International Journal of Grid and Distributed Computing
Vol. 9, No. 2 (2016), pp.227-244
http://dx.doi.org/10.14257/ijgdc.2016.9.2.20

# Risk Identification Method for Cloud Computing Safety based on LSA-GCC and LSA-SAM

Fan Lin[1]*, Wenhua Zeng[2], Yue Wang[3]

[1, 2,3]*Software School, Xiamen University, Xiamen*
*iamafan@xmu.edu.cn*

***Abstract***

*This paper proposes a generalized cluster risk evaluation model by applying a data mining method to the cloud computing risk evaluation. The model maps data sets into a semantic space via singular value decomposition (SVD), uses a clustering algorithm to classify them and to extract the prototype vector of a particular category from clustering results, and assigns a definite weight to each category so as to set up an initial prototype vector model. The model is taken as the basis for risk evaluation of information system. After the data to be evaluated were mapped to the same semantic space, they are calculated with the prototype vector of each category, so as to obtain the similarity of the category, and the cumulative sum of the similarity with the weight of the corresponding category comes out. Finally, a mean value is calculated to obtain the risk value of the data to be evaluated, namely, the risk value of the occasion when the data is obtained. In this paper, the safety risk information is obtained from the operating system log and Web application server log of a virtual host; the Latent Semantic Analysis-based Generalized Cluster Classifier (LSA-GCC) is adopted and the MapReduce-based LSA-GCC and LSA-SAM parallel acceleration experiment is conducted. The experimental results show that in a cloud computing environment of large-scale parallel processing, the method used in this paper can identify the log events of a cloud computing system and conduct risk prompt rapidly.*

***Keywords****: LSA; Risk Identification; MapReduce; GCC; SAM*

## 1. Introduction

Since a cloud computing system consists of a number of virtual computational nodes, the nodes produce quantities of log text data, presenting exponential growth [1]. Therefore, how to organize and manage these log text data efficiently has become an important and pressing problem. Text classification is a process of allocating texts to a predefined category and an important method used to efficiently organize and manage a large number of text data [2]. As an important application in the field of data mining, text classification has been studied by quite a few scholars and many classification methods have been proposed. The most common classification algorithms are algorithm of decision trees, Rocchio, Naive Bayes, neural network, support vector machine, Linear Least Squares Fit (LLSF), KNN, Genetic Algorithm, maximum entropy, Generalized Instance Set, *etc*.[3,4,5].

A cloud computing system contains various monitoring node, network components and virtual host nodes. These nodes are monitoring the operational states of hosts and networks from different perspectives, and a large number of alarms and logs produced by them are correlated [6, 7]. To analyze log information of a single monitoring node is a conventional security situation evaluation method. Single data source and uncertain monitoring node bring about inaccurate analysis results. Also, the conventional security

Fan Lin is the corresponding author.

ISSN: 2005-4262 IJGDC

situation assessment method does not allow for the incidence relation that often exists among synonyms, polysemy and words in a natural language.

Therefore, the paper presents LSA-SAM (LSA-based Security Assessment Model)[8], *i.e*. the latent semantic analysis-based network security situation assessment model, takes logs of several correlated performance monitoring nodes as the data source[9], and uses the method of LSA-GCC to make predictive analysis of the tendency of security situation[10]

## 2. LSA-based Generalized Cluster Classifier

A With the rapid development of computer information technology, massive data are produced every day. To process and analyze these data, it is necessary to consider factors such as time cost, computing cost and semantic structure of natural language. In this case, we propose a Latent Semantic Analysis-based Generalized Cluster Classifier (LSA-GCC) to take full advantage of the efficiency of LSA and Rocchio algorithm [11, 12].

First of all, the classifier maps training sets to a low-dimensional semantic space, constructs a prototype vector model of the subclasses by adopting a constrained single pass clustering algorithm (CSPC) and Rocchio algorithm under all categories of the training sets, and based on the model, solves the similarity of each category in the sample and model to be classified, and selects a category of highest similarity to serve as the category to which the sample to be classified belongs [13]. CSPC covers the potential subclass of each category, so the model built by LSA-GCC is superior to the Rocchio model; also, LSA-GCC model is built in the semantic space, meeting the requirements described herein [14].

The basic thought of the Rocchio algorithm is to build a prototype vector specific to the training text set for each class, and the computational formula of the prototype vector is as follow:

$$C_j = \alpha \frac{1}{|D_j|} \sum_{d_m \in D_j} d_m - \beta \frac{1}{|D - D_j|} \sum_{d_n \in D - D_j} d_n \quad (1)$$

Wherein, Cj is the prototype vector created based on category Cj, D is the training set of the whole text, and Dj and |Dj| represent document set and document quantity in class, $\alpha$and$\beta$ are used for weighing the importance of positive sample set and negative sample set. The process of producing the prototype vectors which form a Rocchio model can be considered as a learning process. Set a text T, calculate the similarity of the text with each prototype vector, and allocate the text T to the category with the highest similarity.

## 3. Introduction of LSA-GCC Algorithm

When the number of the training text set is relatively larger, there will be a problem for text classification, namely high computing cost. In most cases, high computing cost is inapplicable to applications with real-time request [15]. An effective method that can make up such defect is to build a general cluster based classification model. The model can replace the original training sample set and conduct classification to the text. In the meantime, since the model is the summary of the original training sample set (can be called as the "center of mass" of the original training sample set), its classification effect will be better [16].

 

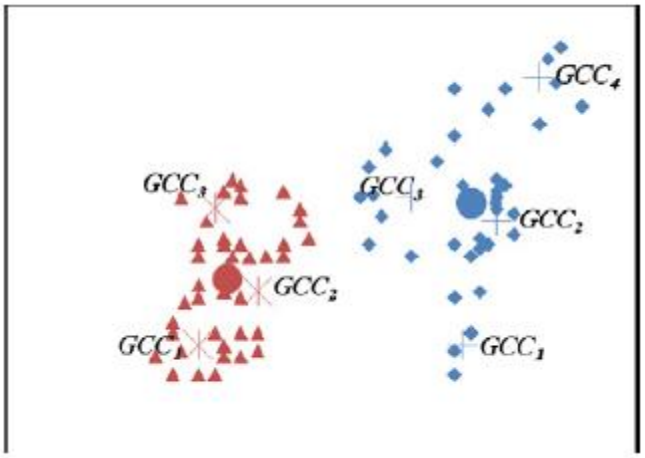


**Figure 1. Subclass-based Prototype Vector Example**

As an effective linear classifier, Rocchio builds the prototype vector model of the training sample set via an efficient clustering method [17]. Therefore, the vector model built by Rocchio is one of the effective methods to improve the computing performance. Nevertheless, there are two obvious defects in Rocchio algorithm. The first defect is that data space is supposed as a group of linearly separable hyper-planar area, however distribution of a lot of data of the real world shall be nonlinear; the other defect is that, it only builds one prototype vector for each category. As clustering is an unsupervised machine learning method, it is feasible to excavate the relationship among the latent fine grits in the training sample set with the method of clustering. Therefore, we present a constrained single pass clustering algorithm (CSPC) to train the training sample set, so as to obtain category sets of the fine grits [18].

