PM&E-CP: Performance Monitor and Evaluator for Cloud Platforms

Dongbo Liu, Peng Xiao
College of Computer & Communication, Hunan Institute of Engineering, Xiangtan, 411104, China
Email: liudongbo74@126.com

***Abstract*—Cloud computing is an emerging infrastructure paradigm that promises to eliminate the need for companies to maintain expensive computing hardware. However, performance evaluation for Cloud computing platforms still remains an open issue. In this paper, we present an integrated performance evaluation middleware, which is aiming to provide users an easy-to-use toolkit to evaluate their Cloud system's runtime performance. In addition, the proposed system introduces a novel resource working model by using queue theory, which provides us a quantitative technique to evaluate the dynamic performance of virtual resources at runtime. Meanwhile, it supports a real-time performance monitoring and profiling service for high-performance Cloud applications. Massive experiments are conducted to investigate the effectiveness of the proposed system, and the results indicate that its configurable feature is very useful when users are conducting performance comparing under different cases.**

***Index Terms*—cloud computing, grid computing, performance profiling, virtual machine, workload generator**

I. INTRODUCTION

Cloud computing is an emerging infrastructure paradigm, which provides large amounts of computing and storage resources to high-performance applications with increased scalability and high availability, and reduced administration and maintenance costs [1]. By using virtualization and resource time-sharing, Cloud platforms can provide a single set of physical resources to many users with diverse requirements. Thus, Cloud computing has the potential to provide their owners the benefits, at the same time, become an alternative for both the industry and the scientific community [2].

In typical Cloud systems, virtualized resources are interconnected together and are provided to consumers' on-demand. Through well defined interfaces over well known Internet protocols, Cloud users are enabled to access to resources anytime and anywhere. Also, users can deploy their software by creating customized virtual machine images and running them on the virtualized resources in Cloud platforms. As a result, many vendors like Amazon, Google, Dell, IBM, and Microsoft are investing billions of dollars to develop their own Cloud-oriented solutions and systems [3, 4]. The Cloud providers are responsible for maintaining the underlying computing and data infrastructure; at the same time they should try best to reduce the administrative and maintenance costs for the users. This is commonly known as Infrastructure as a Service (IaaS) [2, 5].

Ideally, Clouds should provide services with performance equivalent to that of dedicated environments with similar characteristics. However, the performance characteristics of a Cloud may vary over time as a result of changes of dynamic workloads. Therefore, it is of great importance to assess the performance of Cloud infrastructure in terms of various metrics, such as the overhead of co-allocating virtual resources, and the performance of applications with different resource configurations [2, 6-7]. Unfortunately, few systems and techniques are proposed to address this issue at present. The challenges of performance evaluation in Clouds are mainly concentrated on: (1) Virtualization technique decouples the physical resources and the end-users requirements, which makes it difficult to evaluating the virtualized resource performance when user's QoS requirements are taken into account [3,8]; (2) Heterogeneous and distributed resources are dynamically composed and decomposed into abstract virtual machines (VM), the traditional off-line performance evaluation technique is not suitable for such open environments [9]; (3) The runtime performance of virtual resources are affected by too many factors. As a result, effective conclusions on performance evaluation require a flexible and configurable mechanism, which can fix some factors as well as change others at the same time [1-2, 10].

Motivated by these observations, we design and implement an integrated performance evaluation middleware, namely *Performance Monitor and Evaluator for Cloud Platforms* (PM&E-CP), with aiming to provide Cloud users and researchers an easy-to-use toolkit to evaluate their Cloud system's runtime performance, or compare the performance difference when different resource management policy and task scheduling algorithms are taken into account. In the PM&E-CP, a novel resource working model is designed by queue theory, which provides us a quantitative technique to evaluate the dynamic performance of virtual resources at runtime. In addition, it supports real-time performance monitoring and profiling service for high-performance Cloud applications.

The rest of this paper is organized as follows. Section II presents the related work. In section III, we describe the framework of PM&E-CP; In Section IV, experiments are conducted to investigate the effectiveness of our

doi:10.4304/jsw.8.4.761-767

PM&E-CP. Finally, Section V concludes the paper with a brief discussion of future work.

