Generating Association Rules bases on The K-means Algorithm

Tran Cong Hung, Nguyen Van Hoa, *Member, IEEE*

***Abstract* -- Data mining is applied in many areas of life, such as statistics, artificial intelligence, databases, algorithms, parallel computing, knowledge gathering for expert systems, observation data, detection of association rules to support event predicting based router log and others. This paper is focused on analyzing the K-means algorithm, clustering on a data set of Cisco router log, and generating association rules from the clusters.**

***Index Terms* – Association rules, Clustering Algorithm, Data Mining, K-means, Router Logs.**

I. INTRODUCTION

ROUTER syslogs are messages that a router logs to describe many events to be observed. They are considered as one of the most valuable data sources for monitoring network and for troubleshooting network errors and performance anomalies. However, the router log messages are free-form texts with only a minimal structure, and their formats are different between vendors and router OSes [1].

In addition to aiming for tracking and debugging router software/hardware problems, the router syslogs are often too low-level from network service managements. Due to their large storage (e.g., millions per day in a large ISP network), the detailed router syslog messages are typically examined only when required by an on-going troubleshooting investigation or when given a narrow time range and a specific router under suspicion. The automatic systems based on router syslogs, on the other hand, tend to focus on a subset of the mission critical messages (e.g., relating to network fault) to avoid distracting by the diversity and complexity of syslog messages.

The algorithm K-means separates the message router log into K distinct clusters, after that the messages in the same cluster have similar content to support searching process, and create association rules.

The paper contains following sections: section I, introducing the purpose, significance of the router log and the necessity of the application of data mining in processing router log; section II, presenting format log on Cisco IOS XR system; section III, introducing a clustering algorithm that can be applied to process router log (K-means); section IV, presenting the results of generating association rules after executed the K-means algorithm on the data set of the Cisco Router log; section V: Conclusion and development trend.

II. ROUTER LOGS

This section provides an overview of the syntax and semantics of router log messages. Similar to logs on computer servers, router logs are the messages that routers create to record the hardware and software conditions observed by them, such as link and protocol-related state changes (e.g., down or up), alarm environmental measurements (e.g., high voltage or temperature), and warning messages (e.g., triggered when BGP neighbors send more routes than the router is configured to allow) [1][2][3][4][5].

While the log protocol (for transmitting log messages) is standardized, the log messages themselves are not. Table 2.1 shows a few examples of router log messages from two router vendors. They have only a minimal structure in a log message: (1) a timestamp shows when the message is created, (2) the identifier of the router that creates the message (called originating router), (3) message type, also known as the error code, shows the nature of the problem, and (4) detailed message information created by the router OS. In order to understand the correlation of log messages among routers, the clocks (for creating the timestamps) on these routers need to be synchronized regularly through the Network Time Protocol (NTP).

The detailed message information (aforementioned field (4)) is simply free-form texts "printf"-ed by router operating systems with detailed information such as location, state, or measurement readings of an alarming condition embedded in them. For example, under table 2.1 line 1, (Line protocol on Interface Serial13/0.10/20:0, changed state to down), the Serial13/0.10/20:0 part indicates the network interface at which the layer-2 line protocol (PPP) has been impacted and the down part indicates the state of the line protocol. The rest of message can be viewed as the sub type for this type of syslog message. This paper will concentrate on consideration about Cisco router log.

By default, the commonly format of syslog messages is created by syslog process on the Cisco IOS XR software [6][7] following:

node-id : *timestamp* : *process-name* [*pid*] : % *message -group -severity -message -code* : *message-text*

This is an example of syslog messages:

RP/0/RSP0/CPU0:Nov 28 23:56:53.826 : config[65710]: %SYS-5-CONFIG_I : Configured from console by console

The table 2.2 describes commonly format of syslog messages on the Cisco IOS XR software.

III. K-MEANS ALGORITHM

The K-means algorithm is a simple iterative clustering algorithm that partitions a given dataset into a user-specified number of clusters, "*k*". The algorithm is simple to implement and run, relatively fast, easy to adapt, and common in practice. It is historically one of the most important algorithms in data mining [8].

