ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2015

Data Mining Algorithm for Effective Performance Evaluation and Characterization of IO Workload

Dr V.Meenakshi [1], R.A.Shameena Begum [2]

Assistant Professor, Department of Computer Science, R.V. Govt. College, Chengalpet, Tamil Nadu, India [1]

M.Phil Student, Department of Computer Science, Mother Teresa Women's University Chennai, India [2]

ABSTRACT: IO Workload of main components of any system is described by the IO Workload characterization. Workload characterization has been a critical issue for operating system and storage. One such critical method is used for hot-data identification, in which a given logical block address (LBA) is verified to see if it contains frequently accessed data. Hot-data identification not only imposes great impacts on flash-memory garbage collection but strongly affects the performance of flash-memory access and its life time. The advancement of NAND-based storage devices, which bear different physical characteristics from hard-disk-based storage devices, calls for an entirely new way of characterizing IO workloads. To revisit the issue of understanding and identifying the essential constituents of modern IO workloads, NAND based storage device emerges. The analysis is driven by various trace captured from running systems for various standard file system benchmarks. Many of the issues arise in SSD (solid-state drives) design appeared in the memory stack. In solving these difficult problems, there is considerable scope for design choice. The following issues which are relevant for SSD performance are Data placement, Write Ordering, Load Management. As SSDs increase in complexity, existing disk models probably become incomplete for predicting performance. To specify the random write performance, disk lifetime will vary significantly due to the locality of disk write operations. A new model for characterizing this behaviour based on cleaning efficiency is introduced and a new partition based algorithms is suggested for extending SSD lifetime.

KEYWORDS : IO workload characterization; storage and operating systems; SSD; clustering; classification.

I. INTRODUCTION

"Data mining" is the process of extracting hidden patterns from large amounts of data. The goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. It is generally used in a wide-ranging of summarizing practices such as Marketing, Detection of fraud and surveillance. It is also popularly known as Knowledge Discovery in Databases (KDD) which refers to the nontrivial, previously unknown implicit useful information from existing data in databases. Several data mining techniques are Classification, Pattern recognition and Association.

A common approach for classifiers is to use decision trees to partition and segmentations. New segment records can be classified by traversing the tree from the root through nodes and branches, to a leaf represented classes. Clustering and classification are often used for purposes of segmenting the records. The proposed work will focus on challenges related to the integration of clustering and classification techniques. Classification has been identified as an important problem in the emerging field of data mining. Given the goal of classifying large data sets, the focus is mainly on decision tree classifiers. Decision tree classifiers are relatively fast as compared to other classification methods. A decision tree can be converted into simple and easy to understand classification rules.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2015

II. CLUSTERING AND CLASSIFICATION

Two common data mining techniques for finding hidden patterns in data are clustering and classification analysis. It is mentioned in the same breath, they have different analytical approaches. There are a variety of algorithms used for clustering, but they all share the property of repeatedly assigning records to a cluster, calculating a measure (usually similarity, and/or distinctiveness) for identifying the throughput of the system. Cluster analysis has been the most popular statistical technique for automatically dividing the workload into workload classes.

Fig.1. shows the block diagram of steps of evaluation and comparison. Classification Technique (Naive Bayes classifier) is done using WEKA Tool. Classification is two-step process, first, it build the classification model based on training data. All objects of the dataset must be pre-classified and its class label must be given. Secondly, the model that is generated in the preceding step is to be tested by assigning class labels to data objects in a test dataset.

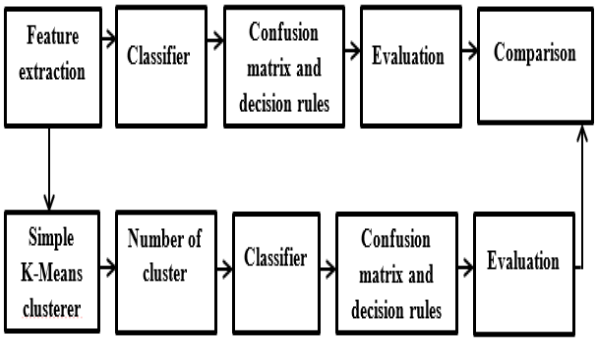

Fig. 1.Block Diagram

A. Clustering Methods

The goal of clustering is to organize objects which are related to each other or have similar characteristics. Clustering is grouping similar objects (item) into same group.

B. Partitioning Clustering

The partitioning method uses a set of M clusters and each object belongs to only one cluster. Every cluster can be represented by a cluster representative. The description of all objects are contained in a cluster. This characterization will depend on the type of the object which is clustered. The real valued data are the arithmetic mean of the attribute vectors for all objects within a cluster provides an appropriate representative while alternative types of centroid may be required in other cases. When the number of clusters is larger than cluster representative, it can be further clustered which develop hierarchy within a dataset.

