International Journal of Hybrid Information Technology
Vol.6, No.5 (2013), pp.287-296
http://dx.doi.org/10.14257/ijhit.2013.6.5.27

The Performance Improvement of an Enhanced CPU Scheduler Using Improved D_EDF Scheduling Algorithm

Chia-Ying Tseng[1] and Po-Chun Huang[2]

Department of Computer Science and Engineering, Tatung University, Taipei, Taiwan (R.O.C.)

[1]cytseng@ttu.edu.tw, [2]g10006025@ms.ttu.edu.tw

Abstract

With the development of cloud computing, mobile device and industry automation, virtualization plays an important role today. The core of virtualization is hypervisor which directly determines the performance of platform. Therefore, how to allocate resource effectively becomes an important issue. Xen, one of widely used open source projects, is a virtual machine monitor. Simple EDF (Earliest Deadline First) scheduler that is a dynamic-priority real-time scheduler in Xen implements the famous EDF scheduling algorithm. Due to EDF scheduling strategy has miss the deadline and inefficiency in overloaded condition, we have improved the D_EDF scheduling algorithm that combines Deadline-Monotonic scheduling strategy with EDF scheduling in virtualized environment. And we have extended the Simple EDF scheduler which provided by Xen using improved D_EDF scheduling algorithm. Our experiment demonstrates that the proposed scheduling algorithm increased the performance under CPU-intensive and memory-intensive workload in overloaded condition.

Keywords: *Scheduling Algorithm, Virtualization, Earliest Deadline First Scheduling*

1. Introduction

With the development of cloud computing, mobile device and industry automation, virtualization is widely adopted. It has also been widely applied to enterprise infrastructure, embedded virtualization, and embedded systems [1]. The features of virtualization are multi virtual machines and isolated virtual machines share a physical machine. It brings many benefits, including power management, cost down, easy to manage and increase CPU utilization.

The core of virtualization is virtual machine monitor (hypervisor) that is responsible for allocate resource and manage virtual machine. Therefore, hypervisor is directly deciding the performance of platform. As Figure 1 shows, hypervisor includes type-1 and type-2. Type-1 (Native hypervisor) which has its own scheduler is running directly on the hardware and allocates resource to virtual machine. Some examples are Xen Hypervisor and VMware ESX, etc. The type-2 (Hosted hypervisor) is running on the operating system level. Since type-2 hypervisor does not have scheduler itself, therefore, depends on operating system task scheduler. Some examples are VirtualBox and VMware Workstation.

ISSN: 1738-9968 IJHIT

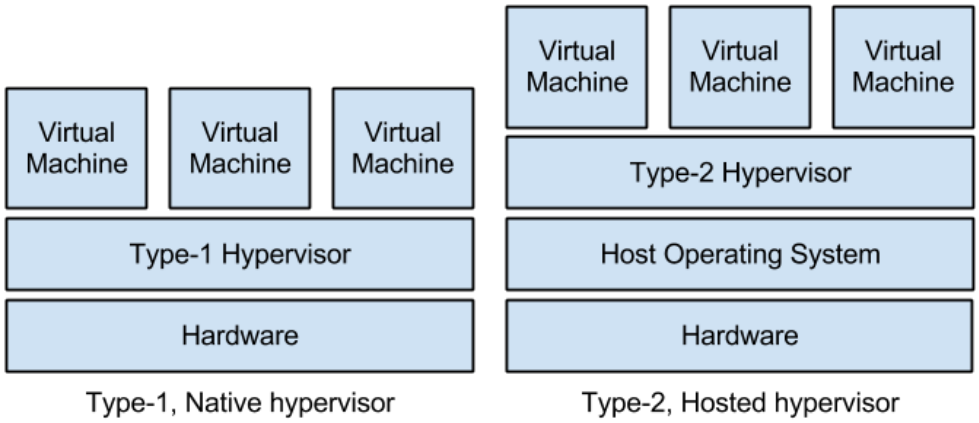

Figure 1. Native hypervisor and hosted hypervisor architecture

2. Background

This section will introduce Xen virtualization. And it will describe the advantages and disadvantages of schedulers which are Credit and SEDF in Xen, and Real-time scheduling algorithm.

2.1. Xen Virtualization

Xen virtualization technology has para-virtualization and full virtualization [2]. Para-virtualization guest virtual machine has better performance than full virtualization guest virtual machine, but need to modify guest operating system kernel. On the other hand, full virtualization which is implemented based on the hardware assisted method (Intel VT and AMD-V) used unmodified guest operating system [3]. In this paper, we used para-virtualization machines as experiment platform.