Manuscript received Dec 11, 2013 Accepted: Dec 27, 2013.

Tran Cong Hung, Ph.D. is with the Post and Telecommunications Institute of Technology, Vietnam (e-mail: conghung@ptithcm.edu.vn).

Nguyen Van Hoa, Eng. is with the Post and Telecommunications Institute of Technology, Vietnam (e-mail: vanhoanguyenkt@gmail.com).

TABLE 2.1
ROUTER LOG MESSAGES

Vendor	Message timestamp	Router	Message-type/error-code	Detailed message
V1	2010-01-10 00:00:15	R1	LINEPROTO-5-UPDOWN	Line protocol on Interface Serial13/0.10/20:0, changed state to down
V1	2010-01-10 00:00:15	R5	LINK-3-UPDOWN	Interface Serial2/0.10/2:0, changed state to down
V1	2010-01-10 00:00:15	R8	SYS-1-CPURISINGTHRESHOLD	Threshold: Total CPU Utilization(Total/Intr): 95%/1%, Top 3 processes(Pid/Util): 2/71%, 8/6%, 7/3%
V1	2010-01-10 00:00:26	R8	SYS-1-CPUFALLINGTHRESHOLD	Threshold: Total CPU Utilization (Total/Intr) 30%/1%
V2	2010-01-10 00:00:23	Ra	SNMP-WARNING-linkDown	Interface 0/0/1 is not operational
V2	2010-01-10 00:00:24	Rb	SVCMGR-MAJOR-sapPortStateChangeProcessed	The status of all affected SAPs on port 1/1/1 has been updated
V2	2010-01-10 00:00:26	Ra	SNMP-WARNING-linkup	Interface 0/1/0 is operational

TABLE 2.2
GENERAL SYSLOG MESSAGE FORMAT ON CISCO ASR 9000 SERIES

Field	Description
node-id	Node from which the syslog message originated.
timestamp	Time stamp in the form *month day HH:MM:SS*, indicating when the message was generated. **Note:** The time-stamp format can be modified using the **service timestamps** command.
process-name	Process that generated the syslog message.
[*pid*]	Process ID *(pid)* of the process that generated the syslog message.
%message-group-severity-message-code	Message group name, severity *(table 2.3)*, and message code associated with the syslog message.
message-text	Text string describing the syslog message.

TABLE 2.3
ERROR MESSAGE SEVERITY LEVELS

Severity Keyword	Level	Description	Syslog Definition
emergencies	0	System unstable	LOG_EMERG
alerts	1	Immediate action needed	LOG_ALERT
critical	2	Critical conditions	LOG_CRIT
errors	3	Error conditions	LOG_ERR
warnings	4	Warning conditions	LOG_WARNING
notifications	5	Normal but significant condition	LOG_NOTICE
informational	6	Informational messages only	LOG_INFO
debugging	7	Debugging messages	LOG_DEBUG

Historically, K-means in its essential form has been discovered by several researchers across different disciplines, most notably by Lloyd (1957, 1982), Forgey (1965), Friedman and Rubin (1967) , and McQueen (1967) .

The k-means algorithm applies to objects that are represented by points in a d-dimensional vector space. Thus, it clusters a set of d-dimensional vectors, $D = \{x_i \mid i = 1, \ldots, N\}$, where $x_i \in R^d$ denotes the ith object or "data point". K-means is a clustering algorithm that partitions D into k clusters of points. That is, the K-means algorithm clusters all of the data points in D such that each point x_i falls in one and only one of the k partitions. One can keep track of which point is in which cluster by assigning each point a cluster ID. Points with the same cluster ID are in the same cluster, while points with different cluster IDs are in different clusters. One can denote this with a cluster membership vector m of length N, where m_i is the cluster ID of m_i.

The value of k is an input to the base algorithm. Typically, the value for k is based on criteria such as prior knowledge of how many clusters actually appear in D, how many clusters are desired for the current application, or the types of clusters found by exploring/experimenting with different values of k.

In K-means, each of the k clusters is represented by a single point in R^d. Let us denote this set of cluster representatives as the set $C = \{c_j \mid j = 1, \ldots, k\}$.