II. Related Work

In typical high-performance platforms, performance evaluation is often conducted by a set of standard benchmark suites [11,12]. The objective is to measure the peak and average service capability of the target systems. This is suitable for tight-couple parallel computing systems, however, it can not be applied to large-scale distributed systems since there are too many factors that will affect the performance of these systems, such as network traffic, unpredictive workload, un-reliabile resources and etc. To overcome these difficulties, researchers tends to use synthetic workload to evaluate the performance of large-scale distributed systems. Therefore, many random workload generators are proposed, i.e. Lublin-Feitelson model [13], Cirne-Berman model [14] and Tsafrir-Estion model [15]. These workload generator is based on real-world logs and allows users define their favor features of the generated workloads. As the synthetic workload generator can produce various type of random workload, researchers are enabled to compare the performance of their systems when different resource management policies or scheduling algorithms are adopted. In this paper, we incorporated three most used models to generate synthetic workload in an extensible manner.

In the past years, many studies are taken to evaluate the performance of virtualization technology. For instance, an early comparative study of the DawningCloud is deprived from performance comparison in Eucalyptus system [16]. In the study of [17], performance comparsion of executing a famous scientific workflow (Montage) are presented so as to investigate the performance trade-offs between Clouds and Grids. In [18], Palankar et al. studied the performance of Amazon S3 when large-size files are transferred between EC2 and S3. In [19], Menon et al. studied the virtual resource performance by using general benchmarks, and their results indicated that the overhead caused by virtualization can be limited below 5% for computation tasks and 15% for networking tasks. All the above studies are based on benchmarks approaches, so their conclusions are only meaningful for the tested systems. On the contrary, our framework is to provide a generalized middleware to evaluate the performance of virtual resource environments.

To assess and analyze the performance of Cloud platforms, Yigitbasi et al. designed a framework called C-Meter, which is implemented as an extension of GrenchMark [20]. It consists of four components including *Workload Generator*, *Job Core*, *Cloud Interaction* and *Utility Toolkit*. By using C-Meter, users can assess the overhead of acquiring and releasing the virtual computing resources, also they can evaluate the performance of different scheduling algorithms under different configurations. Unlike C-Meter, our PM&E-CP framework applies a general model to describe the working status of individual VMs. So, it provides a general approach to evaluating the performance of VMs, which in turn can be used to profile the execution performance of Cloud applications.

III. The Framework of PM&E-CP

A. The Architecture of PM&E-CP

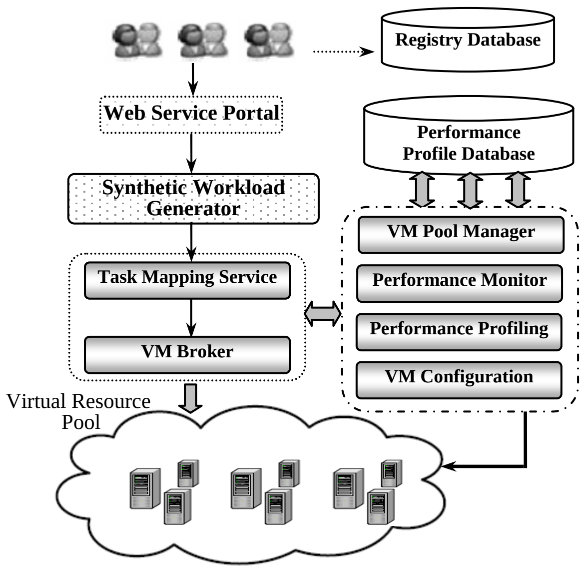

Fig. 1. The Architecture of PM&E-CP

The architecture of PM&E-CP is illustrated in Fig. 1, which is composed of four key components including *Web Service Portal*, *Synthetic Workload Generator*, *VM Scheduler* and *VM Manager*. The brief descriptions of these components are presented as following: (1) *Web Service Portal* is responsible for accepting user's benchmark requirements which is described by a set of XML files. (2) *Synthetic Workload Generator* is responsible for translating the user's abstract requirements into a set of workloads. The generated synthetic workloads are characterized by several factors, such as task arriving interval, task execution time, task resource requirements, task type and etc. (3) *VM Scheduler* works like conventional meta-scheduler but we separate the task mapping and VM broker by two subcomponents. It is because that we can easily replace the scheduler algorithm or policy when we want to evaluate the performance of different scheduling algorithm. (4) *VM Manager* consists of four subcomponents, and they are designed for VM resource pool management and VM performance analysis. For example, the provision of VM is controlled by VM pool manager and VM configuration, and the runtime performance and capability of individual VM are monitored and logged through Performance Monitor and Performance Profiling subcomponents.

