3. Jin Hyun Son, Myoung Ho Kim. An analysis of the optimal number of servers in distributed client/server environments[J]. Decision Support Systems,2002, 363.
4. Shiyuan Wang,Divyakant Agrawal,Amr El Abbadi. Towards practical private processing of database queries over public data[J]. Distributed and Parallel Databases, 2014, 321.
5. Suchendra M. Bhandarkar, Lakshmish Ramaswamy, Hari K. Devulapally. Collaborative caching for efficient dissemination of personalized video streams in resource constrained environments[J]. Multimedia Systems, 2014, 201.
6. Jeff Offutt, Vasileios Papadimitriou, Upsorn Praphamontripong. A case study on bypass testing of web applications[J]. Empirical Software Engineering, 2014, 191.
7. Heyoung Lee, Heejune Ahn, Samwook Choi, Wanbok Choi. The SAMS: Smartphone Addiction Management System and Verification[J]. Journal of Medical Systems, 2014, 381.
8. Zhi-Wei Xu. Cloud-Sea Computing Systems: Towards Thousand-Fold Improvement in Performance per Watt for the Coming Zettabyte Era[J]. Journal of Computer Science and Technology, 2014, 292.
9. G. L. Laing, J. L. Bruce, D. L. Skinner, N. L. Allorto, D. L. Clarke,C. Aldous. Development, Implementation, and Evaluation of a Hybrid Electronic Medical Record System Specifically Designed for a Developing World Surgical Service[J]. World Journal of Surgery, 2014, 386.
10. Bulent Tugrul, Huseyin Polat. Privacy-Preserving Inverse Distance Weighted Interpolation[J]. Arabian Journal for Science and Engineering, 2014, 394.
11. Manfred F. Buchroithner, Guido Ehlert, Bernd Hetze, Horst Kohlschmidt, Nikolas Prechtel. Satellite-Based Technologies in Use for Extreme Nocturnal Mountain Rescue Operations: a Synergetic Approach Applying Geophysical Principles[J]. Pure and Applied Geophysics, 2014, 1716.

Research on the Improvement of Proactive Workload Management Based on the Multi-Objective Genetic Algorithms

Guangqing Shan*

Chongqing City Management College, Chongqing, 401331, China

Abstract

In this paper, the author researched the improvement of proactive workload management based on the multi-objective genetic algorithms. The authors were irresolute between building a new simulation environment or utilizing an existent one. The answer for this question was to find a suitable simulation which provide most of the

main component of Cloud environments; accordingly, after a deep investigation we clarify that CloudSim simulator package satisfy the elemental requirements to start tasting the PWM model.
Keywords: PROACTIVE WORKLOAD MANAGEMENT, MULTI-OBJECTIVE GENETIC ALGORITHMS, MULTI-OBJECTIVE OPTIMIZATION PROBLEMS.

1. Introduction

Even there are a lot of rough challenges (i.e. security, and reliability) which threat the migration to the Cloud, the utility solution of Cloud Computing becomes a fact in the IT industry and can not be ignored; at the same time, dozens of researches are running now to facilitate these challenges. Cloud computing is information processing model in which centrally administered computing capabilities are delivered as services, on an as-needed basis, across the network to a variety of user-facing "devices" [1]. These services are introduced in the form of Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service [2] and recently some researches talk about Network as a Service (NaaS) [3-4].

For many years the cloud technology tried to show up through various forms, but it did not take the current shape only after business use of the Internet has grown, the cloud has moved from a throwaway symbol in the architect's diagram to something more substantial and specific. From this time the cloud technology moved to become an auxiliary computing supplied through Web services, where the operation of standard business applications can be handled with.

Businesses built around Web services, such as Google, Amazon.com, and eBay, produced a new type of data center that was more standardized, more automated, and built from mass-produced personal computer parts. Access to these data centers was kept under wraps for several years as their builders sought to maintain a competitive advantage [5-6]. As the notion caught on that it was possible to provide more and more powerful services over the Internet, cloud computing came to mean an interaction between an end user, whether a consumer or a business computing specialist, and one of these services "in the cloud."