In clustering algorithms, points are grouped by some notion of "closeness" or "similarity." In K-means, the default measure of closeness is the Euclidean distance. In particular, one can readily show that k-means attempts to minimize the following nonnegative cost function:

$$Cost = \sqrt{\sum_{i=1}^{N} \arg\min_j (x_i - c_j)^2} \qquad (3.1)$$

In other words, K-means attempts to minimize the total squared Euclidean distance between each point $\boldsymbol{x_i}$ and its closest cluster representative $\boldsymbol{c_j}$.

The K-means algorithm, depicted in Algorithm 3.1, clusters D in an iterative fashion, alternating between two steps: (1) reassigning the cluster ID of all points in D and (2) updating the cluster representatives based on the data points in each cluster. The algorithm works as follows. First, the cluster representatives are initialized by picking k points in R^d. Techniques for selecting these initial seeds include sampling at random from the dataset, setting them as the solution of clustering a small subset of the data, or perturbing the global mean of the data k times. In Algorithm 3.1, we initialize by randomly picking k points. The algorithm then iterates between two steps until convergence.

Step 1: Data assignment. Each data point is assigned to its closest centroid, with ties broken arbitrarily. This results in a partitioning of the data.

Step 2: Relocation of "means". Each cluster representative is relocated to the center (i.e., arithmetic mean) of all data points assigned to it. The rationale of this step is based on the observation that, given a set of points, the single best representative for this set (in the sense of minimizing the sum of the squared Euclidean distances between each point and the representative) is nothing but the mean of the data points. This is also why the cluster representative is often interchangeably referred to as the cluster mean or cluster centroid, and where the algorithm gets its name from.

The algorithm converges when the assignments (and hence the cj values) no longer change. One can show that the K-means objective function defined in *Equation- 3.*1 will decrease whenever there is a change in the assignment or the relocation steps, and convergence is guaranteed in a finite number of iterations.

Noted that each iteration needs $N \times k$ comparisons, which determines the time complexity of one iteration. The number of iterations required for convergence varies and may depend on N, but as a first cut, K-means can be considered linear in the dataset size. Moreover, since the comparison operation is linear in d, the algorithm is also linear in the dimensionality of the data.

Limitations: The greedy-descent nature of k-means on a nonconvex cost implies that the convergence is only to a local optimum, and indeed the algorithm is typically quite sensitive to the initial centroid locations. In other words, initializing the set of cluster representatives C differently can lead to very different clusters, even on the same dataset D. A poor initialization can lead to very poor clusters.

As mentioned, choosing the optimal value of k may be difficult. If one has knowledge about the dataset, such as the number of partitions that naturally comprise the dataset, then that knowledge can be used to choose k. Otherwise, one must use some other criteria to choose k, thus solving the model selection problem. One naive solution is to attempt several different values of k and choose the clustering which minimizes the K-means objective function (Equation 3.1). Unfortunately, the value of the objective function is not as informative as one would hope in this case. For example, the cost of the optimal solution decreases by increasing k till it hits zero when the number of clusters equals the number of distinct data points. This makes more difficult to use the objective function to (a) directly compare solutions with different numbers of clusters and (b) find the optimum value of k. Thus, if the desired k is not known in advance, one will typically run K-means with different values of k, and then use some other, more suitable criterion to select one of the results.

ALGORITHM 3.1

THE K-MEANS ALGORITHM

Input: Dataset D, number clusters k

Output: Set of cluster representatives C, cluster

membership vector m

/* Initialize cluster representatives C */

Randomly choose k data points from D

Use these k points as initial set of cluster representatives C

repeat

/* Data Assignment */

Reassign points in D to closest cluster mean

Update m such that m_i is cluster ID of ith point in D

/* Relocation of means */

Update C such that c_j is mean of points in jth cluster

until convergence of objective function

$$Cost = \sqrt{\sum_{i=1}^{N} \arg\min_j (x_i - c_j)^2}$$

In order to be not lost generally inside generating association rules after clustering, the K-means algorithm executed many times with some different values of K.