The prototype vectors from which the category sets are built are named as generalized prototype vector sets (GCCs)[19]. Then, the original training sample sets are replaced by GCCs to construct a generalized clustering based classification model. Suppose that the triangle represents a large category, including three subclasses; and the square represents the other big class, including four subclasses. In the Rocchio algorithm, the triangle category and square category are represented by their own generalized prototype vectors (namely there are only two "centers of mass") respectively. Under the ideal LSA-GCC model [20]. However, according to CSPC algorithm, the triangle category will produce three subclasses, and the square category will produce four subclasses; then according to the Rocchio algorithm, the generalized prototype vectors are produced for these two categories corresponding to their respective subclasses (namely there will be seven "centers of mass"). Finally, we conduct further calculation with the generalized prototype vectors rather than all the training sets.

Different classifications are represented by different characteristic words, and prototype vectors constructed by LSA-GCC are distributed nonlinearly in different locations of data space [21]. Therefore, LSA-GCC can well make up the defects of Rocchio algorithm, enhancing the semantic expression capability of the model constructed. Since LSA-GCC originates from the prototype vectors and prototype vectors are the "centers of mass" of a training sample set, the model is insensitive to a single training sample. Furthermore, the LSA-GCC based on prototype vector reduces the number of training texts to greatly accelerate the decision-making process of subsequent classification to some extent. Accordingly, the method proposed by this paper can achieve good effects whether from efficiency or effectiveness.

## 4. LSA-GCC Log Analysis Model

We have adopted Rocchio algorithm to build the algorithm model (an improved Rocchio model) combined with a clustering algorithm so as to make preparations for the late-period classification [22]. To ensure the scalability and applicability of the method proposed by the paper, the clustering algorithm with scalability and applicability is required to be applied to cluster the large-scale texts.

Large-scale text clustering is a problem of high-dimensional data clustering analysis, which enables most traditional clustering algorithms to be invalid. Various clustering algorithms (such as subspace clustering and incidence clustering) have been proposed to solve the problem of large-scale and high-dimensional data [23]. The incremental clustering algorithm is featured by low time consumption, non-iteration, and one-time text scanning. Based on such characteristics, the incremental clustering algorithm can solve the aforementioned problem.

The single-pass clustering algorithm is an incremental one, with approximately linear time complexity [24]. Therefore, the algorithm adopted by the paper is a CSPC (step of the whole clustering process is described as is shown in Item (1)-(12) of the Figure). This algorithm only scans the text for one time and then incorporates the text into the category most similar to the text (description of the corresponding steps is shown in Item (7)-(8) of the Figure).

**Algorithm.1 Construct Pseudo Codes of the Prototype Vector Set**

Input: training set WS-DREAM, clustering threshold r
Output: GCCs (each GCC is corresponding to one class)
Procedure GCC (D)
(1) Set mc as the category set, mgcc as the set of GCCs,
(2) Initialize mc and mgcc to be empty,
(3) Repeat,
(4) Enter a new text p,
(5) Calculate the similarity of all classes in the text p and category set mc,

$$C_q = sim(D_q, D_{d_i}) = \frac{\sum_{i=1}^{k}(D_{d_{qi}} \cdot D_{q_i})}{\sqrt{\sum_{i=1}^{k}(D_{d_{qi}})^2 \cdot \sum_{i=1}^{k}(D_{q_i})^2}} \quad (2)$$

(6) Find out the category ci0 of the highest similarity with text p;
(7) If sim (p,ci0)>=r,
(8) Merge p into the classci0;
(9) Otherwise,
(10) Create a new category ci0,
(11)And add the new category ci0 to the category set mc,
(12) Till the sample in the training set is empty.

A category set $m_c = \{C_1^0, C_2^0, C_3^0, ..., C_i^0\}$ can be obtained through CSPC algorithm clustering, and then the category set is used to build LSA-GCC model.

Prototype vectors can improve the robustness of classification model, extracting definite correlated characteristics to some extent. The paper uses the Rocchio algorithm to build the GCCS of prototype vector of the category (the step is shown in Item (15) of the Figure). At this step, each category builds prototype vectors via the following formula:

 

$$gcc_i = \alpha \frac{1}{\left|C_i^0\right|} \sum_{d_m \in C_i^0} d_m - \beta \frac{1}{\left|D - C_i^0\right|} \sum_{d_n \in D - C_I^0} d_n \quad (3)$$

In the formula, gcci represents the prototype vector of the ith category Ci0 in LSA-GCC, and Ci0 and |Ci0| represent the document and the quantity in category Ci0respectively. The intention of the formula is quite clear: the document in the category Ci0 is considered as positive sample and the rest as negative sample, then the documents shall be accumulated and normalized; finally, the vector subtraction is made on the vectors of positive and negative samples to extract characteristics with identification ability, thereby constructing the prototype vector model $gcc_i$ of the category Ci0. These two parameters α and β are used to weigh the weight of the positive and negative samples, so as to obtain the optimal $gcc_i$(in this paper,$\alpha$=5$\beta$).

In the process of clustering, some small category sets will be generated, and the so-called small category means that it only contains a small amount of text. These texts are very likely to be abnormal data and could also contain important information used for classification or only be "noise". Therefore, to speed up the classification of the models, the paper sets up a threshold value to integrate or filter the small categories (the step is described in Item (13) and (14) of the Algorithm.1.

The value of the clustering threshold r of Step (7) in Algorithm.1 could affect the quality and efficiency of the whole clustering process. If the value "r" increases, the number and time consumption of subclasses will increase as well. To obtain a stable threshold value "r", the paper has adopted a sampling technology to determine the threshold value r [4, 5, 6].

The concrete steps are described as follows:

Step 1: Select $N_0$text pairs in the text set.

Step 2: Calculate the similarity of each text pair.

Step 3: Calculate all the similarities obtained from Step 2, and obtain the average similarity value "savagism".

Step 4: value of the threshold value "r" is $\epsilon$*savagism, wherein $\epsilon$ >=1.

In the above steps, N0 represents the quantity of the text selected, avgsim is the mean value of N0 for text similarity, $\epsilon$ is a parameter used to adjust the value of threshold value "r" according to different application situations. When "NO" a relatively large value, "avgsim" is stays stable. In this research, N0 is 8000. The result shows that the experiment has obtained the cluster result of higher quality under the circumstance where value range of $\epsilon$ is 5~13.

The LSA-GCC model and Rocchio model are compared and the experimental results are shown in Tables 1 and 2 below.