Generally speaking, PM&E-CP is designed and implemented as a portable and extensible framework for generating and submitting both real and synthetic workloads so as to analyze the performance of Cloud environments. In PM&E-CP, VM management policy and task scheduling algorithm can be flexibly plugged in at runtime. In this way, researchers and administrators are able to compare the effectiveness and efficiency of different policies and algorithms. So, it can be used as a test-bed middleware for large-scale Cloud computing systems.

B. The Procedure of using PM&E-CP

In PM&E-CP, there are six steps when users conduct performance evaluation for Cloud-based systems. The details of each step are described as following:

Step 1. Users submit their benchmark files to the *Service Portal* service. These files define all the requirements of this performance evaluation, including the task type and size, scheduling algorithm, VM configuration and etc. All of these requirements have a default setting if they are not explicitly specified in the files. So, users can ignorant many settings and only focus on their most interested parts, i.e. task scheduling algorithm.

Step 2. According to the benchmark files, *Synthetic Workload Generator* component will use proper random workload model to produce a set of corresponding synthetic workloads. Currently, the PM&E-CP use Lublin-Feitelson model [13] as default workload generator, which is derived from real workload logs of large-scale distributed systems. Other candidates include Cirne-Berman model [14] and Tsafrir-Estion model [15]. The VM configurations specified by users are applied to corresponding subcomponents in *VM Manager*.

Step 3. The *Task Mapping Service* schedules the synthetic workloads through pre-defined task algorithms. It is noteworthy that VM resource allocation and task dispatching are carried by *VM Broker* components. So *Task Mapping Service* only produces a scheduling scheme in this step.

Step 4. *VM Broker* is responsible VM allocation and task dispatching. In this step, *VM Broker* maintains several queues for each active VM in the virtual resource pool with aiming to monitor the availability of VMs.

Step 5. When *VM Broker* obtains enough available VMs for executing current task, it dispatches the task onto one or more VMs for execution.

Step 6. The subcomponents in *VM Manager* is always monitoring and logging the performance statistics of VM instances. When it notices a new VM configuration requirement, corresponding action will be taken by certain subcomponents. All the performance logs are organized and stored in a separated profiling database, which can be used for later analyzing.

C. Design of Online Performance Profiling Service

In the framework of PM&E-CP, performance monitor and profiling service are two novel components, which are designed for online performance evaluation for virtual resources in Cloud environments. As mentioned above, *VM Broker* maintains task queues of each active VM in virtual resource pool. When these VMs are serving for arriving tasks, the workload of each VM will dynamically changed at runtime. Consequently, the practical serving capability will fluctuate dramatically with the changing of its workloads.

In order to realize online performance monitoring and profiling service for these virtual resources, we apply the classical Stochastic Queue Model [21] to describe the working status of each VM. Therefore, each active VM can be described by a set of quantitative parameters, such as mean length of waiting queue, mean serving time, parallel serving capability and etc. By collecting this performance information, we can apply queue theory to analyze the performance of each VM, and then evaluate the execution performance of the whole system. So, the flowchart of online performance profiling in PM&E-CP is shown in Fig. 2.

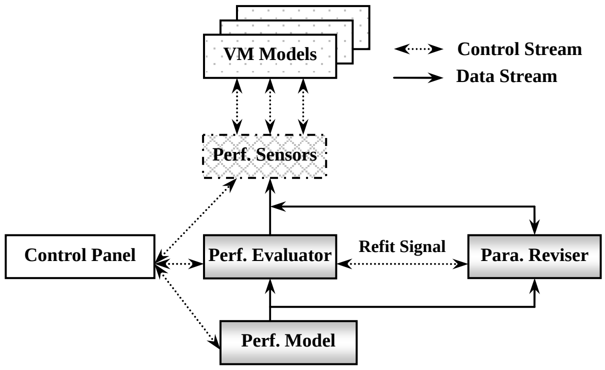

Fig. 2. Flowchart of Online Performance Profiling

At first, a set of *performance sensors* are hooked when a VM is in active state. Each sensor is responsible for monitoring a single measurement of all the active VMs. These measurements are analyzed by *performance evaluator* to figure out the statistic features of a certain parameter. Meanwhile, a closed loop controlling is designed for parameter revising and refitting. The *control panel* is a set of options to configure the online performance profiling, such as error limitation, frequency of sampling and etc. Finally, the performance parameters are sent to *performance model* to construct a proper queue model that can properly describe the current performance status of the active VMs. The queue model of an active VM is 6-tuple noted as $<A(t), B(t), C(t) D(t), U(t), F(t)>$, where $A(t)$ is probability density function (PDF) of task arriving interval, $B(t)$ is the PDF of serving time, $C(t)$ mean length of waiting queue, $D(t)$ is the parallel serving capability, $U(t)$ is the real-time rate of utility, $F(t)$ is the fault rate at runtime. After obtaining such a 6-tuple model, it is stored into the performance profiling database with distinguished time stamp.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we present the experimental results on the PM&E-CP framework. At present, we mainly concern about the effects of VM management policy and scheduling algorithm on the performance Cloud systems. So, we conduct extensive experiments with different VM management policies and scheduling algorithms.