IV. GENERATING ASSOCIATION RULES

The below results are enforced with language R [9] and Java. The application operates on a Laptop Lenovo B460, configuration: *CPU Intel (R) Core(TM) i3 M 370 @ 2.40GHz*, *RAM 3.2GB, Windows 7 Ultimate 32bit Service Pack 1.*

Performed the same input data has 971 records, the number of iterations of algorithm is over 2 times, with values k = 2, 3, 4, 5, 6. The logs edited by eliminating fields: *node-id, timestamp, process-name, pid* because the logs generated in the same Router, arranged in order to series time.

- k = 2 (2 clusters);

The first execution, the number of members in each cluster by: 829 and 142, the time of clustering (*cost*) is 3,945 ms. The number of association rules generated is 15 rules in table 4.1.

The second execution, the number of members in each cluster by: 243 and 728, the time of clustering is 4,100 ms. The number of association rules addition generated is 6 rules in table 4.2, so the total number of rules generated is 21 rules.

From the third execution, it is not generated one more rules.

- k = 3;

The first execution, the number of members in each cluster by: 728, 101, and 142, the time of clustering is 4,135 ms. The number of association rules generated is 15 rules in table 4.1.

TABLE 4.1
THE SET OF ASSOCIATION RULES – PART 1

Rule: 1 – ConfXY ***(The confidence of a rule: X => Y)*****: 0.01997802417340925 %**
(X) %MGBL-SYS-5-CONFIG_I : Configured from console by vanhoa@backbone on vty1 (10.74.225.49)
(Y) %L2-PLIM_ETHER-2-RX_RF : Interface TenGigE0/3/0/7, Detected Remote Fault

Rule: 2 - ConfXY: 0.01997802417340925 %
(X) %MGBL-SYS-5-CONFIG_I : Configured from console by vanhoa@backbone on vty1 (10.74.225.49)
(Y) %L2-PLIM_ETHER-2-RX_LOSS : Interface TenGigE0/3/0/7, Detected Rx Loss of Signal

Rule: 3 - ConfXY: 0.01997802417340925 %
(X) %MGBL-SYS-5-CONFIG_I : Configured from console by vanhoa@backbone on vty1 (10.74.225.49)
(Y) %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface TenGigE0/3/0/7, changed state to Down

Rule: 4 - ConfXY: 0.01997802417340925 %
(X) %MGBL-SYS-5-CONFIG_I : Configured from console by vanhoa@backbone on vty1 (10.74.225.49)
(Y) %PKT_INFRA-LINK-3-UPDOWN : Interface TenGigE0/3/0/7, changed state to Down

Rule: 5 - ConfXY: 0.01997802417340925 %
(X) %MGBL-SYS-5-CONFIG_I : Configured from console by vanhoa@backbone on vty1 (10.74.225.49)
(Y) %ROUTING-RIB-5-TABLE_NH_DAMPED : Nexthops in Vrf: default Tbl: default (0xe0000000) Safi: Unicast are getting damped

Rule: 6 - ConfXY: 0.025396825396825397 %
(X) %L2-PLIM_ETHER-2-RX_RF : Interface TenGigE0/3/0/7, Detected Remote Fault
(Y) %L2-PLIM_ETHER-2-RX_LOSS : Interface TenGigE0/3/0/7, Detected Rx Loss of Signal

Rule: 7 - ConfXY: 0.025806451612903226 %
(X) %L2-PLIM_ETHER-2-RX_RF : Interface TenGigE0/3/0/7, Detected Remote Fault
(Y) %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface TenGigE0/3/0/7, changed state to Down

Rule: 8 - ConfXY: 0.025806451612903226 %
(X) %L2-PLIM_ETHER-2-RX_RF : Interface TenGigE0/3/0/7, Detected Remote Fault
(Y) %PKT_INFRA-LINK-3-UPDOWN : Interface TenGigE0/3/0/7, changed state to Down

Rule: 9 - ConfXY: 0.02848597065945022 %
(X) %L2-PLIM_ETHER-2-RX_RF : Interface TenGigE0/3/0/7, Detected Remote Fault
(Y) %ROUTING-RIB-5-TABLE_NH_DAMPED : Nexthops in Vrf: default Tbl: default (0xe0000000) Safi: Unicast are getting damped