**Table 1. LSA-GCC**

| r | Right | Error | Uncertain | Recall(%) | | | | | | Accuracy(%) |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Error event | Information event | Failure Audit event | Warning event | Success Audit event | Success event | |
| 0.4 | 9914 | 86 | 0 | 65.19 | 99.83 | 99.60 | 99.80 | 100.0 | 54.24 | 99.14 |
| 0.5 | 9888 | 84 | 28 | 44.45 | 99.92 | 99.60 | 99.80 | 100.0 | 54.24 | 98.88 |
| 0.6 | 9888 | 73 | 39 | 44.45 | 99.92 | 99.60 | 99.80 | 100.0 | 54.24 | 98.88 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0.7 | 9843 | 66 | 91 | 44.45 | 99.92 | 99.31 | 99.51 | 99.83 | 6.80 | 98.43 |
| 0.8 | 9717 | 20 | 263 | 8.89 | 98.95 | 99.31 | 97.74 | 98.97 | 11.86 | 97.17 |
| 0.9 | 9036 | 6 | 958 | 7.41 | 72.97 | 99.19 | 93.52 | 98.61 | 11.86 | 90.36 |

**Table 2. Rocchio**

| r | Right | Error | Uncertain | Recall (%) | | | | | | Accuracy (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Error event | Information event | Failure Audit event | Warning event | Success Audit event | Success event | |
| 0.4 | 6757 | 103 | 3140 | 56.29 | 100.0 | 99.82 | 100.0 | 33.09 | 13.55 | 67.57 |
| 0.5 | 5704 | 70 | 4226 | 44.44 | 100.0 | 99.82 | 100.0 | 10.94 | 11.86 | 57.04 |
| 0.6 | 5552 | 27 | 4421 | 8.14 | 95.74 | 99.82 | 99.80 | 10.94 | 11.86 | 55.52 |
| 0.7 | 5399 | 26 | 4575 | 8.14 | 91.45 | 99.36 | 99.60 | 10.06 | 11.86 | 53.99 |
| 0.8 | 4502 | 2 | 5496 | 8.14 | 54.56 | 99.25 | 97.74 | 10.06 | 11.86 | 45.02 |
| 0.9 | 3414 | 1 | 6585 | 8.14 | 10.56 | 99.25 | 93.52 | 10.06 | 11.86 | 34.14 |

For the parameters in Figure 1 and Figure 2, “r” is the classification threshold; “Right” refers to the correctly classified record count; “Error” refers to the mistakenly classified record count; “Uncertain” refers to the record count of all the categories which cannot be assigned to the built models under the designated classification threshold value; “Error Event”, “Information Event”, “Failure Audit Event”, “Warning Event”, “Success Audit Event” and “Success Event” are the determination of security events; “Recall” is the recall rate of the determinations; and “Accuracy” is the accuracy of the models.

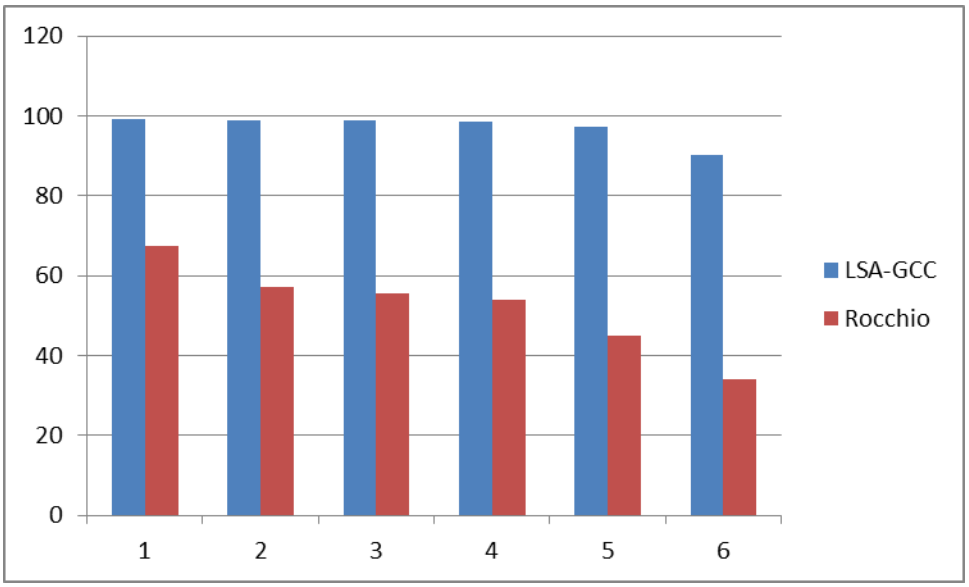


**Figure 2. Comparison of Identification Accuracy between Two Methods LSA-GCC and Rocchio**

## 5. LSA-based Risk Identification Framework of Cloud Computing System

### 5.1 Risk Identification Indicators

**5.1.1 Alarm Purity:** This indicator mainly measures the detectability of a model for anomalous events. Alarm means that a warning message is given out based on whether the risk value worked out by the model is lower than a certain threshold.

(1) Detection Precision Rate
The percentage of the correctly detected alarm to the total number of alarm can be represented as:

$$DPR = \frac{\#RA}{\#A} \times 100 \quad (4)$$

Wherein, "#RA" (Right Alert) is the correctly detected total number of alarm, and "#A" is the total number of alarm.

(2) False Alarm Rate
The percentage of false alarm to the real alarm can be represented as:

$$FAR = \frac{\#EA}{\#TA} \times 100 \quad (5)$$

Wherein, "#EA" (Error Alert) is the total number of false alarm, and "#TA" (True Alert) is the total number of real alarm.

**5.1.2. Model Fitting Accuracy:** This indicator mainly measures the determination ability of a model for all security events. Determination means that the category of security event is determined based on whether the risk value worked out by the model is lower than a certain threshold.

(1) Determine Accuracy
The percentage of accurate determination to all determinations can be represented as:

$$DA = \frac{\#RD}{\#AD} \times 100\% \quad (6)$$

In the above formula, #RD (Right Determine) is the total number of accurate determinations, and #AD (All Determine) is the total number of determinations.

(2) Determine Recall
The percentage of accurate determination to real determination can be represented as:

$$FAR = \frac{\#EA}{\#TA} \times 100 \quad (7)$$

In the above formula, #RD (Right Determine) is the total number of right determination; #TD (True Determine) is the total number of real determination.

## 6. Risk Identification Method

Cloud computing network contains a large number of performance monitoring nodes, network components, and virtual host nodes. These performance monitoring nodes monitor the operational states of hosts and networks from different perspectives, and a large number of alarms and logs produced by them are correlated.

To analyze log information of a single monitoring node is a conventional security situation evaluation method. Single data source and uncertain monitoring node bring about inaccurate analysis results. Also, the conventional security situation assessment

method does not allow for the incidence relation that often exists among synonyms, polysemies and words in a natural language.