A. Performance Comparison with Different VM Management Policies

Generally speaking, VM management policy includes the VM provision policy and VM price policy. The VM provision policy is to decide that how many VMs should be provided so as to satisfy the user demands. We mainly test three policies, which are MTPP (Maximize Throughput Provision Policy) [16], MRPP (Minimize Response-time Provision Policy) [23], MUPP (Maximize Utilization Provision Policy) [16]. So, our first experiment is to test the performance with different VM provision policies.

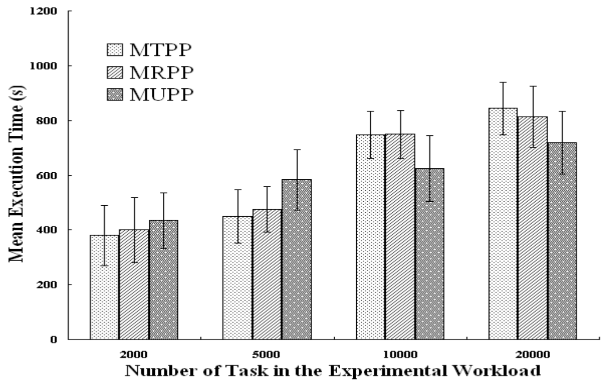

Fig. 3. Mean Execution Time with various VM Provision Policies

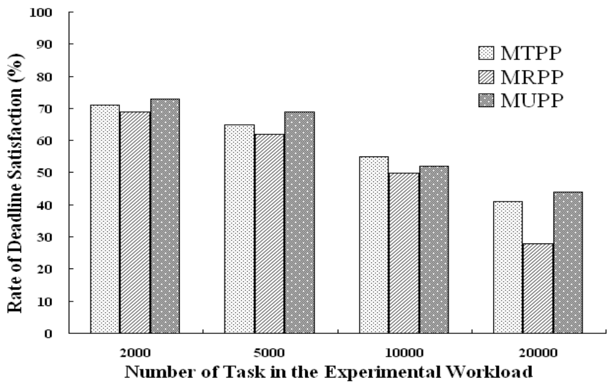

Fig. 4. Rate of Deadline Satisfaction with Various VM Provision Policies

In this experiment, we used synthetic workload generator to produce four sets of workloads, and the number of tasks in each workload are 2000, 5000, 10 000 and 20000 respectively. Each task is characterized by its arrival time, resource demands, estimation of execution time and a cost budget. The performance metrics we concern are *Mean Execution Time* and *Deadline Satisfaction Rate*. The experimental results are shown in Fig. 3 and Fig. 4.

From the above results, we notice that size of workload is of significant importance when evaluating the performance of a VM provision policy. For example, the MTPP policy outperforms MRPP and MUPP in term of *Mean Execution Time* when the size of workload is less than 10000. However, its performance reduces quickly when workload becomes heavy. On the contrary, the performance of MUPP policy seems relative stable when workload increases from 2000 to 20000. By this result, we might draw a conclusion that MUPP is more adaptive in presence of dynamical workload.

Deadline Satisfaction Rate metric is to measure that how many tasks can be completed before its deadline constraint. This metric is very important for soft real-time tasks since high deadline-missing rate might cause results in execution failure. By out experimental result, we can see that MUPP is most effective to meet the deadline constraint and MRPP performs worst. At same time, workload size also affects this metric. Simply saying, heavy workload will significantly lower down the deadline satisfaction rate. In detail, the effects of workload size are also different when using different VM provision policy. For instance, such negative effects on MRPP policy are biggest.