Rule: 10 - ConfXY: 0.03807348181991243 %
(X) %L2-PLIM_ETHER-2-RX_LOSS : Interface TenGigE0/3/0/7, Detected Rx Loss of Signal
(Y) %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface TenGigE0/3/0/7, changed state to Down

Rule: 11 - ConfXY: 0.03807348181991243 %
(X) %L2-PLIM_ETHER-2-RX_LOSS : Interface TenGigE0/3/0/7, Detected Rx Loss of Signal
(Y) %PKT_INFRA-LINK-3-UPDOWN : Interface TenGigE0/3/0/7, changed state to Down

Rule: 12 - ConfXY: 0.042078687144961074 %
(X) %L2-PLIM_ETHER-2-RX_LOSS : Interface TenGigE0/3/0/7, Detected Rx Loss of Signal
(Y) %ROUTING-RIB-5-TABLE_NH_DAMPED : Nexthops in Vrf: default Tbl: default (0xe0000000) Safi: Unicast are getting damped

Rule: 13 - ConfXY: 0.047778308647873864 %
(X) %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface TenGigE0/3/0/7, changed state to Down
(Y) %PKT_INFRA-LINK-3-UPDOWN : Interface TenGigE0/3/0/7, changed state to Down

Rule: 14 - ConfXY: 0.05602240896358543 %
(X) %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface TenGigE0/3/0/7, changed state to Down
(Y) %ROUTING-RIB-5-TABLE_NH_DAMPED : Nexthops in Vrf: default Tbl: default (0xe0000000) Safi: Unicast are getting damped

Rule: 15 - ConfXY: 0.1508295625942685 %
(X) %PKT_INFRA-LINK-3-UPDOWN : Interface TenGigE0/3/0/7, changed state to Down
(Y) %ROUTING-RIB-5-TABLE_NH_DAMPED : Nexthops in Vrf: default Tbl: default (0xe0000000) Safi: Unicast are getting damped

TABLE 4.2
THE SET OF ASSOCIATION RULES – PART 2

Rule: 16 - ConfXY: 2.0048557606522997E-4 %
(X) %MGBL-SYS-5-CONFIG_I : Configured from console by vanhoa@backbone on vty1 (10.74.225.49)
(Y) %ROUTING-OSPF-5-ADJCHG : Process 1, Nbr 27.68.252.1 on Bundle-Ether2 in area 0 from LOADING to FULL, Loading Done,vrf default vrfid 0x60000000

Rule: 17 - ConfXY: 1.9574547216254706E-4 %
(X) %L2-PLIM_ETHER-2-RX_RF : Interface TenGigE0/3/0/7, Detected Remote Fault
(Y) %ROUTING-OSPF-5-ADJCHG : Process 1, Nbr 27.68.252.1 on Bundle-Ether2 in area 0 from LOADING to FULL, Loading Done,vrf default vrfid 0x60000000

Rule: 18 - ConfXY: 1.8959538448995997E-4 %
(X) %L2-PLIM_ETHER-2-RX_LOSS : Interface TenGigE0/3/0/7, Detected Rx Loss of Signal
(Y) %ROUTING-OSPF-5-ADJCHG : Process 1, Nbr 27.68.252.1 on Bundle-Ether2 in area 0 from LOADING to FULL, Loading Done,vrf default vrfid 0x60000000

Rule: 19 - ConfXY: 1.862466184598336E-4 %
(X) %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface TenGigE0/3/0/7, changed state to Down
(Y) %ROUTING-OSPF-5-ADJCHG : Process 1, Nbr 27.68.252.1 on Bundle-Ether2 in area 0 from LOADING to FULL, Loading Done,vrf default vrfid 0x60000000

Rule: 20 - ConfXY: 1.7813388721096664E-4 %
(X) %PKT_INFRA-LINK-3-UPDOWN : Interface TenGigE0/3/0/7, changed state to Down
(Y) %ROUTING-OSPF-5-ADJCHG : Process 1, Nbr 27.68.252.1 on Bundle-Ether2 in area 0 from LOADING to FULL, Loading Done,vrf default vrfid 0x60000000