Therefore, the paper presents LSA-SAM (LSA-based Security Assessment Model), *i.e.* the latent semantic analysis-based network security situation assessment model, takes logs of several correlated performance monitoring nodes as the data source, and uses the method of LSA-GCC to make predictive analysis of the tendency of security situation.

The Figure below is the evaluation process of evaluation model, as shown in Figure 6.

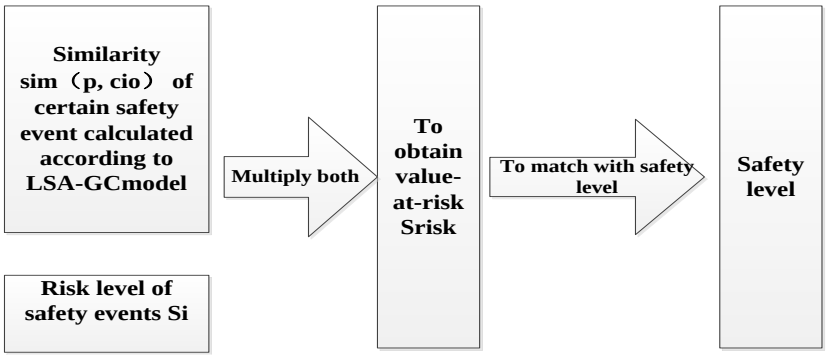


**Figure 3. LSA-based Security Assessment Model (LSA-SAM)**

In this paper, a security event set considers that: the event log set of Windows operating system in a virtual machine is SES={SUCCESS EVENT, SUCCESS AUDIT EVENT, INFORMATION EVENT, WARNING EVENT, FAILURE EVENT, ERROR EVENT}; and in the typical Tomcat Webservice server, the event log set collected with Log4j is SESt={AUDIT, DEBUG, INFO, WARN, ERROR, FATAL}. Safety values are assigned according to the corresponding security events (as shown in Table 1). Security grades "Gs" are {safe, relatively safe, general, unsafe}, and each grade falls within a definite numerical range. A risk value is obtained via the security assessment model. If the risk value falls within the numerical value of a certain security grade, it belongs to such security grade. The range of security grade is shown in Form 3:

**Table 3. Weight Value Grading Classification of Security Events**

| OS security events | WebService events | Weight value |
|---|---|---|
| SUCCESS EVENT | AUDIT | 1.0 |
| SUCCESS AUDIT EVENT | DEBUG | 0.8 |
| INFORMAION EVENT | INFO | 0.6 |
| WARNING EVENT | WARN | 0.4 |
| FAILURE EVENT | ERROR | 0.2 |
| ERROR EVENT | FATAL | 0.0 |

**Table 4. Risk Level Range**

| Level | Value range |
|---|---|
| Safe | 8.5~1.0 |
| Relatively safe | 6.0~8.5 |

 

| | |
|---|---|
| General | 3.5~6.0 |
| Unsafe | 0~3.5 |

## 7. Experiment

The experiment uses the LSA-SAM algorithm to verify the overall situation of security risk. In the experiment, we suppose that only one security event occurs in the same time slice and a risk value is assigned to a given security event. Some experimental results are shown in Figure 4 below:

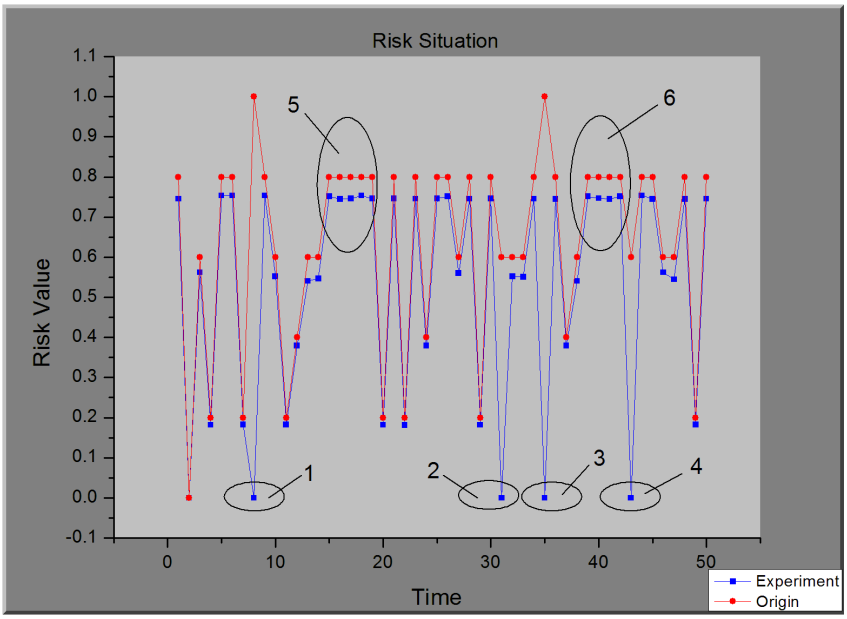


**Figure 4. Risk Situation Simulations**

In Figure 4, the blue line represents the experimental result, and the red line represents the original risk conditions. Figure 7 shows that the experimental result is basically consistent with the original risk conditions. However, we discover some inconsistent phenomena, such as the situation of the four points 1, 2, 3 and 4 marked in the Figure 4. As mentioned above, the security situations at risk are set. The risk value range of unsafe event is 0~3.5, while risk values of these four points are 0. Therefore, the four points cannot be classified as the situation of giving an alarm under abnormal conditions because they are unable to be judged. However, they fall within the security range corresponding to the original conditions, where alarms should not be given. What is the reason for that? In fact, the four points are uncertain conditions in Experiment 1, namely the conditions not covered by the model. Since they are unable to be allocated to the corresponding classes, they are taken as anomalous events, which shall be disposed by the administrator. In the experiment, Mark 1 and 3, Mark 2 and 4 are respectively security events. If an alarm is still given when an initial alarm reoccurs after being determined by the administrator as a security event, then it should be a problem required to be solved. In this case, the advantage of the LSA-SAM model can be fully taken to learn according to the determination of the administrator; the point is considered as a new small category and the "center of mass" of such category is worked out. After learning, the risk conditions are showed in Figures 4~7. It can be observed after learning that the conditions of false alarm

are improved. We also compare the models both before improvement and after improvement, as shown in Figure 8 below.