In fact, Cloud Computing overcome an existing technology called Grid Computing which offers access to many heterogeneous resources [7]; however, a user typically needs a very specific environment that is customized to support specific requirements or legacy applications while resources' providers clearly cannot support the diversity of all the required environments and users are often unable to use what is available [8].

Virtualization in its simplest form is the process of taking a physical machine and subdividing it through software into the equivalent of several discrete machines [9-10]. Although they share the hardware resources of a single server, they work independently from each other without conflicts (see Figure 1).

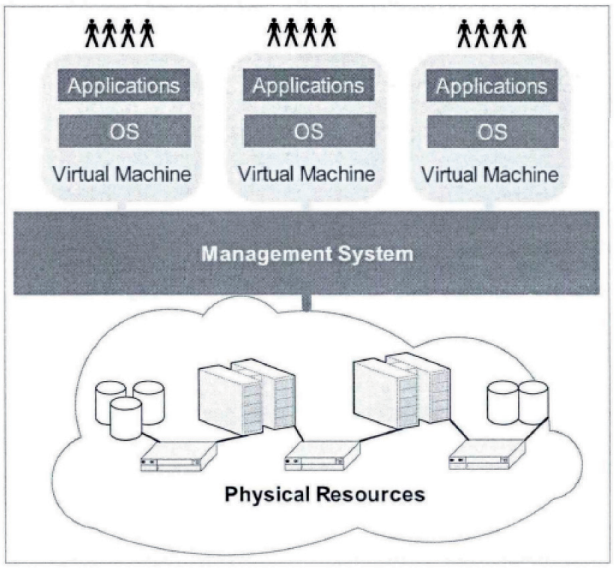

Figure 1. Title Virtualization common basic structure

Guest OS is the easiest virtualization concept to understand. In this scenario a virtualization application executes in the same way similar to other application such as a word processor. This virtualization application is able to host multiple standard unmodified operating systems such as Windows, Linux, UNIX or Mac OS X.

The simplicity of this concept doesn't come with any cost, as the running OS Guests usually show limited performance. Some examples of Guest OS include Virtual Box and VMware server. The figure 2 shows the guest operating system virtualization.

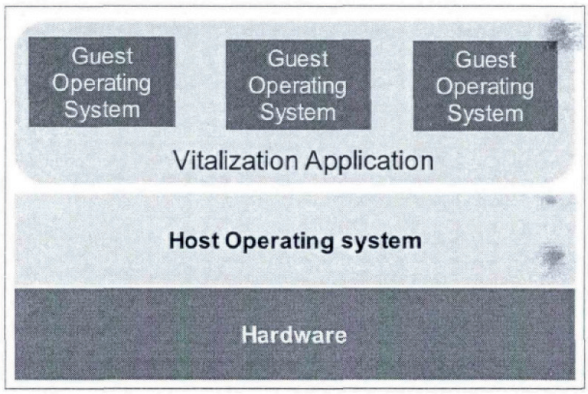

Figure 2. Guest Operating System Virtualization

2. The PWMM Evaluation

Many commercial and free systems with different virtualization systems where developed such as Open VZ, z/VM, VMware, Virtual Box, Hyper-V. These systems developed with different level of virtualization vary in their goals in two directions, one of them concerned with increasing the degree of isolation between the running MS at the expense of performance such as Full virtualization and Para virtualization techniques, where the opposite direction concerned more with the performance such as OS level technique. The common rule among all these systems is to successfully manage the resources between VMs.

It is worth to mention that, practically the word "Hypervisor" is used as a common term for all virtualization systems regardless of the technique used. We list some common Systems/Hypervisors to depict the differences between the features and the techniques used.