Rule: 21 - ConfXY: 1.6899156816570636E-4 %
(X) %ROUTING-OSPF-5-ADJCHG : Process 1, Nbr 27.68.252.1 on Bundle-Ether2 in area 0 from LOADING to FULL, Loading Done,vrf default vrfid 0x60000000
(Y) %ROUTING-RIB-5-TABLE_NH_DAMPED : Nexthops in Vrf: default Tbl: default (0xe0000000) Safi: Unicast are getting damped

From the second execution, it is not generated one more rules.

- $k = 4$;

The first execution, the number of members in each cluster by: 566, 101, 162, and 142, the time of clustering is 4,203 ms. The number of association rules generated is 15 rules in table 4.1.

From the second execution, it is not generated one more rules.

- $k = 5$;

The first execution, the number of members in each cluster by: 142, 566, 101, 20, and 142, the time of clustering is 4,385 ms. The number of association rules generated is 15 rules in table 4.1.

From the second execution, it is not generated one more rules.

- $k = 6$;

The first execution, the number of members in each cluster by: 38, 566, 104, 74, 88, and 101, the time of clustering is 4,700 ms. The number of association rules generated is 7 rules.

The second execution, the number of members in each cluster by: 566, 6, 142, 98, 17, and 142, the time of clustering is 4,318 ms. The number of association rules addition generated is 8 rules, so the total number of rules generated is 15 rules in table 4.1.

From the third execution, it is not generated one more rules.

With values K ≥ 7, the number of association rules generated is not greater than 15 rules. The association rules match with the 15 rules in table 4.1.