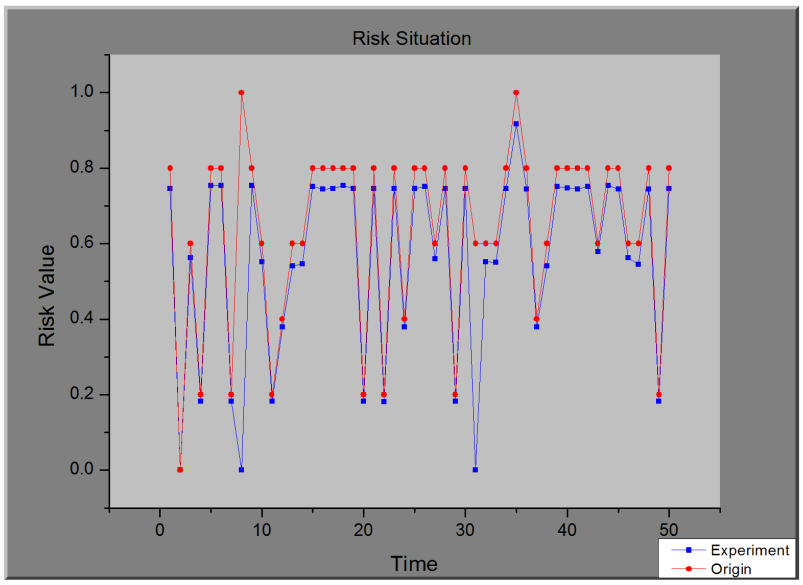


**Figure 5. Risk Conditions after Improvement**

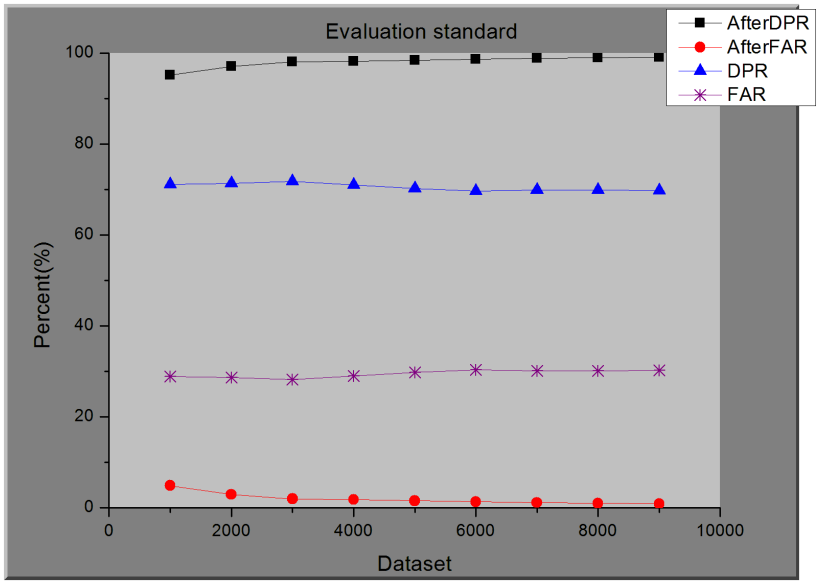


**Figure 6. Comparison of Model Effects of LSA-SAM Before Improvement and After Improvement**

It can be seen from Figures 4~9 that the rate of correct detection DPR of abnormal events of LSA-SAM model before improvement is only around 70%, while the rate of correct detection of model after improvement is around 98%; and judging from rate of false alarming FAR, it is only around 2% of LSA-SAM model after improvement while it is 29% of LSA-SAM model before improvement. Therefore, the improvement of LSA-SAM model performance through learning is outstanding. And DPR and FAR of LSA-

SAM model do not fluctuate significantly with the increase of data sets, which further indicates the correction of LSA-SAM model.

In the meantime, it can be observed from Figure 7 that Mark 5 and Mark 6 are multiple continuous points with the same risk value. Any behavior of frequent operation is considered as a kind of abnormal situation and could be under attack, since the experiment has assumed that there is only one event happening at the same time. In this case, it is necessary to give an alarm in time, which is handled by an administrator.

In case of the assumption of the experiment that there is only one event happening at the same time, it is an unreasonable situation in practical application. However, the assumption herein is only for the convenience of experimental simulation. It is only required to consider the minimum values of all safety events as the risk value of such occasion, in case of applying the LSA-SAM model into reality. Certainly it is necessary to take the following two situations into consideration: (1) it is required to lower the risk value within insecure scope and give an alarm, in case of multiple identical safety events happening at the same time; (2) it is required to give an alarm respectively for those safety events so that the administrator can confirm where the problem is, in case the risk values of multiple safety events are within insecure scope at the same time.

## 8. Parallelized Acceleration of MapReduce of LSA-GCC and LSA-SAM

The volume of log file data is enormous, usually surprisingly large. With the rapid development of information technology of computers, there are increasingly network equipments, safety equipments and application systems in enterprises such as host computer, server, firewall, switch, anti-virus wall and wireless router. Massive log information produced by these equipments have become an important part of fast-growing data in the Age of Big Data, and the work brought thereby such as log management and safety audit becomes increasingly complex. In the face of mass data, it is impossible to accomplish such work only depending on manual work in management. Therefore, how to efficiently collect, to process and to analyze mass data, and the pursuit of computing ability have motivated people to continuously develop new technologies to meet their demands. Distributive computing model has been evolved under such motivation, and high-performance computing [4], grid computing [5], pervasive computing [6] and cloud computing [7] emerge in succession, among which the parallel cloud computing method of MapReduce put forward by Google Company in the field of cloud computing is a new research hotspot in recent years [8]. It can carry out parallelization processing for complex and large-scale problems, as well as rapid distributive computing, especially suitable for application in the type of data mining and machine learning [9]. This paper will adopt MapReduce as the parallel acceleration method for verification.

## 9. Parallel Acceleration of LSA-GCC based on MapReduce Framework

During the process of modeling of LSA-GCC, the computation complexity and time cost of such model increase along with the increase of training sets. The MapReduce framework has many advantages such as greatly-simplified algorithm, and settlement of big data processing problem and high time cost problem. Therefore, this paper adopts the MapReduce framework to realize LSA-GCC algorithm: MR-LSA-GCC. The steps of algorithm MR-LSA-GCC are as follows:

Step 1: The training set D (after being processed by LSA) is divided into “m” parties, and each part of training set Di is distributed to MAP;

Step 2: MAP: The distributed training set Di is read, and the CSCP algorithm is applied to conduct clustering processing of Di to get the category set “mc” contained in training set Di; then the Rocchio algorithm is used to create the corresponding

prototype vectors for the categories in the category set "mc" to get the prototype vector set mgcc, of all categories in Di;

Step 3: REDUCE: The prototype vector set mgcc obtained from MAP is collected, and the CSPC algorithm is applied to merge prototype vectors in the prototype vector set mgcc so as to obtain the final prototype vector set mfinal-gcc.

The whole flow chart of MR-LSA-GCC algorithm is as showed in Figure 10.

