining scenario, the cost effectiveness is of extremely high significance.

In the experiments, we limited the impact of the randomness in the initial centers selection on the experimental results by using the method presented in Section III.A. However, through our other experiments, we found that poorly selected initial centers led to significant long tail phenomena. Here, the significance of a long tail is measured by the ratio between the period of time when the accuracy increases rapidly and the period of time when the accuracy increases slowly. As an example, one of the experiments was performed on the same Gaussian data set with *k* = 20 but with random initial centers. Figure 4 shows the corresponding results - a long tail much more significant than those presented in Figure 2 and Figure 3. The inner diagram enlarges the part of the outer diagram where the accuracy exceeds 0.99. The algorithm took only 2.23% of the total computation time to reach the 0.99 (0.9927 to be exact) accuracy. To reach the 0.999 accuracy and the 0.9999 accuracy, it took only 3.79% and 6.50% of the total computation time respectively. In Figure 4, it can also be seen that the computation time taken by the algorithm for each iteration on the long tail is highly uneven, some much longer than the others. This was because the algorithm had to change the centers during the process, which did not happen often in the experiments with better selected centers, as presented in Sections IV.C and IV.D.

The same phenomenon was observed also in our other experiments on the Road Network data set where the initial centers were selected randomly. Figure 5 presents the results of one such experiment with *k* = 10. Specifically, the algorithm took 0.32% of the total computation time to reach the 0.99 accuracy, 2.26% to reach 0.999 and 33.43% to reach 0.9999.

The long tails presented and discussed in Sections IV and V are significant. A major reason is due to the fast convergence of the employed k-means algorithm. The long

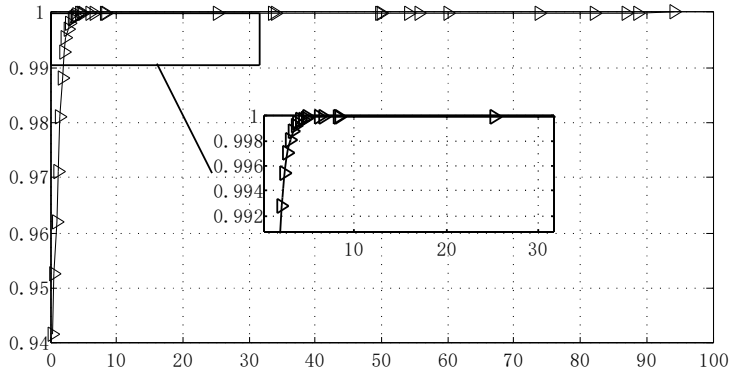

Figure 4. Accuracy and computation time of k-means on Gaussian data set with *k* = 20 and random initial centers.

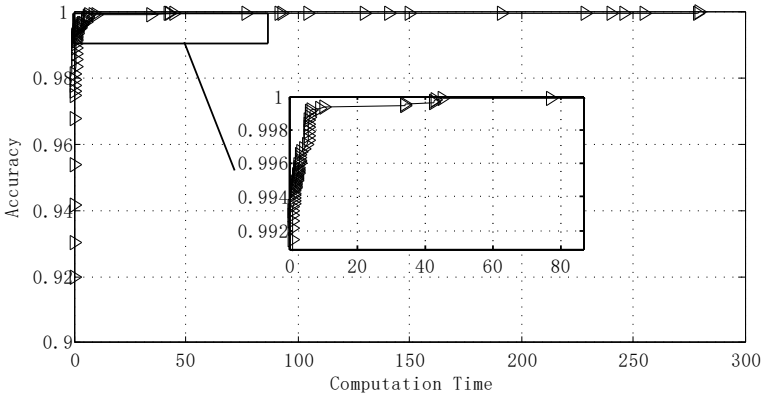

Figure 5. Accuracy and computation time of k-means on Road Network data set with k = 10 and random initial centers.

tail phenomenon might not be as significant in datamining scenarios with other data mining techniques running in the cloud. However, how to achieve sufficiently satisfactory result at the lowest possible computation cost in those scenarios is still critical as long as the data mining algorithm gradually reaches the optimal solution over time.

VI. Threats to Validity

In this section, we discuss the key threats to the validity of our case study.

Threats to construct validity. The main threat to the construct validity of our case study is the adopted metric for evaluating the accuracy of an intermediate partition during the k-means process, i.e., the Rand index. As presented in Section III.B, the calculation of the Rand index relies on the final partition. Thus, it is an *external index*. In most real-world data mining scenarios, especially unsupervised data mining scenarios, *internal indexes* that do not rely on prior knowledge of the dataset, e.g., the final partition, are adopted [30]. Popular internal metrics include the CH (Calinski-Harabasz) index, DB (Davies-Bouldin) index, Silhouette index, Dunn index, etc [31]. These internal indexes might have different ranges from the Rand index's [0, 1]. For example, the DB index and Dunn index are limited to the interval [0, ∞]. As the k-means algorithm proceeds, the increase or decrease in their values might not be consistently correlated with the Rand index. Thus, the Rand index is not a usual choice for real-world mining scenarios, especially unsupervised data mining scenarios. However, this threat to validity is minimal in our case study because at this stage our objective is to explore and demonstrate the possibility of stopping a data mining process at some point during its process to achieve high cost effectiveness. Internal indexes are not suitable because they do not necessarily indicate the true goodness of a partition. On the contrary, the Rand index fulfils this objective by accurately evaluating how close an intermediate partition is to the final partition.