In general, Virtualization technologies are used to enhance the hardware load on server systems. A few commercial solutions available for allocating virtual machines during their operation time to optimize the actual server workload (e.g. VMware DRS, Virtual Iron Live Capacity). This operation represented in C1oudSim as following, Datacenter Controller uses a VM Load Balancer to determine which VM should be assigned to the next request for processing. Most common VM load balancer is the active monitoring load balancing algorithms as mentioned before in Introduction.

Active VM load balancer maintains information about each VMs and the number of requests currently allocated to which VM. When a request to allocate a new VM arrives, it identifies the least loaded VM. If there are more than one, the first identified is selected. Active VM load balancer returns the VM id to the Data Center Controller. The data Center Controller sends the request to the VM identified by that id.

Datacenter Controller notifies the Active VM load balancer of the new allocation. This process becomes complex with frequent workload changes, so when the Active VM load balancer can't take the right decision based on the available monitored information to dispatch the new coming requests to the proper VMs, peak loads occur.

In this situation the Datacenter Controller tried to handle this situation by selecting targeted VMNMs and scale the available resources to absorb the coming workloads. Scaling the available resources can be in the form of increasing shared resources (CPU, Memory) slices in the same machine or by migrating to another machine with more power. Hereafter we will call this method Active Monitoring Scaling to distinguish from scaling based on our Proactive Model in the following Section.

The DVFS technique can achieve the vertical scaling at some limits; however, this requires using special non cheap server equipment and technologies which contradicts with the trend of using commodity for building the Cloud. Thus, application behaviors prediction technique can be a better alternative.

On the other hand, as mentioned in the introduction many monitoring tools are available for different hypervisors. (i.e. Oprofile, XenOprof and Xenmon are available for Xen hypervisor, and a collection of monitoring tools for OpenVZ such as Beanmonitor, Yyabeda and Munin); however, using these tools to adapt to dynamic changes statically (i.e Xen) or even dynamically (i.e z/VM) lead to infrequent peak loads which drive to low average utilization of resources. Some hypervisors tried to lighten the side effects of this behavior by different scheduling techniques for example the Xen team designed CPU scheduler termed the credit scheduler to minimize wasted CPU time. This makes it a "work-conserving" scheduler, in that it tries to ensure that the CPU will always be working, whenever there is work for it to do. As a consequence, if there is more real CPU available than the domUs are demanding, all domUs get all the CPU they want. When there is contention that is, when the domUs in aggregate want more CPU than actually exists then the scheduler arbitrates fairly between the domains that want CPU. However this can't solve the problem as resources has direct impact over each other, for example if a VM have available CPU power and lack the memory still the problem exists.

3. The SBG-MOGA Algorithm

In SBG-MOGA, each generation of evolution is considered as a game and each objective to be optimized is considered as a player. A player is clever to know to select an algorithm contains a sequence strat-

egy to get the biggest income in the game. The terms of rounds, in each round players play with each other, which will provide the population with tensile force to pull them to move toward the true Pareto front; In addition elitism mechanism adopted.

(1) A current position in an N-dimensional search space $X_i^k = \left(X_1^k, ..., X_n^k, ..., X_N^k\right)$ where $X_n^k \in [l_n, u_n]$, $1 \le n \le N$, l_n, u_n is lower and upper bound for the nth dimension, respectively.

(2) A current velocity V_i^k, $V_i^k = \left(V_1^k, ..., V_n^k, ..., V_N^k\right)$ which is bounded by a maximum velocity $V_{\max}^k = \left(V_{\max,1}^k, ..., V_{\max,n}^k, ..., V_{\max,N}^k\right)$, and a minimum $V_{\min}^k = \left(V_{\min,1}^k, ..., V_{\min,n}^k, ..., V_{\min,N}^k\right)$.