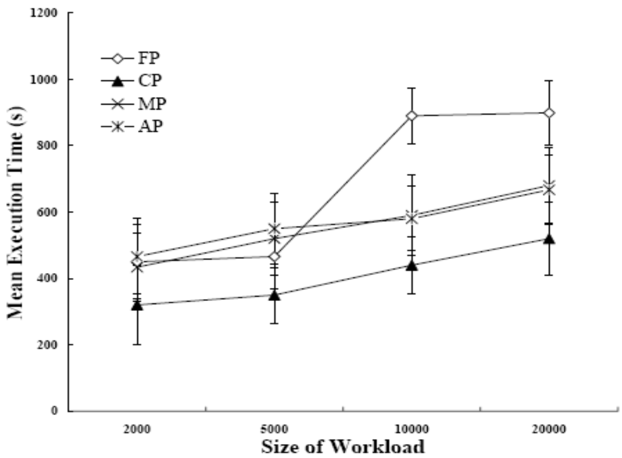

Fig. 5. Mean Execution Time with various Price Mechanisms

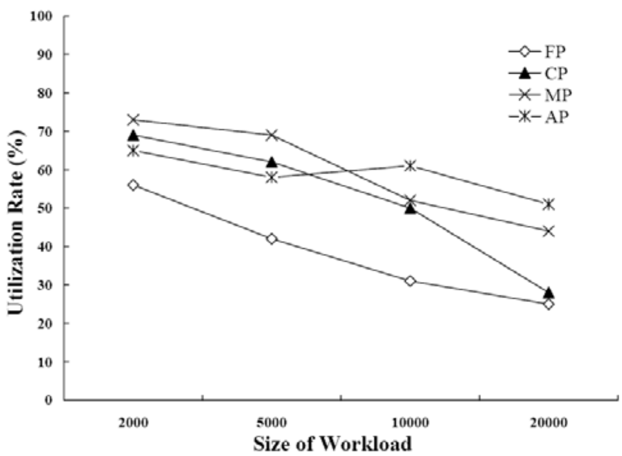

Fig. 6. Resource Utilization Rate with with various Price Mechanisms

The VM price mechanism is to decide that how much resource providers should charge when users access their resources. Fixed price mechanism is the most frequently used one in current Cloud systems. Recently, several market-based price mechanisms have been proposed by researchers. We implement four price mechanisms in the prototype of PM&E-CP, including Fixed Price (FP)[18], Capability-based Price (CP)[23], Market-based Price (MP) [25], Auction-based Price (AP)[24]. In this experiment, each VM price mechanism are examined with MUPP as VM provision policy. The results are shown in Fig. 5 and Fig. 6.

As shown in Fig. 5, VM price mechanism also has effects on the task execution. FP mechanism is very sensitive to the workload size; however other three mechanisms seem insensitive to it. In all the four price mechanism, CP performs best, the performance of MP and AP are almost the same. So, we might conclude that Capability-based price mechanism is effective to improve the task execution efficiency. The other metric we measured are Resource Utilization Rate. It is well known that high price will result in low utilization when resources are allocated by economic computing principle. Our experimental results confirm this again, since MP and AP outperform FP and CP. At present, many real world Cloud systems are using FP mechanism because of its easy implementation. However, out results indicate FP mechanism will results in low resource utilization rate. So, these Cloud systems are of great potential to improve their resource utilization if applying more flexible price mechanisms.

B. Performance Comparison with Different Scheduling Algorithms

In our PM&E-CP framework, task scheduling algorithms are implemented in *Task Mapping Service* component. In order to provide an extensible framework for comparing the performance of different algorithms, *Task Mapping Service* exposes an abstract interface, namely ITaskScheduling, which can be implemented in different approaches. At present, we implement four kinds of scheduling algorithms, which are *Round-Robin Algorithm* (RRA) [26], *Capability-based Random Algorithm* (CRA) [27], *Cluster Minimized Algorithm* [28] (CMA), *Task Duplication Algorithm* [29] (TDA). All these algorithms are widely studied and used in many high-performance distributed systems. Since PM&E-CP only uses abstract interface for task scheduling, anyone can incorporate other algorithms into it.

In order to test the performance of different scheduling algorithms, we need to fix other factors which have effects on the final results. So, we conduct the experiments four times, each with an identical VM price mechanism. Then, we control the provision of VM by increasing the number of VM from 50 to 500. The size of workload is set as 20 times of the VM number in each experiment. In this way, we can clearly comparing the performance of different algorithms under different conditions. For Task Duplication Algorithm (TDA), we set the redundant degree K=2 and K=3. The results of these experiments are shown in Fig. 7 ~ Fig. 10.