C. Hierarchical Clustering

The algorithms require a pre-specified number of clusters as input and are non-deterministic. Hierarchical clustering outputs a hierarchical tree structure that is more informative than the unstructured set of clusters formed by flat clustering. This clustering also does not require to specify the number of clusters in progress. The clusters are created either by top-down or bottom up fashion by recursive partitioning. Hierarchical clustering are two types namely Hierarchical Agglomerative methods and Hierarchical Divisive clustering methods.

D. Density Based Clustering

Density-based clustering algorithms try to find clusters based on density of data points in a realm. The basic idea for density-based clustering is that for each instance of a cluster the neighbourhood of a given radius (Eps) has to contain at least a minimum number of instances (MinPts). Density based clustering is based on probability distribution and points from one distribution are assumed to be part of one cluster. This approach identifies the clusters and their parameters.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2015

III. RELATED WORK

[1] Bumjoon Seo et al. (2013) proposed the K-means clustering Algorithm. It is one of the simplest unsupervised learning algorithms that solve the well-known clustering problems. These concept is followed by simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori.

The main idea is to define k centroid, one for each clusters. The centroid need to place in a cunning way because of different location causes different result and the better choice is to place them as possible far away from each other. The result of this loop may notice that the k centroids change their location step by step until no more changes are done. [2] Jen-Wei Hsieh et al. (2005) proposed the Garbage Collection Algorithm to evaluate the performance of the multi-hash-function framework in terms of false hot-data identification.

The performance of the proposed multi-hash-function framework might depend on the hash table size. A naive extension of the multi-hash-function framework was adopted for comparison, in which a hash table of a virtually unlimited size was adopted. [3] Hunki Kwon, Eunsam Kim, Jongmoo Choi, Donghee Lee, Sam H.Noh (2010) worked on Janus-FTL in finding the Optimal Point on the Spectrum Between Page and Block Mapping Schemes. [4] Hsieh J.W, Chang L.P., and Kuo T.W (2005), evaluated an efficient on-line identification of hot data for flash-memory management. [8] Mehrnoosh Sameki et al. (2008) proposed the IO Scheduling Algorithm to improve the efficiency of hard disk utilization, an Operating System (OS) reschedules IO requests to examine more read and write requests in one disk rotation.

The pattern of the reordered IO requests is modified from its original sequence to a random sequence. However, a random sequence of IO requests may impose large performance and endurance overhead to solid state devices. [7] Sungjin Lee et al. (2007) proposed the hot-cold swapping Algorithm. The hot pages are likely to be kept in the hot partition. These blocks which belong to the hot partition are intensively raised. On the other hand, blocks with cold data are rarely updated and hence the erase counts of these blocks becomes much smaller than those of other blocks. So, need comes to adopt a hot-cold swapping algorithm, which tries to balance erase cycles by periodically swapping the blocks containing hot data with blocks having cold data. [5] Jesung Kim et al. (2002) proposed the Single-Linkage Hierarchical Algorithm. The algorithm identifies the Minimum Spanning Tree (MST) of a complete weighted graph with edge weights given by a distance function applied on the vertices, which is equivalent to solving the single-linkage hierarchical clustering problem.

IV. PROPOSED METHOD

To find a set of representative patterns that can characterize IO workloads using trace tool to collect IO trace in operating system over runtime and to use the computational analysis tool to find number of IO trace samples and patterns. The focus is on challenges related to the integration of clustering and classification techniques. Classification has been identified as an important problem in the emerging field of data mining. Given the goal of classifying large IO Workload Trace data sets, the focus is mainly on decision tree classifiers. Decision tree classifiers are relatively fast as compared to other classification methods. A decision tree can be converted into simple and easy to understand classification rules.

A. IO Trace Data Collection

The twenty features of IO trace events are listed in Table 1. Features F1 and F2 indicate the length of a trace in terms of pages and time. Feature F3 is the length of the longest segment. Features F4–F7 represented the total number of Input and Output requests, Break Points, Continued Points and non-random segments. The other features are the ratios (F8–F10) and basic statistics, such as arithmetic means (F11– F13), range (F14), and standard deviations (F16–F20) of other features. The features transform each IO trace sample into a vector of twenty dimensions for downstream analysis. It converts the input trace data into a matrix of 58,758 rows (recall that used 58,758 traces in our experiments) and twenty columns and then the matrix normalized, so that every column has zero mean and unit variance.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2015

Table 1
List of Features for Characterizing IO Traces.