Threats to external validity. The main threat to the external validity of our case study is the representativeness of the data sets used in the experiments. In the experiments, we used the Road Network data set, a data set widely used in a variety of research [32, 33]. This data set has its own characteristics and thus does not exactly represent all data sets. Experiments on a different data set will probably generate results different from what is presented in Figure 3, Figure 5 and Table II. However, the major features of the corresponding figures, e.g., the monotonically increasing accuracy over time, the long tail phenomenon, etc., will be similar. This threat to external validity is further minimized by using a data set randomly generated according to Gaussian distribution in the same way as in [27] and [28]. The results obtained from the experiments on this data set are more representative in general. In the meantime, the similarity in the results obtained from experiments on the Road Network data set and the Gaussian data set indicate that the threat to the external validity of our case study is minimized.

Threats to internal validity. The main threat to the internal validity of our case study is the comprehensiveness of the experiments. The intermediate and final partitions of a data set achieved by the k-means algorithm relies on the prespecified value of k, as well as the accuracies of the intermediate partitions calculated with formula (2). Figure 2 and Figure 3 demonstrate the results obtained from experiments with 6 different k values respectively, i.e., 2, 4, 8, 16, 32 and 64. More experiments with other k values were conducted. Due to space limit, it is not presented in this paper. However, the correlation between computation time and accuracy observed in those experiments are similar to those presented in Figure 2 and Figure 3. Thus, the threat to the internal validity of our experiments is not significant.

Threats to conclusion validity. The main threat to the conclusion validity of the experiment is the reliability of the final partition of a data set as its optimal partition. Finding the answer to a clustering problem is NP-hard [17]. The k-means used in the experiments, as presented in Section III.A, attempts to approximate the optimal partition. Thus, the final partition is not necessarily the optimal partition. As a result, Figure 2 and Figure 3 do not necessarily demonstrate how the accuracy of the intermediate partition of a data set approaches its real optimal partition. However, we believe that the final partition, which is achieved with the k-means algorithm presented in Section III.A, is adequately reliable for demonstrating the long tail phenomenon in the clustering process. The real optimal k-means algorithm is most likely to take more time and result in a more significant long tail

phenomenon. Thus, the threat to the conclusion validity of our experiments exists, however is not significant.

VII. Conclusions and Future Work

To mine big data in the cloud using computational resources offered by cloud vendors, cost effectiveness is a critical issue that has been commonly ignored by the research community. In this research, we investigated this issue by using the k-means algorithm as a case study. The experimental results confirm the significance of the cost effectiveness of big data mining in the cloud. In the experiments, the k-means algorithm took only 0.32%-46.17% of the total computation time to achieve a 99% accuracy. That is up to 99.68% monetary cost saving with an accuracy concession of only 1%. In the future, we will further investigate the cost-effectiveness of data mining in the cloud with internal accuracy indexes and other widely used data mining algorithms.

Acknowledgment

This work is partly supported by Australian Research Council Discovery Project DP150101775.

References

[1] A. Labrinidis and H. V. Jagadish, "Challenges and Opportunities with Big Data," the VLDB Endowment, vol. 5, no. 12, pp. 2032-2033, 2012.

[2] (2012). Bringing Big Data to the Enterprise: What is Big Data. Available: http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html

[3] H. Chen, R. H. Chiang, and V. C. Storey, "Business Intelligence and Analytics: From Big Data to Big Impact," MIS Quarterly, vol. 36, no. 4, pp. 1165-1188, 2012.

[4] G.-H. Kim, S. Trimi, and J.-H. Chung, "Big-Data Applications in the Government Sector " Communications of the ACM, vol. 57, no. 3, pp. 78-85, 2014.

[5] W. Raghupathi and V. Raghupathi, "Big Data Analytics in Healthcare: Promise and Potential," Health Information Science and Systems, vol. 2, no. 3, pp. 1-10, 2014.

[6] C. Shen, W. Tong, J.-N. Hwang, and Q. Gao, "Performance Modeling of Big Data Applications in the Cloud Centers," The Journal of Supercomputing, pp. 1-26, 2017.

[7] A. Gupta, P. Faraboschi, F. Gioachin, L. V. Kale, R. Kaufmann, B.-S. Lee, *et al.*, "Evaluating and Improving the Performance and Scheduling of HPC Applications in Cloud," IEEE Transactions on Cloud Computing, vol. 4, no. 3, pp. 307-321, 2016.

[8] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, *et al.*, "Top 10 Algorithms in Data Mining," Knowledge and Information Systems, vol. 14, no. 1, pp. 1-37, 2008.