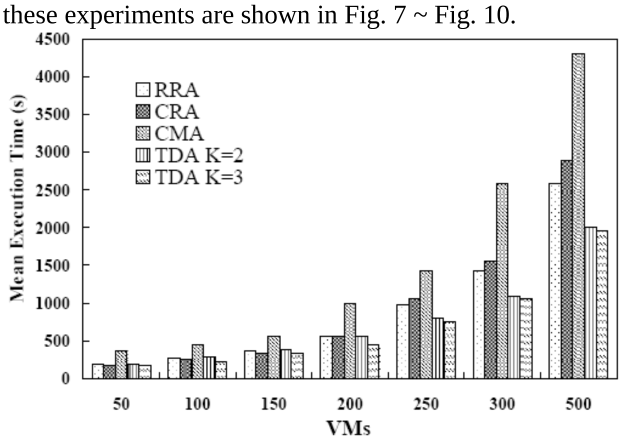

Fig. 7. Mean Execution with FP Mechanism

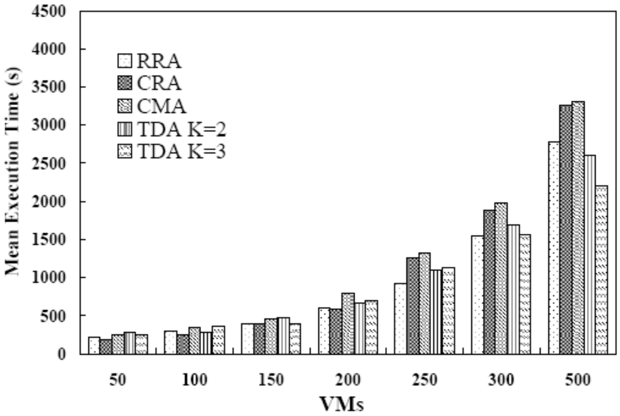

Fig. 8. Mean Execution with CP Mechanism

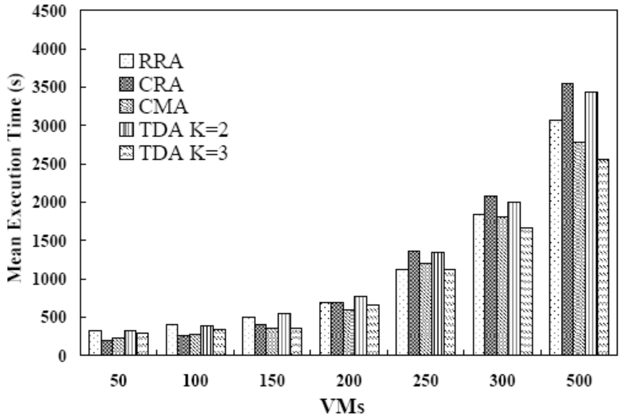

Fig. 9. Mean Execution with MP Mechanism

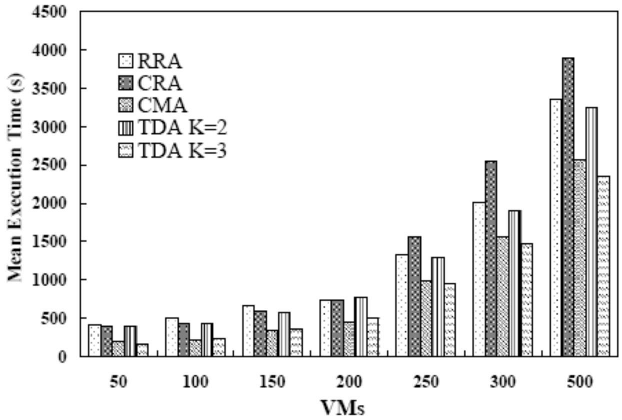

Fig. 10. Mean Execution with AP Mechanism

From the above results, we can see that the task execution time increases with the increasing of VM numbers. It is because that our test workload size is always 20 times of VM number. An interesting finding is that four algorithm seem perform the same when MP mechanism is used (as shown in Fig. 9). By examining the experimental data in detail, we find that MP mechanism is of great effects on the VM allocation. So, it plays an dominate role when scheduling tasks onto VMs. As to the FP mechanism, the importance of scheduling algorithm is significant great than the VM price mechanism. As shown in Fig. 7, the performance differences of these algorithms are most significant. Generally speaking, we notice that TDA outperforms other algorithms in term of *Mean Execution Time*, especially when K=3. It is because that TDA always dispatches multiple tasks replica onto different VMs, and select the quickest results as the task's return. Even thought, we still can not say that TDA is the best algorithm of these four algorithms, since it will result in low effective utilization especially with large K value. RRA is a very simple scheduling algorithm; however, we notice that its performance keeps relatively stable when using different VM price mechanisms. Maybe, it is why many practical systems use it as the default task scheduling algorithm.