ID	ID Future Description	a	b
F1	Trace size in pages		
F2	Duration of trace (= sum of all intervals between IO requests)		○
F3	Length of the longest segment		
F4	Number of IO requests		
F5	Number of BPs (= segments)		
F6	Number of CPs		○
F7	Number of non-random segments	○	○
F8	Ratio of the number of pages in random segments to that of all pages		○
F9	Ratio of the number of CPs to that of BPs	○	○
F10	Ratio of the number of up-segments to that of all segments		
F11	Average segment length		
F12	Average interval between IO requests		
F13	Average access length of IO requests		
F14	Range of intervals between IO requests		
F15	STD dev. of access lengths		
F16	STD dev. of the starting LBAs of IO requests	○	
F17	STD dev. of the page indices of BPs		
F18	STD dev. of the page indices of CPs		
F19	STD dev. of the starting page indices of random segments		
F20	STD dev. of intervals between IO requests		

a - The features selected by backward elimination
b - The features used by tree-based model

A model has been built for classifying IO traces first to see what type of IO traces appear in data storage systems. The particular number of all available patterns may be difficult to determine but is expected to find a small number of representative patterns. The patterns of IO traces are automated using Weka tool[9] that patterns therefrom. The clustering is a perfect fit for using unsupervised learning enable us to find novel and previously unnoticed patterns besides known to expected ones. Classification is useful when the set of possible classes is already determined but as a supervised learning techniques. It cannot discover novel patterns unlike clustering.

B. IO Trace Data Description

By default, blkparse expects to run in a post-processing mode, where the trace events have been saved by a previous run of blktrace and blkparse. blktrace and blkparse are combining event streams and dumping formatted data. Blkparse are run in a live manner concurrently with combining it with the live option for blktrace and blktrace by specifying -i - to blkparse. An example is:

```
% blktrace -d /dev/sda -o - | blkparse -i -
```

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2015

The number of blkparse batches event are set via the -b option, the default is to handle events in batches of 512. The event traces in blktrace is saved with different output names (via the -o option to blktrace), and the same input name is specified via the -i option. The resultant data can be controlled via the -f or –F options.

C. Decision Tree Classification

Decision-tree-based classifier for classifying IO traces in runtime is constructed using Hierarchical classification method. This section explains how to train the classifier and validate it with real traces. Also, the proposed tree-based classifier is compared with other existing state of the art classifiers in terms of classification accuracy and run time. The purpose of the length of trace samples is also considered for classification performance.

D. K-means clustering algorithm

An algorithm for partitioning (or clustering) N data points into K disjoint subsets S_j containing data points so as to minimize the sum-of-squares criterion

$$J = \sum_{j=1}^{K} \sum_{n \in s} | x_n - \mu_j |^2$$

Where $x_{n\ is}$ denotes a vector representing the n^{th} data point and u_j is the geometric centroid of the data point in s_j.

E. Calculate Median Value

To calculate median silhouette value s(i) of all cluster samples.

$$S(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Where $a(i)$ represents the average distance of sample i with all other samples within the same cluster, and $b(i)$ denotes the lowest average distance to i of the other clusters. By definition, a silhouette value can range from −1 to +1. +1 represents the perfectly clustered sample, whereas −1 indicates the opposite.

V. PERFORMANCE COMPARISON AND RESULT ANALYSIS

The performance of different classifiers approaches are compared in terms of their classification error and running time. Figure.1 compares three classification methods Naive Bayes, Simple K-means, Logistic Regression in terms of their accuracies. The accuracy of the logistic regression-based classifier is the highest, and that of the Naive Bayes classifier is the lowest. The performance of the Simple K-means[6], one of the most popular classifiers, is similar to that of the best one.

The graph shows the running time required for classifying 58,758 IO traces by the methods used for comparison. As expected, the proposed method requires the least amount of time to complete the classification task. The advantage in running time obviously comes from the simplicity of the tree-based model.

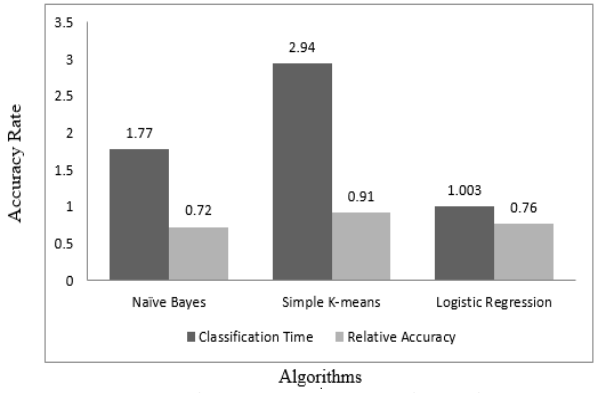

Fig. 2 Performance Comparisons of Classifiers.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2015