[9] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "A Performance Analysis of EC2 Cloud Computing Services for Scientific Computing," Proc. International Conference on Cloud Computing, 2009, pp. 115-131.

[10] P. Mehrotra, J. Djomehri, S. Heistand, R. Hood, H. Jin, A. Lazanoff, *et al.*, "Performance Evaluation of Amazon EC2 for NASA HPC Applications," Proc. 3rd Workshop on Scientific Cloud Computing Date, 2012, pp. 41-50.

[11] A. Iosup, S. Ostermann, M. N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "Performance Analysis of Cloud Computing Services for Many-tasks Scientific Computing," IEEE Transactions on Parallel and Distributed systems, vol. 22, no. 6, pp. 931-945, 2011.

[12] G. B. Berriman, G. Juve, E. Deelman, M. Regelson, and P. Plavchan, "The Application of Cloud Computing to Astronomy: A Study of Cost and Performance," Proc. 6th IEEE International Conference on e-Science Workshops, Brisbane, Australia, 2010, pp. 1-7.

[13] A. G. Carlyle, S. L. Harrell, and P. M. Smith, "Cost-Effective HPC: The Community or the Cloud?," Proc. 2nd IEEE International Conference on Cloud Computing Technology and Science, 2010, pp. 169-176.

[14] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, "The Cost of Doing Science on the Cloud: the Montage Example," Proc. ACM/IEEE Conference on Supercomputing, 2008, p. 50.

[15] Z. Wang, M. Hayat, N. Ghani, and K. Shaaban, "Optimizing Cloud-Service Performance: Efficient Resource Provisioning Via Optimal Workload Allocation," IEEE Transactions on Parallel and Distributed Systems, 2016.

[16] K. Hwang, X. Bai, Y. Shi, M. Li, W.-G. Chen, and Y. Wu, "Cloud Performance Modeling with Benchmark Evaluation of Elastic Scaling Strategies," IEEE Transactions on Parallel and Distributed Systems, vol. 27, no. 1, pp. 130-143, 2016.

[17] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "NP-Hardness of Euclidean Sum-of-Squares Clustering," Machine Learning, vol. 75, no. 2, pp. 245-248, 2009.

[18] S. Lloyd, "Least Squares Quantization in PCM," IEEE Transactions on Information Theory, vol. 28, no. 2, pp. 129-137, 1982.

[19] A. K. Jain, "Data Clustering: 50 Years beyond K-Means," Pattern Recognition Letters, vol. 31, no. 8, pp. 651-666, 2010.

[20] P. S. Bradley and U. M. Fayyad, "Refining Initial Points for K-Means Clustering," Proc. International Conference on Machine Learning, 1998, pp. 91-99.

[21] S. S. Khan and A. Ahmad, "Cluster Center Initialization Algorithm for K-means Clustering," Pattern Recognition Letters, vol. 25, no. 11, pp. 1293-1302, 2004.

[22] A. Likas, N. Vlassis, and J. J. Verbeek, "The Global K-means Clustering Algorithm," Pattern Recognition, vol. 36, no. 2, pp. 451-461, 2003.

[23] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis* vol. 3: Wiley New York, 1973.

[24] M. Erisoglu, N. Calis, and S. Sakallioglu, "A New Algorithm for Initial Cluster Centers in K-means Algorithm," Pattern Recognition Letters, vol. 32, no. 14, pp. 1701-1705, 2011.

[25] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, "Constrained K-means Clustering with Background Knowledge," Proc. 18th International Conference on Machine Learning, Williamstown, MA, USA, 2001, pp. 577-584.

[26] W. M. Rand, "Objective Criteria for the Evaluation of Clustering Methods," Journal of the American Statistical Association, vol. 66, no. 336, pp. 846-850, 1971.

[27] D. Pelleg and A. Moore, "Accelerating Exact K-means Algorithms with Geometric Reasoning," Proc. 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999, pp. 277-281.

[28] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An Efficient K-means Clustering Algorithm: Analysis and Implementation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 881-892, 2002.

[29] M. Kaul, B. Yang, and C. S. Jensen, "Building Accurate 3D Spatial Networks to Enable Next Generation Intelligent Transportation Systems," Proc. 14th IEEE International Conference on Mobile Data Management, Milan, Italy, 2013, pp. 137-146.

[30] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster Validity Methods: Part I," ACM Sigmod Record, vol. 31, no. 2, pp. 40-45, 2002.

[31] E. Rendón, I. Abundez, A. Arizmendi, and E. Quiroz, "Internal Versus External Cluster Calidation Indexes," International Journal of computers and communications, vol. 5, no. 1, pp. 27-34, 2011.

[32] N. Tradisauskas, J. Juhl, H. Lahrmann, and C. S. Jensen, "Map Matching for Intelligent Speed Adaptation," IET Intelligent Transport Systems, vol. 3, no. 1, pp. 57-66, 2009.

[33] B. Yang, M. Kaul, and C. S. Jensen, "Using Incomplete Information for Complete Weight Annotation of Road Networks," IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 5, pp. 1267-1279, 2014.