Summary the above experiments, we obtain the following conclusions: (1) The performance of Cloud systems are effected by many factors, even more, the factors might be effected by each other; (2) There is no best resource management policy nor best task scheduling algorithm. The target system should make its management decisions according the features of dynamic environment and its design objectives; (3) PM&E-CP is an extensible framework, which provides an easy-to-use toolkit to test the performance of Cloud systems and the effectiveness of different resource management policies or scheduling algorithms.

VI. Conclusion

In this paper, we present an integrated performance evaluation middleware, namely PM&E-CP, which is aiming to provide users and researchers an easy-to-use toolkit to evaluate their Cloud system's runtime performance, or compare the performance when different resource management policy and task scheduling algorithms are taken into account. The PM&E-CP is consists of four key components including *Service Portal*, *Synthetic Workload Generator*, *VM Scheduler* and *VM Manager*. In order to realize online performance monitoring and profiling service for these virtual resources, we apply the classical queue model to describe the working status of each VM. Massive experiments are conducted to investigate the effectiveness of the proposed system, and the results indicate that its configurable feature is very useful when users are conducting performance comparing under different conditions.

Acknowledgment

This work was supported by a grant from the National Natural Science Foundation of China (No. 60970038). It is also a project supported by Scientific Research Fund of Hunan Provincial Education Department (No. 09c270). The authors sincerely thank the hard work of reviewers, and thank Ms. Lee Xi in Chicago University for the correction of English writing.

References

[1] L. Youseff, M. Butrico, and D. DaSilva, "Towards a Unified Ontology of Cloud Computing," Proc. Grid Computing Environments Workshop (GCE '08), Nov. 2008.

[2] R. Prodan and S. Ostermann, "A Survey and Taxonomy of Infrastructure as a Service and Web Hosting Cloud Providers," Proc. Int'l Conf. Grid Computing, pp. 1-10, 2009.

[3] Amazon, Inc., "Amazon Elastic Compute Cloud (Amazon EC2)," http://aws.amazon.com/ec2/, Dec. 2008.

[4] GoGrid, "GoGrid Cloud-Server Hosting," http://www.gogrid. com, Dec. 2008.

[5] C. Teixeira, R. Azevedo, J.S. Pinto and T. Batista. "User Provided Cloud Computing". Proc. of IEEE/ACM Int'l Conf. on Cluster, Cloud and Grid Computing, pp.727-732, 2010.

[6] A. Iosup, S. Ostermann, M.N. Yigitbasi, et al. "Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing", IEEE Trans. on Parallel and Distributed System, vol.22, no.6, pp.931-945, 2011.

[7] N Yigitbasi, A Iosup, D Epema, S Ostermann. "C-Meter: A Framework for Performance Analysis of Computing Clouds". Proc. of IEEE/ACM Int'l Symp.on Cluster Computing and the Grid, pp.472-477, 2009.

[8] U.F. Minhas, J. Yadav, A. Aboulnaga, and K. Salem, "Database Systems on Virtual Machines: How Much Do You Lose?" Proc. IEEE 24th Int'l Conf. Data Eng. (ICDE) Workshops, pp. 35-41, 2008.

[9] B. Quetier, V. Neri, F. Cappello, "Scalability Comparison of Four Host Virtualization Tools," J. Grid Computing, vol. 5, no. 2, pp. 83-98, 2007.

[10] A.B. Nagarajan, F. Mueller, C. Engelmann, and S.L. Scott, "Proactive Fault Tolerance for HPC with Xen Virtualization," Proc. ACM 21st Ann. Int'l Conf. Supercomputing (ICS), pp. 23-32, 2007.

[11] A. Kowalski, "Bonnie:File System Benchmarks," technical report, Jefferson Lab, http://cc.jlab.org/docs/scicomp/benchmark/ bonnie. html, 2002.

[12] D.G. Feitelson, L. Rudolph, U. Schwiegelshohn, K.C. Sevcik, and P. Wong, "Theory and Practice in Parallel Job Scheduling," Proc. Job Scheduling Strategies for Parallel Processing (JSSPP), pp. 1-34, 1997.