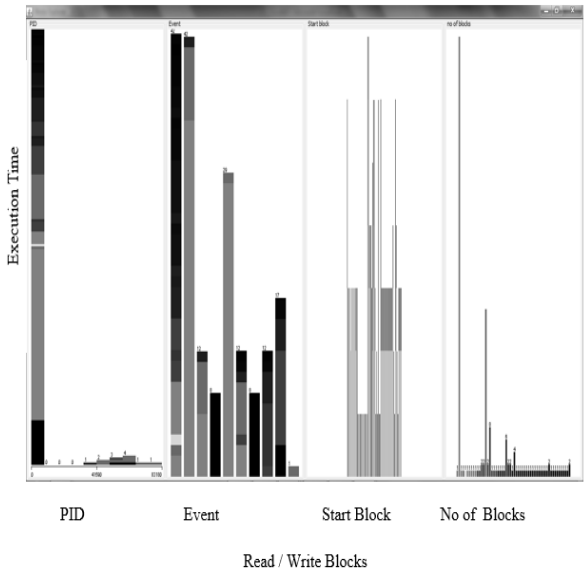

Fig. 3 Classification of Block IO Workload

When executed in an embedded CPU, the proposed classifier would further widen the performance gap among the ratio of the number of continued points and the number of break points, and the standard deviation of the starting LBAs of IO requests. For IO classes, nine essential workload classes and three random, three sequential and three mixture classes are considered. Finally a classification model of IO traces and is developed a pattern classifier is constructed for classifying IO traces in runtime.

The results of various data mining algorithms are shown in Figure 2 shows that the Simple K-mean clustering algorithm provides better results than hierarchical and DBSCAN algorithm. Simple k-mean is a faster and safer algorithm than the other data mining algorithms are used.

VI. CONCLUSION

The novel IO workload characterization and classification is done based on the data mining approach. In this method for IO workload clustering and simple k-means algorithm yields the best silhouette value despite its simplicity. The work needs to be done in the general field of I/O cache performance. Each file type has a distinct size distribution and reuse ratio. File type and size information can direct I/O requests to different parts of an attribute cache. The attribute cache is allocated a wide amount of space to capture these requests. The portion of the cache should have small blocks to effectively capture the temporal nature. Small executable and data files tend to have temporal access patterns, whereas large executable and data files tend to have sequential access patterns. A cache should direct large files to a sequential sub cache, and smaller files to a temporal sub cache. The sequential sub cache has large blocks to capture sequential locality while the temporal sub cache has small blocks to minimize unused space. I/O caches should make use of file types to improve their efficiency. The future work is to develop FTL techniques that can effectively utilize the classification model proposed in this work.

REFERENCES

[1] Bumjoon Seo, Jaehyuk Cha, Jongmoo Choi, Sooyong Kang, Youjip Won, Sungroh Yoon(2013), ”IO Workload Characterization Revisited: A Data-Mining Approach”, in IEEE Transactions on Computers, accepted for Publication, Vol. 6, No. 1, pp. 1541-1551.

[2] Jen-Wei Hsieh, Li-Pin Chang, Tei-Wei Kuo (2006), ”Efficient On-line Identification of Hot Data for Flash-Memory Management”, IEEE Transactions on Computers, Vol. 6, No.1, pp. 581-590.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2015

[3] Hunki Kwon, Eunsam Kim, Jongmoo Choi, Donghee Lee, Sam H.Noh (2010), "Janus-FTL: Finding the Optimal Point on the Spectrum Between Page and Block Mapping Schemes " Proc. International Conference on Embedded software, Vol. 4, pp. 169-178.

[4] Hsieh J.W, Chang L.P., and Kuo T.W (2005), "Efficient on-line identification of hot data for flash-memory management", International Symposium in Applied computing, Vol. 1, pp. 838–842.

[5] Jesung Kim, Jong Min Kim, Sam H. Noh, Sang Lyul Min and Yookun Cho (2002), "A Space-Efficient Flash Translation Layer for Compact Flash Systems", IEEE Transactions on Consumer Electronics, Vol. 48, No. 2, pp. 366-375.

[6] Sungjin Lee, Shin, Kim Y.-J and Kim J (2008), "LAST: Locality-Aware Sector Translation for NAND Flash Memory-Based Storage Systems", International symposium in Operating Systems Design and Implementation, 2008, Vol. 42, pp. 36–42.

[7] LiPin Chang (2007), "On Efficient Wear Leveling for Large Scale Flash Memory Storage Systems", Proc. International Conference in Applied Computing, Vol. 6, pp. 1126-1130.

[8] Mehrnoosh Sameki, Amirali Shambayati (2008), "An IO Scheduling Algorithm to Improve Performance of Flash-Based Solid State Disks", International Conference in Hossein Asadi Sharif University of Technology, Vol. 4, pp. 833-762.

[9] Priya Kakkar, Anshu Parashar (2014), "Comparison of Different Clustering Algorithms using WEKA Tool", International Journal of Engineering and Science , Vol. 1, No. 2, pp. 633-662.