In each iteration of SBG-MOGA, the algorithm is updated by the following equations:

$$V_i^{k+1} = \omega^{V_i^k} + C_1 r_1 \left(P_i^k - X_i^k\right) + C_2 r_2 \left(P_g^k - X_i^k\right) \quad (1)$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \quad (2)$$

where P_i is the best previous position of the ith particle (also known as pbest). Therefore, the inertia weight factor cc can be defined as follows.

$$\omega = \omega_{\max} - iter \times \frac{\omega_{\max} | \omega_{\min}}{iter_{\max}} \quad (3)$$

where $\omega_{\max}$ and $\omega_{\min}$ is the maximum and minimum.

The variance of the population's fitness (σ^2) is introduced and defined as follows:

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} \left[\frac{f(X_i - favg)}{f_d} \right]^2 \quad (4)$$

$$E_{TX}(l,d) = \begin{cases} l^* E_{elec} + l\varepsilon_{fs} d^2 (d < d_0) \\ l^* E_{elec} + l\varepsilon_{amp} d^4 (d > d_0) \end{cases} \quad (5)$$

$$E_{RX}(l) = l^* E_{elec} \quad (6)$$

In the formula, $d_0 = \sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{amp}}}$.

The computational formula of consuming by a node is as follows:

$$E_{da-fu}(l) = l^* E_{DA} \quad (7)$$

$$T = t_{work} + t_{sleep} \quad (8)$$

$$T_s = t_{s1} + t_{s2} \quad (9)$$

Currently, all MOEAs are based on non-dominated sorting approaches, which push the populations to move toward the true Pareto front. The non-dominated sorting approach has a good performs at earlier stages of the evolution; however it becomes a hypodynamic at later stages. In this article, based on analyzing the static Bayesian game model, we proposed a novel mufti-objective genetic algorithm optimization. After six different difficult tests problems, the proposed SBG-MOGA was able to maintain a better distribution than NSGA-II, and is approximate to NSGA-II in convergence in the obtained no dominated front.

4. The Experiment Result

In this part we study the performance of our Proactive model and compare the results with Monitoring Model. For the sake of this, we performed two benchmarks. In the first, the Active VM load balancer maintains information about all VMs and the number of requests currently allocated to each VM, this information collected using Active Monitoring methodology through a set of monitors built in the CloudSim architecture.

In the second benchmark, the Active VM load balancer maintains information manipulated by our Proactive model after monitoring. Each benchmark runs for 35 hours approximately where Sequence. A used as the workload request in both benchmarks. The results exhibited in the following figures. Figure 3 shows Cumulative number of peak loads occurs and figure 4 shows cumulative amount of energy overhead

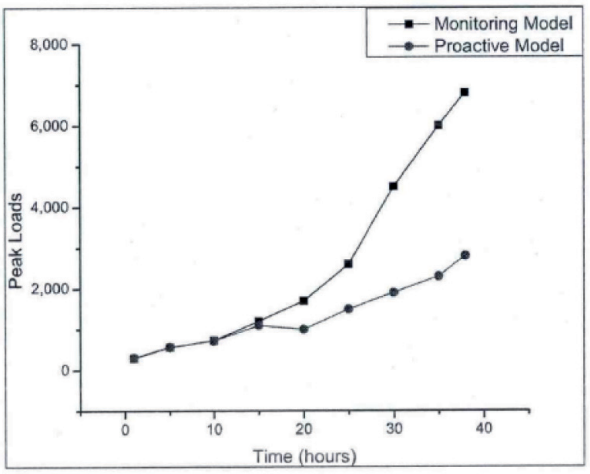

Figure 3. Cumulative Number of Peak Loads Occurs

Figure 2 shows the cumulative number of nodes suffer from Peak Loads during the experiment time; while Figure 4 shows the cumulative amount of energy overhead result from each benchmarks.