[13] U. Lublin, D.G. Feitelson, "Workload on Parallel Supercomputers: Modeling Characteristics of Rigid Jobs," J. Parallel and Distributed Computing, vol. 63, no. 11, pp.1105-1122, 2003.

[14] W. Cirne and F. Berman, "A Comprehensive Model of the Supercomputer Workload". 4th Ann. Workshop Workload Characterization, 2001.

[15] D. Tsafrir, Y. Etsion and D.G. Feitelson, "Modeling User Runtime Estimates". 11th Workshop on Job Scheduling Strategies for Parallel Processing (JSSPP), pp. 1-35, Jun 2005.

[16] D. Nurmi, R. Wolski, C. Grzegorczyk, et al., "The Eucalyptus Open-Source Cloud-Computing System," Technical Report 2008-10, uCSD, http://eucalyptus.cs.ucsb.edu/, 2008.

[17] E. Deelman, G. Singh, M. Livny, J.B. Berriman, and J. Good, "The Cost of Doing Science on the Cloud: The Montage Example," Proc. IEEE/ACM Supercomputing (SC), pp. 41-50, 2008.

[18] M.R. Palankar, A. Iamnitchi, M. Ripeanu, and S. Garfinkel, "Amazon S3 for Science Grids: A Viable Solution?" Proc. DADC '08: ACM Int'l Workshop Data-Aware Distributed Computing, pp. 55-64, 2008.

[19] A. Menon, J.R. Santos, Y. Turner, G.J. Janakiraman, and W. Zwaenepoel, "Diagnosing Performance Overheads in the Xen Virtual Machine Environment," Proc. ACM First Int'l Conf. Virtual Execution Environments (VEE), pp. 13-23, 2005.

[20] A Iosup, D Epema. "GrenchMark: A Framework for

Analyzing, Testing, and Comparing Grids". Proc. of Int'l Symp. on Cluster Computing and the Grid, pp.313-320, 2006.
[21] D. Gross, C.M. Harris. Fundamentals of Queuing Theory. USA: John Wiley and Sons, 1998.
[22] R. Clarke. "User Requirements for Cloud Computing Architecture". Proc. of IEEE/ACM Int'l Conf. on Cluster, Cloud and Grid Computing, pp.625-630, 2010.
[23] R. Jeyarani, R.V. Ram, N. Nagaveni. "Design and Implementation of an efficient Two-level Scheduler for Cloud Computing Environment". Proc. of IEEE/ACM Int' Conf. on Cluster, Cloud and Grid Computing, pp.585-586, 2010.
[24] W.Y. Lin, G.Y. Lin, H.Y. Wei. "Dynamic Auction Mechanism for Cloud Resource Allocation". Proc. of IEEE/ACM Int'l Conf. on Cluster, Cloud and Grid Computing, pp.591-592, 2010.
[25] Y.C. Lee, C. Wang, et al. "Profit-driven Service Request Scheduling in Clouds". Proc of IEEE/ACM Int'l Conf. on Cluster, Cloud and Grid Computing, pp.15-24, 2010.
[26] C.L. Dumitrescu, I. Raicu, I. Foster. "The Design, Usage, and Performance of GRUBER: A Grid Usage Service Level Agreement based Brokering Infrastructure". J. of Grid Computing, vol.5, no.1, pp.99-126, 2007.
[27] V. Berten, J. Goossens, E. Jeannot. "On the Distribution of Sequential Jobs in Random Brokering for Heterogeneous Computational Grids". IEEE Trans. on Parallel and Distributed Systems, vol.17, no.2, pp.113-124, 2007.
[28] H.H Mohamed, D.H.J Epema. "Experiences with the KOALA Co-Allocating Scheduler in Multiclusters". Proc of Int'l Symp on Cluster Computing and the Grid, pp.784-791, 2005.
[29] H Casanova. "Benefits and Drawbacks of Redundant Batch Requests". J. of Grid Computing, vol.5, no.2, pp. 235-250, 2007..

Dongbo Liu was born in 1974. He received his master degree in Jianshu University in 2001. Now he works as a associate professor in Hunan Institute of Engineering, also he is a Ph.D candidate in Hunan University. His research interests include distributed intelligence, multi-agent systems, high-performance application. He is a member of CCF association and IEEE Computer Society.

Peng Xiao was born in 1979. He received his master degree in Xiamen Universy in 2004. Now, he works in Hunan Institute of Engineering and is a Ph.D candidate in Central South University. His research interests include grid computing, parallel and distributed systems, network computing, distributed intelligence. He is a member of IEEE and ACM.