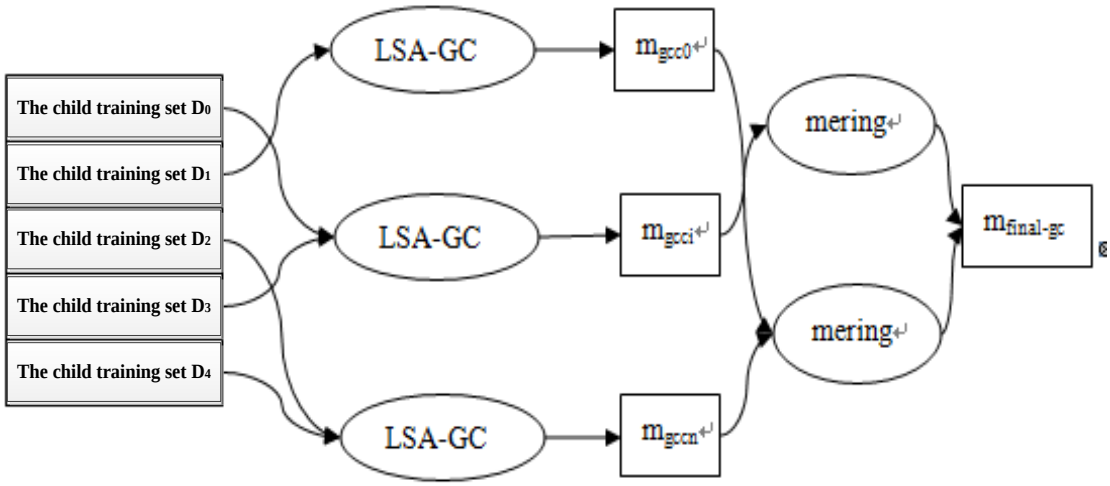


**Figure 7. Program Model of MR-LSA-GCC**

Algorithm 4.2 below is the implementation of pseudocode of MR-LSA-GCC.

**Algorithm 4.2 MR-LSA-GCC Pseudocode**

Algorithm 1 MR-LSA-GCC pseudocode
Function mapper (){
Runs LSA-GCC; //the same LSA-GCC program is operated on MAP
Ejects the result mgcc to Reducer;
}
Function reducer (mgcc){
foreach gcci, gccj in mgcc:
If (similarity (gcci, gccj)>r){
Merging (gcci, gccj)
}
}
Main (){
Set input method; //input training set Di after being processed by LSA
Set output method; //output the final prototype vector set mfinal-gcc
Set mapper; //setup numbers of Mapper
Set reducer; //setup numbers of Reduce
Submit job; //submit task

## 10. Rowing Acceleration of LSA-SAM Evaluation Model based on Map Reduce

In the LSA-SAM evaluation model, the similarity valuessim (p, ci0) is obtained through the computation of similarity between test sample T and established LSA-GCC model, and then the similarity values sim(p,ci0) is multiplied by the risk level "Si " of safety event to get the safety value of such sample; and the safety value is matched with the security level to obtain the security level of such sample. Combining with the

 

advantages of MapReduce framework, we can consider using MapReduce for the acceleration when the data set is relatively large, since each test sample is independent. Therefore, this paper puts forward this MR-LSA-SAM algorithm. Steps of algorithm of MR-LSA-SAM are as follows:

Step 1: Step 1: Test sample T is mapped to the latent semantic space created by training set D to obtain a vector representation set T' with semantic structure, and then the test set T'i is divided into "m" parts and the test set "T'i" is allocated to MAP;

Step 2: Step 2: MAP: The distributed test set T'I is read, and the LSA-SAM evaluation model is applied to obtain the security level of test sample;

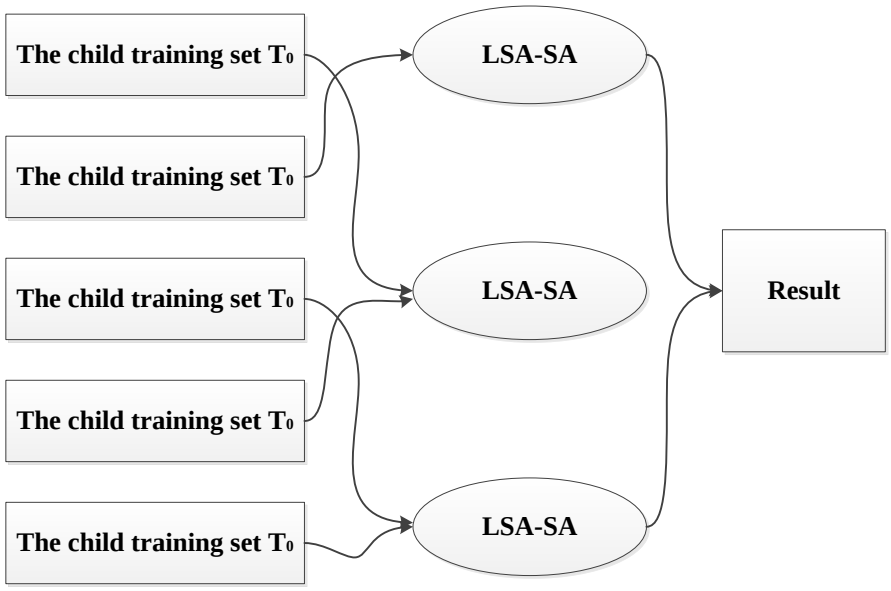


**Figure 8. Program Model of MR-LSA-SAM**

The whole flow chart of MR-LSA-SAM algorithm is as showed in Figure 11. The pseudocodes are as follows:

**Algorithm 4.3 Pseudocodes of Programmed Algorithm of MR-LSA-SAM**

```
Algorithm 2 pseudocodes of MR-LSA-SAM
Function mapper () {
Runs LSA-SAM;     //the same LSA-SAM program is operated in MAP.
}
Main () {
Set input method;   //input test set with method structure
Set output method; //output all security levels of corresponding test set T
Set mapper;   //set numbers of Mapper
Submit job;   //submit job
}
```

By the analysis of pseudocodes, users only need to implement two functions of Map and Reduce when using MapReduce framework and other steps are automatically completed by MapReduce framework, greatly simplifying the implementation of algorithm. Also, the MapReduce framework itself is designed for large-scale program calculation, so the algorithm itself can be easily expanded to multiple computational nodes. The algorithm implemented based on MapReduce framework can be expanded to large-scale clusters, without changing any code. This is highly beneficial for the solution to problems of variable scale, especially in the case of processing mass data [10,11,12].

The analysis of log record studied in this paper is the large-scale data. Therefore, the method described in this paper shows advantages in a higher degree after being strengthened by MapReduce framework.

1. Analysis of acceleration rate based on MapReduce framework

The most important objective is to reduce the time of solution to the algorithm in order to adopt parallel algorithm to solve optimization problem, and this paper is looking forward to predicting the enhanced acceleration rate of the two algorithms theoretically.

Algorithm acceleration rate S: for one problem, the time ratio spent between serial algorithm solution and parallel algorithm solution:

$$S = \frac{T_s}{T_p} \tag{8}$$

Wherein, Ts and Tp respectively refer to the time spent in the solution of problem between serial algorithm and parallel algorithm.