In Figure 5 we exhibit the resource wastage percentage at each instance of the benchmarks running time where we represent the resource wastage in the

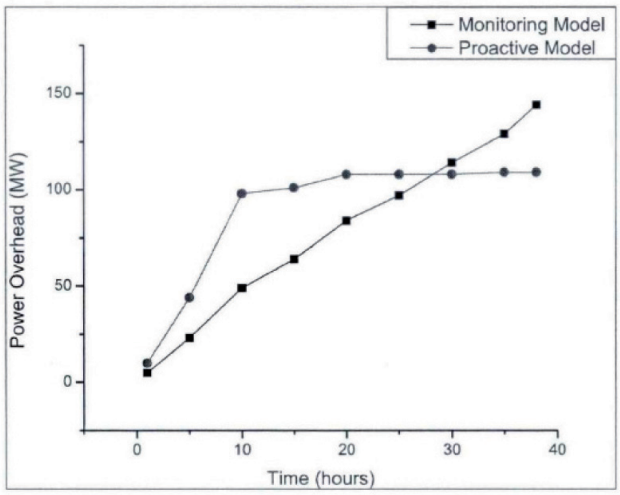

Figure 4. Cumulative Amount of Energy Overhead

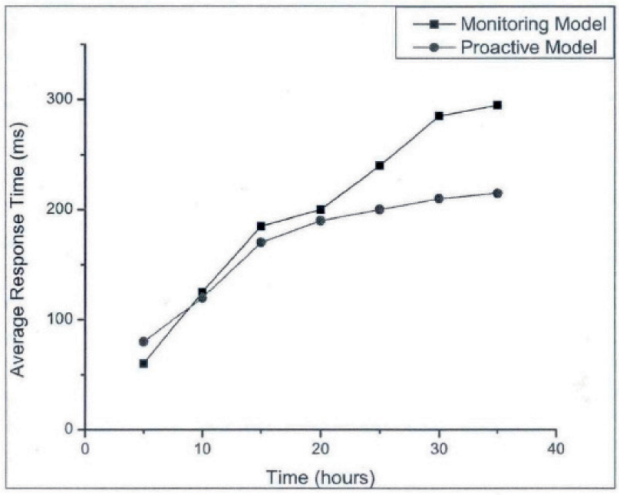

Figure 5. Cumulative Number of Peak Loads Occurs

form of $RW = \sum_{i=k}(R_i - R_k)$ where R denotes the normalized residual (the ratio of residual resource to total resource), k to identify the dimension that has the smallest normalized residual capacity, and i for dimensions. For example we consider three dimensions (i.e. CPU, memory and network) and finally, we report the average response time also at each instance of the mining time in Figure 6. From these figures, we want to highlight two observations. The first observation is that, on the long term run of the two benchmarks, the Proactive Model outperforms the monitoring model for all reported results. And the second observation is that, in all figures at the first third of the experiment time the Monitoring Model performance surpasses or almost equal to the performance of the Proactive Model, this can be directly explained referring to the experiment in the previous Section which mentions that the Proactive Model needs a learning period to start predicting.

In Xen environment resources allocation must be initiated statically for each VM before launch; however, Xen Cloud Platform comes with Dynamic Memory Control (DMC), this technology makes it possible to change the amount of host physical memory assigned to any running Virtual machine without rebooting it. Using DMC, it's possible to operate a guest virtual machine in one of two modes:

Target Mode: The administrator specifies a memory target for the guest and XCP adjusts the guest's memory allocation to meet the target.

Dynamic Range Mode: The administrator specifies a dynamic memory range for the guest. XCP chooses a target from within the range and adjusts the guest's memory allocation to meet the target.

For the sake of our experiment, we adopt the Target mode; however, after initiating resources, and later instead of tuning the resources manually we build a Workload Manager (WM) that works in two modes Figure 7.