Virtualization has three general classes: CPU, memory, and I/O [4]. We will focus on virtual CPU (VCPU).

2.2. Schedulers in Xen

Xen has Credit and Simple Earliest Deadline First (SEDF) scheduler. Credit scheduler is default scheduler in Xen 4.2.1 version. In Symmetric Multiprocessor (SMP) architecture, Credit scheduler achieves global load balancing [5], but is latency-sensitive and I/O-intensive [6]. Credit scheduler has two parameter, *weight* and *cap* [7], *weight* stands for the weight of use CPUs between virtual machine. The default value of *weight* is 256. If a domain which *weight* is 512 can get two times of resources than the domain *weight* is 256. The *cap* stands for physical CPU (PCPU) usage percentage, decides how many resources that domain can get. The default value is zero, means WC-mode, there is no limit. If *cap* is 50, means NWC-mode, it can be assigned half PCPU. If *cap* is 100, it can be assigned one PCPU. In a similar way, *cap* value equals to 400, it can be assigned four PCPUs.

SEDF, which is implemented by famous EDF scheduling algorithm [8], is a dynamic-priority real-time scheduler. It provides five parameters which are period, slice, latency, *extra* and *weight*. It has different result with different combination of five parameters. In real-time environment, SEDF basically use period and slice, fully specify a domain. The period determined the period of VCPU. And slice stands for the worse case

execution time. The *extra* is a Boolean type, determined a domain whether use remaining time. Therefore, domain which set *extra* equals 1 uses Round-robin (RR) scheduling to share the available time.

Although EDF have been proved to be an optimal algorithm for single processor system under the preemptive and under-loaded condition. In higher load, it guarantees deadline can be met, but in over-loaded condition it will cause many of deadline miss.

2.3. Related Scheduling Algorithm

Deadline-Monotonic (DM) scheduling is fixed-priority preemptive scheduling algorithm. Deadline is relative with absolutely deadline. In higher load, it can't achieve real-time like the EDF. But in the overloaded condition, it can guarantee the higher priority task met the deadline and let lower priority task miss deadline. Therefore, in overloaded condition, DM is better than EDF.

Susi Xi [9] proposed RT_XEN which provide four type schedule strategies. The Deferrable Server is fixed-priority scheduling algorithm which has higher capacity and better than other three strategies. It also assigns priority to tasks of guest operating system by Rate-Monotonic (RM) scheme. And implement the hierarchy real time scheduling in Xen.

Devendra Thakor [10] proposed D_EDF scheduling algorithm which combines EDF with DM. Switch algorithm by recording the deadline miss count and deadline met count. As Figure 2 shows, if two jobs miss the deadline continuously occur then switch EDF to DM. If ten jobs achieve the deadline then switch back to EDF. In this way, we combine with those advantages of two algorithms to speed up overall performance.

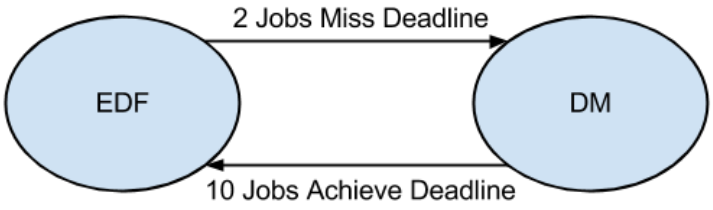

Figure 2. D_EDF switching threshold

This is a good concept, but there was an issue about the threshold of switch. For example, the number of domain is dynamic in Xen environment. Power on a domain, the number of VCPU is increase. Power down a domain, the number of VCPU is decrease. So the number of VCPU is dynamically in Xen environment. Consider the two situations that we have 5 and 512 of VCPUs in soft real-time systems which 6% deadline miss ratio is significant. If 2 of the 5 VCPUs that miss deadline mean miss ratio equal 40%, then we should switch to DM to guarantee high priority achieve the deadline. If 2 out of the 512 VCPUs that miss deadline mean miss ratio equal 0.3%, then it is not necessary to switch scheduling algorithm. Therefore, directly consider threshold by the deadline miss ratio is better than deadline miss count.

In this paper, we improve the D_EDF depend on miss ratio as threshold in Xen environment. We will discuss in the next section.

3. Improved D_EDF Scheduling Algorithm

Virtualization brings many benefits include improve CPU utilization, scalability, and energy efficiency, *etc*. For enjoying these benefits they need to meet the soft and hard real-time demand, and have to do more tests. As described in Section 2, EDF algorithm will miss deadline in overloaded condition. D_EDF is a solution of guarantee that the high priority task is completed on schedule. In the Xen environment, the number of VCPU is dynamic. It can be 1~512. How to improve D_EDF that accurately determine the system overload will be described in Section 3.2.