2. Theoretical acceleration rate of MR-LSA-GCC algorithm

According to the theoretical model of MR-LSA-GCC in Figure 10, there are five parts in Tp:

(1) Tfork referring to the time of copying the program from host nodes to other slave nodes;
(2) Tmap referring to the time of conducting LSA-GCC algorithm, namely the operation time of LSA-GCC;
(3) Tout referring to the time spent in sending computational results to REDUCE by MAP, namely the transmission time;
(4) Treducer referring to the operating time of conducting mering program;
(5) Tresult referring to the time of writing results of REDUCE into the hard disk.
(6) Therefore, Tp = Tfork+Tmap+Tout+Treducer+Tresult. We have:

$$S_{MR-LSA-GCC} = \frac{T_s}{T_p} = \frac{1}{\mathrm{Tfork} + \mathrm{Tmap} + \mathrm{Tout} + \mathrm{Treducer} + \mathrm{Tresult}} \tag{9}$$

From the model MR-LSA-GCC, we are aware that the copying and result output of the program are carried out once, therefore, the consumption of such time is relatively small, which can be ignored. So, we have:

$$S_{MR-LSA-GCC} = \frac{T_s}{T_p} \approx \frac{1}{\mathrm{Tfork} + \mathrm{Tout} + \mathrm{Treducer}} \tag{10}$$

It is assumed that in MR-LSA-GCC there is (are) m virtual machine (s) to conduct LSA-GCC algorithm as slave nodes, hence there is Ts=m*Tmap. Accordingly, there is:

$$S_{MR-LSA-GCC} = \frac{m}{1 + \frac{T_{out}}{T_{map}} + \frac{T_{reducer}}{T_{map}}} \tag{11}$$

It can be observed from the formula that the acceleration rate of MR-LSA-GCC depends on the time of conducting LSA-GCC algorithm. Therefore, MR-LSA-GCC is suitable for solving problems which are very complex with a demand for long time to get a solution.

 

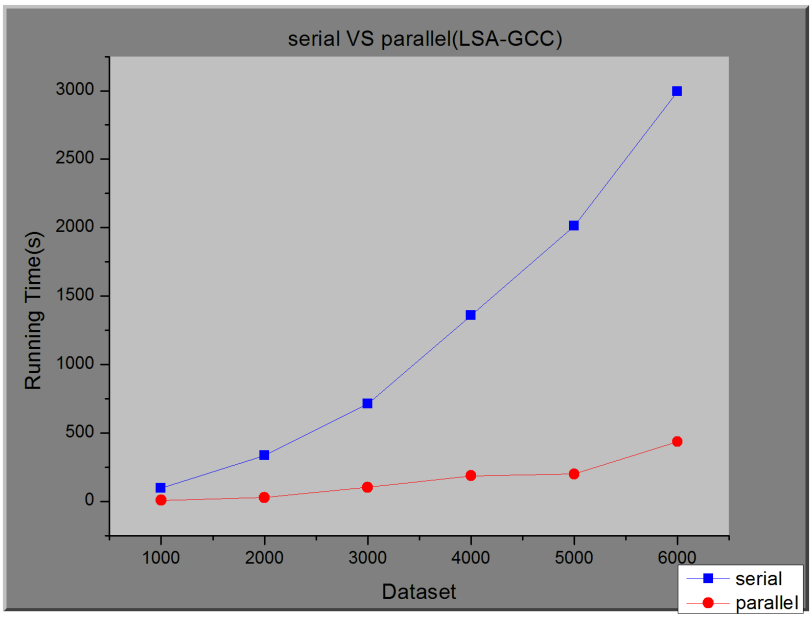


**Figure 10. Experimental Results of MR-LSA-GCC**

In Figure 13, it shows conditions of operation time of LSA-GCC model and MR-LSA-GCC model for different data sets. In the figure, the abscissa refers to the number of data sets and the ordinate refers to the operation time of LSA-GCC model and MR-LSA-GCC model for corresponding datasets under different conditions. It can be observed from Figure 10 that with the increase of data volume, the increasing rate of time of LSA-GCC model is relatively faster while the increasing rate of time of MR-LSA-GCC model is slower. The experiment shows the MR-LSA-GCC model has implemented the acceleration of up to 12.16 times for LSA-GCC model to some extent, proving the correctness and efficiency of MR-LSA-GCC model.

3. Theoretical acceleration rate of MR-LSA-SAM algorithm

According to the theoretical model of MR-LSA-SAM in Figures 4~12, Tp has three parts:

(1) Tfork referring to the time of copying the program from host nodes to other slave nodes;
(2) Tmap referring to the time of conducting LSA-SAM algorithm, namely the operation time of LSA-SAM;
(3) Tresult referring to the time of writing results of REDUCE into the hard disk.
(4) Therefore, Tp = Tfork+Tmap+Tresult. We have:

$$S_{MR-LSA-SAM} = \frac{T_s}{T_p} \quad S_{MR-LSA-SAM} = \frac{T_s}{T_p} \tag{12}$$

From the model MR-LSA-SAM, we are aware that the copying and result output of the program will only be carried out once, therefore, the consumption of such time is relatively small which can be ignored. Therefore we have:

$$S_{MR-LSA-SAM} = \frac{T_s}{T_p} \approx \frac{T_s}{T_{map}} \tag{13}$$

It is assumed that in MR-LSA-SAM there is (are) m PC (s) to conduct LSA-SAM algorithm as slave nodes, there is Ts=m*Tmap. Therefore, there is:

$$S_{MR-LSA-SAM} = \frac{T_s}{T_P} \approx m \quad (14)$$

It can be seen from the formula that the acceleration rate of MR-LSA-SAM is m. Therefore, MR-LSA-SAM is suitable for solving problems which can be divided into several independent sub-problems.

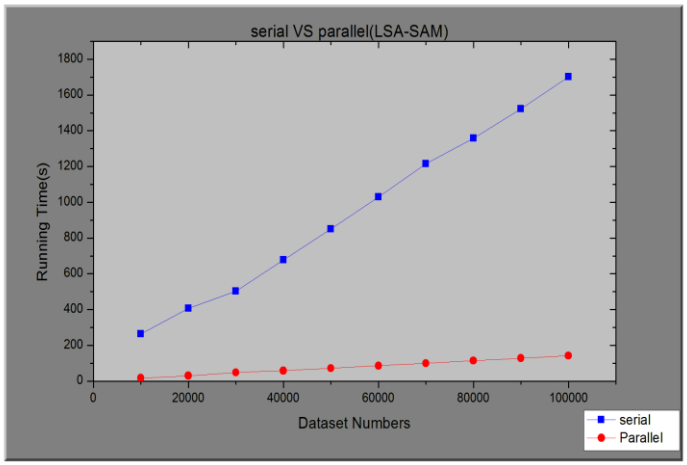


**Figure 11. Experimental Results of MR-LSA-SAM and LSA-SAM**

In Figure 14, it shows the conditions of operation time of LSA-SAM model and MR-LSA-SAM model. In the figure, the abscissa refers to the number of data sets and the ordinate refers to the operation time of LSA-GCC model and MR-LSA-SAM model of corresponding data sets under different conditions. It can be observed from Figure 14 that with the increase of data volume, the increasing rate of time of LSA-GCC model shows approximately linear growth, while the increasing rate of time of MR-LSA-SAM model also shows approximately linear growth but is is relatively slower. The experiment shows that the MR-LSA-GCC model has implemented the acceleration of up to 15.53 times for LSA-GCC model to some extent, proving the correctness and efficiency of MR-LSA-SAM model.