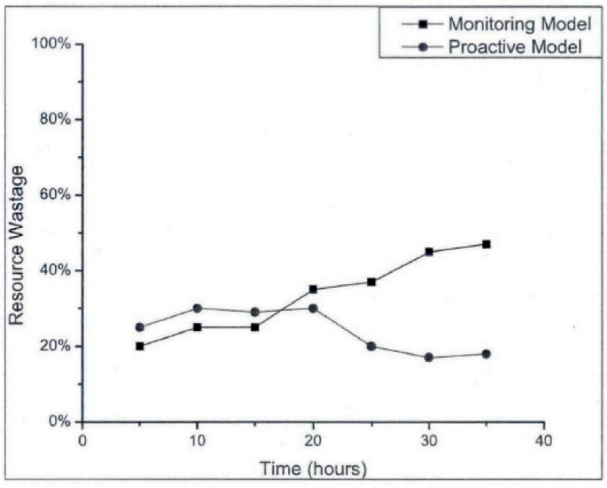

Figure 6. Occurs Amount of Resource Wastage at Each Instance of Time

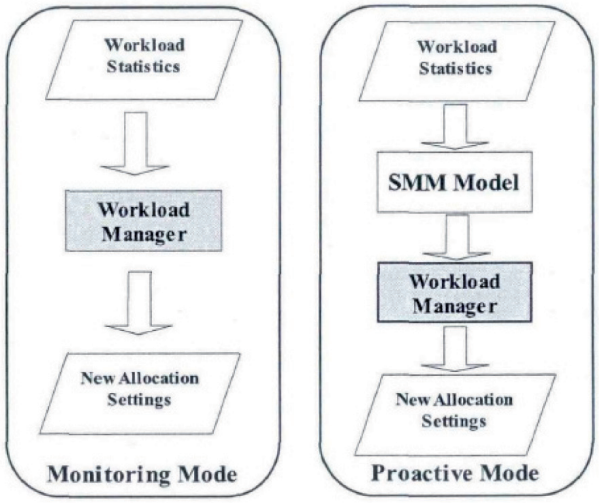

Figure 7. Workload Management Modes

Retrieving workload statistics and feed the WM directly to calculate the new recommended resources setting for each VMs, hereafter we will call this procedure Monitoring Mode.

Retrieving workload statistics, and manipulate this information with SMM prediction model, then we feed the WM with the modified information to calculate the new recommended resources setting for each VMs, hereafter we will call this procedure Proactive Mode.

The Job of VIM is not only to modify the amount of Memory assigned, but also to modify CPU Allocations. Fortunately; it is also possible to change CPU allocations dynamically in XCP. All runnable virtual CPUs (VCPUs) are managed by the local run queue on a physical CPU. This queue is sorted by VCPU priority. In the queue, every VCPU gets its fair share of CPU resources. The status of a VCPUs priority can have two values: It is "over" if it has consumed more CPU resources than it normally would be allowed to and it is "under" if it has not yet reached that value. If a VCPU has a current status of under, it will always come first when the scheduler next decides what VCPU to service. We can manage priorities by manipulating two parameters the Weight and the Cap values. The weight parameter is used to assign the amount of CPU cycles that a domain receives. Weight is relative. A VCPU with a weight of 128 would receive twice as much CPU cycles as a VCPU with a weight of 64.The second parameter to tune what a CPU may be doing, is the cap parameter. This parameter defines in a percentage the maximum amount of CPU cycles that a domain will receive. This is an absolute value; if it is set to 100, it means that the VCPU may consume 100% of available cycles on a physical CPU, if you set it to 50, then that would mean that the VCPU can consume never more than half of the available cycles.

To calculate the new resource allocation setting (priorities) with the help of XCP available techniques, the WM firstly calculate the Memory needs for each VM, then it tune resources between VMs is divided by the CPU along the same lines as the available RAM. Thus, a VM with 25% of the RAM also has a minimum share of 25% of the CPU cycles, this notice was practiced in prgmr.com.

The simple way to do this is to assign each CPU a weight equal to the number of megabytes of Memory it has, and leave the Cap empty. The Credit scheduler will then handle converting that into fair proportions, so that VM with half the RAM will get about as much CPU time as the rest of the VMs together. If all domains but one are idle, that one can have the entire CPU to itself. In multiprocessor/Multicore systems, we can translate this simple memory=weight formula to allocate VCPUs in proportion to memory; for example, a domain with half the RAM on a box with four cores should have at least two VCPUs.