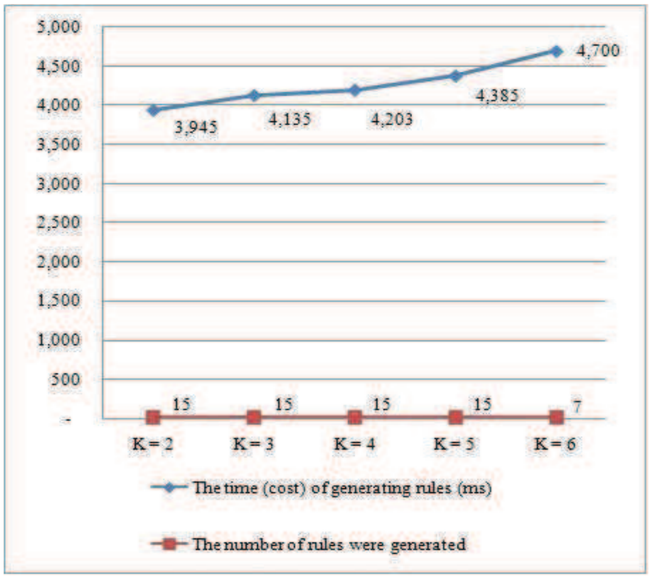

Fig. 4.1. The time of generating rules at the first time

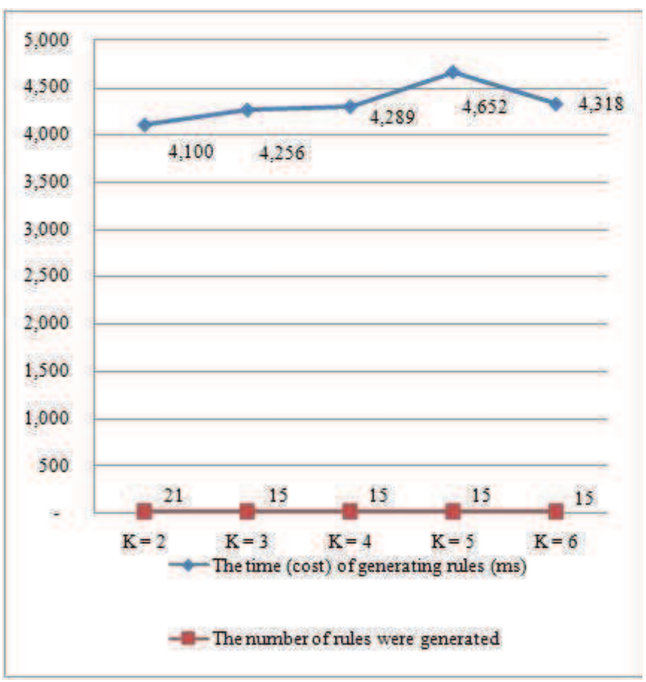

Fig. 4.2. The time of generating rules at the second time

V - CONCLUSION AND DEVELOPMENT TREND

Through the results above, it can be concluded that each execution picks randomly K initial values inside the K-means algorithm, which leads the number of members in each group, and the time of execution to be often different on the same input data set (Fig. 4.1, Fig. 4.2). Moreover, in case the k initial values are selected randomly high suitability that the convergence time of the K-means algorithm is quite good.

With the way of generating as mentioned above, it helps to reduce the time of generating association rules. So association rules only generated between the logs in the same cluster. The generality of the set of association rules is inverse proportion to the number of clusters, and direct proportion to the number of logs in a cluster.

No limit on the time correlation between the logs, it supports to propose many association rules for reference throughout the entire logs generated.

In the future, development trend will be researched to predict upcoming events in the network. This prediction bases on the set of association rules generated. In addition, the development trend also compares with other algorithms, and expands the application scope for the different Router venders.

REFERENCES

[1] Qiu, T. Ge, Z. Pei, D. Wang, J, and Xu, J. (2010), *What Happened in my Network? Mining Network Events from Router Syslogs,* IMC'10, Australia.

[2] Qiu, T. (2011), *Understanding a large-scale IPTV Network via System Logs,* Georgia Institute of Technology, USA.

[3] M. Rahmati, and S.M. Mirzababaei, *Data Mining on the Router Logs for Statistical Application Classification*, World Academy of Science, Engineering and Technology 12 2005.

[4] Karel Slavicek, Jaroslav Ledvinka, Michal Javornik, and Otto Dostal. (2008), *Mathematical Processing of Syslog Messages from Routers and Switches*, ICIAFS08, IEEE.

[5] Meiyappan Nagappan, and Mladen A. Vouk. (2010), *Abstracting Log Lines to Log Event Types for Mining Software System Logs*, MSR 2010, IEEE.

[6] Cisco Website, *Configuring System Message Logging*, accessed date: 12/01/2013. http://www.cisco.com/en/US/docs/switches/lan/catalyst2950/software/release/12.1_9_ea1/configuration/guide/swlog.html#wp1030180

[7] Cisco Systems Inc (2011), *Cisco IOS XR System Error Message Reference Guide,* Release 4.1, USA, access date: 12/01/2013. http://www.cisco.com/en/US/docs/ios_xr_sw/iosxr_r4.1/error/messages/em41sems.html

[8] Wu, X. Kumar, V. (2009), *The Top Ten Algorithms in Data Mining*, Taylor & Francis Group, LLC, USA.

[9] *The R Project for Statistical Computing*, accessed date: 12/01/2013. http://cran.r-project.org

AUTHORS

TRAN CONG HUNG was born in VietNam in 1961.

He received the B.E in electronic and Telecommunication engineering with first class honors from HOCHIMINH university of technology in VietNam, 1987.

He received the B.E in informatics and computer engineering from HOCHIMINH university of technology in VietNam, 1995.

He received the master of engineering degree in telecommunications engineering course from postgraduate department HaNoi University of technology in VietNam, 1998.
He received Ph.D at HaNoi University of technology in VietNam, 2004.
His main research areas are B – ISDN performance parameters and measuring methods, QoS in high speed networks, MPLS.
He is, currently, Associate Professor PhD. of Faculty of Information Technology II, Posts and Telecoms Institute of Technology in HOCHIMINH, VietNam.

NGUYEN VAN HOA was born in Vietnam in 1986.
Obtained B.E in Information Technology from Post and Telecommunication Institute of Technology (PTIT), Vietnam, 2010.
He received Master in PTIT, 2013, major in Information System.
He is, currently, Sales engineering at VDC Company, 2010.