## 11. Summary

This paper introduces the latent semantic analysis (LSA), including principle of LSA, flow chart of solution and its application. For the massive log analysis generated from a cloud computing system [24, 25], this paper adopts the generalized clustering classification (LSA-GCC) based on latent semantic analysis to conduct data mining and to realize risk identification of event log on a semantic level. Unlike the event log type defined by the system itself, this method can discover potential risks in a mass of common logs and decide risk levels. This paper also compares the recognition effects of LSA-GCC by using the machine learning method and uses MapReduce to accelerate this recognition process. The experimental verification demonstrates that there is preferable improvement in the accuracy rate and recognition speed of the combination of the above-mentioned methods, and compared with ordinary event log statistics [25], the combination brings about a more accurate judgment of single-point risk level of virtual machine in a cloud computing system.

 

## Acknowledgements

The paper is supported by Nature Science Foundation of China (No. 61402386, No.61402389 and No. 61305061)

## References

[1] H. Zhi-lin, W. Chun-hong, "Application of Matrix Singular Value Decomposition（SVD）in Latent Semantic Information Retrieva"l. Modern Computer, **(2011)**, pp. 21-23.
[2] B. Baharudin, L. Lee, H., & Khan, K. (2010). A review of machine learning algorithms for text-documents classification. Journal of advances in information technology, vol. 1, no. 1, **(2011)**, pp. 4–20.
[3] Z. Shui-Geng, G. Ji-Hong, H. Yun-Fa, "Implied semantic indexing and its application in Chinese text processing research [J]", Small microcomputer system, vol. 22, no. 2, **(2001)**, pp. 239-243.
[4] S. Y. Jiang , Song, X. Y., & Wang, H. A clustering-based method for unsupervised intrusion detections. Pattern Recognition Letters, vol. 27, no. 5, **(2006)**, pp. 802–810.
[5] K. Dowd. High performance computing [M]. Sebastopol, CA, USA: O'Reilly & Associates, Inc. **(1993)**.
[6] I. Foster, C. Kesselman. The Grid Blueprint for a New Computing Infrastructure (Second Edition) [M]. Beijing: China Machine Press, **(2005)**.
[7] Xu Guang-hu, Shi Yuan-Chun, Xie Wei-Kai. Universal computer [J]. Journal of computers, , vol. 26, no. 9, **(2003)**, pp. 1042-1050.
[8] J. Dean, S. Ghemawat. MapReduce: Simplified data processing on large clusters [J]. Communications of the ACM, , vol. 51, no. 1, **(2008)**, pp. 107-113.
[9] Open MP [EB/OL]. http://openmp.org/wp/. pp. 2012-2-23.
[10] K.C. Sarma, H. Adeli. Bilevel parallel genetic algorithms for optimization of large steel structures [J]. Computer-Aided Civil and Infrastructure Engineering, vol. 16, no. 5, **(2001)**, pp. 295-304.
[11] R.Shonkwiler. E. Van Vleck, "Parallel speed-up of Monte Carlo methods for global optimization [J]", Journal of Complexity, no. 10, **(1994)**, pp. 64-64.
[12] D. ester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R, "Indexing by latent semantic analysis [J]", Journal of the American Society for Information Science, vol. 41, no. 6, **(1990)**, pp. 391-407.
[13] X. Yungan, Z. Qinghua, "Research on Tag Semantic Retrieval in Social Tagging System Based on LSA. Library and Information Service", vol. 55, no. 4, **(2011)**, pp. 11-14.
[14] H. Mingjie, H. Yuzhu, L. Jianhong, "Fault diagnosis method based on LSA and SVM". IEEE, **(2009)**, pp. 1-4.
[15] Z. Xiao-guo, H. Guang-jun, C. Li-hong, *et al*. "Web Services Filtrate Technologies Based on Latent Semantic Analysis", Computer Engineering, vol. 34, no. 15, **(2008)**, pp. 39-41.
[16] Z. Li-wei, D. Chan-lun, X. Zhi-wei,*et al*, "Study on the Applicat ion of Naive Bayesian Methods in Identifying Syndrome in TCM", Journal of Inner Mongolia university (natural science edition), vol. 38, no. 5, **(2007)**, pp. 568-571.
[17] P. Jun-feng, "Syndrome Element Differentiation Methodology based on Data Mining Technology", Hunan university of Chinese medicine Ph.D. Thesis, **(2007)**.
[18] J-t Sun, Q-y Zhang, Z-t Yuan, et al, "A Junk Mail Filtering Method Based on LSA and FSVM", Fifth International Conference on Fuzzy Systems and Knowledge Discovery, **(2008)**, pp. 111-115.
[19] N. Ishii, T. Murai,T. Yamada,*et al*. "Text Classification by Combining Grouping", LSA and kNN.IEEE,**(2006)**.
[20] C. Doukas, I. Maglogiannis, "Enabling Human Status Awareness in Assistive Environments based on Advanced Sound and Motion Data Classification", presented at The 1st ACM International Conference on PErvasive Technologies Related to Assistive Environments (PETRAE), Athens, Greece, July **(2008)**, pp. 16-19,
[21] C. Doukas, I. Maglogiannis, "Human Distress Sound Analysis and Characterization using Advanced Classification Techniques", to be presented at 5th Hellenic Conference on Artificial Intelligence, Syros, Greece, October **(2008)**, pp. 2-4 .
[22] World Wide Web Consortium (W3C),"Semantic Web Activity", http://www.w3.org/2001/sw/, accessed on June 4, **(2008)**.
[23] D. L. McGuiness, F. V. Hamelin, eds., "OWL Web Ontology Language Overview W3C Recommendation 10 February 2004", accessed on June 4, **(2008).**
[24] L. Hong, H. Shang-tang, "A fuzzy method to learn text classifier from labeled and unlabeled examples", Journal of Harbin Institute of Technology, vol. 11, no. 1, **(2004)**, pp. 98-102.
[25] M. Guang-fu, Z. Liang-kuan, Y. Gen-ting, *et al*, "Kernel Method for Building Fuzzy Classifiers", Intelligent Control and Automation, WCICA 2006. The Sixth World Congress on,, (01), **(2006)**, pp. 4307- 4311.

 