It's worth to mention that, we are carefully that dom0 has sufficient CPU to service I/O requests. We handle this by giving the dom0 a very high weight, higher than any of the domUs, this by weighting each domU with its RAM amount, and weighting the domU at the total amount of physical memory in the box.

Conclusions

In this paper we investigate the Proactive Workload Management Model in a real system. First, we review the basic workload management methodology in Xen cloud system environment to give a complete understanding for the differences between the current workload procedure and the new model. Next, we described the benchmark environment and illustrate some important utilities used to complete the experiments.

In the same way we followed in the simulation experiments. We build two models to manage the workload, the first model based on the direct Monitoring methodology while the other model based on the proposed Proactive model. Through extensive set of experiments, the results show that the Proactive model can offer a great control to handle the dynamic fluctuation compared to the Monitoring model. Finally, we discussed how the history sequence patterns can affect the accuracy of the production model which is an open issue for future research.

Acknowledgements

This work is supported by Chongqing Municipal Education Commission scientific and technological research project, China (No.KJ1503209).

References

1. Furht B., Escalante. A., Handbook of cloud computing: Springer, New York, 2010.
2. Chee B., Franklin C., Cloud Computing: CRC Press, New York, USA, 2010.
3. Zeng C., Guo X., Ou W. et al., Cloud Computing Service Composition and Search Based on Semantic Cloud Computing, Lecture Notes in Computer Science, 2009, 5931(1): 290-300.
4. Vaquero L. M., Rodero-Merino L., Buyya R., Dynamically Scaling Applications in the Cloud, Computer Communication Review, 2011, 41(1): 45-52.

5. Wei-Tek T., Peide Z., Balasooriya J. et al., An Approach for Service Composition and Testing for Cloud Computing, in 2011 10th International Symposium on Autonomous Decentralized Systems (ISADS), 2011: 631-636.
6. Babcock C., Management strategies for the cloud revolution: McGraw-Hill, USA, 2010.
7. Zou G., Chen Y., Yang Y. et al., AI Palming and Combinatorial Optimization for Web Service Composition in Cloud Computing, in proceedings of International Conference on Cloud Computing and Virtualization (CCV-10), 2010: 28-36.
8. Calheiros R. N., Buyya R., De Rose C. A. F., Building an automated and self-configurable emulation test bed for applications, Software: Practice and Experience, 2010, 40(S): 405-429.
9. Hurwitz J. D., Cloud computing for dummies: Wiley Pub., Inc., Indianapolis, IN, 2009.
10. Hoffa C., Mehta G., Freeman T. et al., On the Use of Cloud Computing for Scientific Workflows, in IEEE Fourth International Conference on e-Science, Indiana Univ.. IN. USA. 2008: 640-645.

Random Forest based Online Topic Detection using Topic Graph Cluster

Qian Chen[1,2] , Zhiguo Gui[1,3,4,*] , Xin Guo[2] , Yang Xiang[5]

1 School of Information and Communication Engineering,
North University of China, Taiyuan, Shanxi,030051, China
2 School of Computer and Information Technology, Shanxi University,
Taiyuan, Shanxi, 030006, China
3 Key Laboratory of Instrumentation Science and Dynamic Measurement,
North University of China, Taiyuan, Shanxi, 030051, China
4 National Key laboratory for Electronic Measurement Technology, Taiyuan,030051, Shanxi, China
5 School of Electronics and Information Engineering, Tongji University, Shanghai, 201804, China
** Corresponding author: gui_zg@163.com*

Abstract
We proposed an online topic detection approach using random forest based on topic graph cluster which models a topic in the form of graph comprised of terms and the edges among terms. The topic graph structure can largely enhance the semantic information hidden in the corpus, thus avoided the shortcoming of bag-of-words. Random