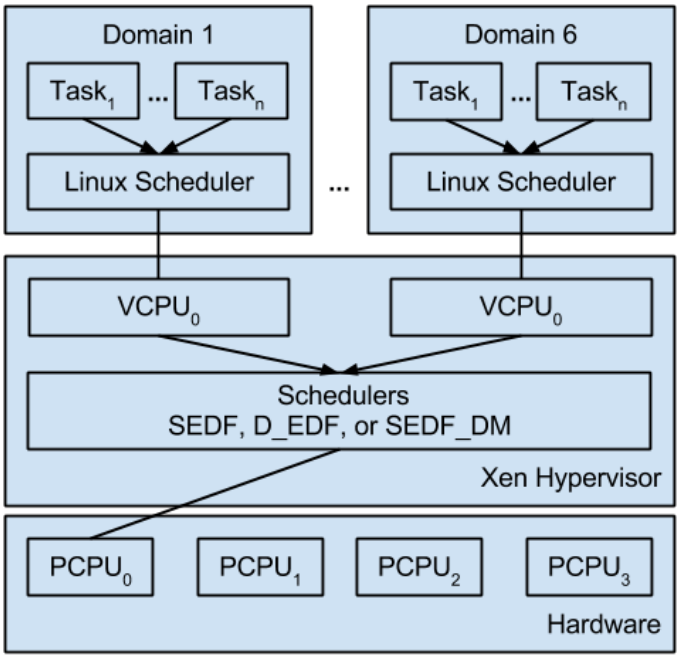

Figure 3. System Model and two level hierarchies

3.1. Scheduling Model

The virtualization environment has two level hierarchies as shown on Figure 3. If it is nested virtualization, then it has multi-level. Here, we have two levels where guest operating system scheduler is responsible for scheduling the tasks, and virtual machine monitor scheduler is responsible for deploying VCPU on PCPU. Every domain has allocate one VCPU and unmodified kernel. All the VCPU 0 of domain-U are used PCPU 0. And VCPU 0 of Domain-0 pin to PCPU 1.

3.2. Improvement of D_EDF

To avoid high frequency switching between two strategies that may cause heavy overhead by little change, define a suitable threshold is very important. First, scheduler must record deadline miss count and number of VCPU, then calculate deadline miss

ratio. Second, it will calculate the deadline miss ratio once the total VCPUs have recorded greater than 256. Figure 4 shows, checking deadline miss ratio if it is achieving threshold or not. If deadline miss ratio greater than 6% means system overloaded, then switch to DM from SEDF. Otherwise, if deadline miss ratio equal zero, then switch to SEDF from DM. Finally, reset the *miss_count* and *total* after check the deadline miss ratio.

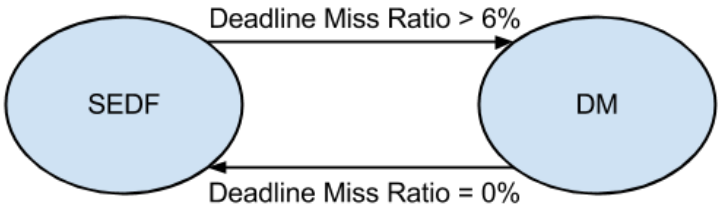

Figure 4. SEDF_DM switching threshold

3.3. SEDF_DM

We extend the SEDF algorithm using improved D_EDF algorithm. SEDF defines that each PCPU has a runnable queue which store runnable VCPU defined in SEDF. The runnable queue is ordered by using *deadl_abs* as comparing operator. When a VCPU needed to be run, it picks the first VCPU from the queue and returns it.

We create a scheduler called *sched_sedfdm.c* which based on *sched_sedf.c* that Xen provided. The scheduler sorts the runnable queue by using merge sort with variable slice for DM. When switch back to SEDF, re-sorting the run queue with variable *deadl_abs*.

In *update_queues()* as Figure 5 shows, create two variable that *miss_count* and *total* to record deadline miss and total number of VCPUs in queue. If deadline miss, then *miss_count* increase. And determine whether deadline miss ratio greater than 6% by following relationship:

```
FOR each VCPU in runnable queue
    total increase
    IF deadline miss THEN
        miss_count increase
    END IF
END FOR
```

$$Deadline\ Miss\ Ratio = \frac{Miss\ Count}{Total} > 6\%$$

Figure 5. Pseudo code of partly *update_queue()*

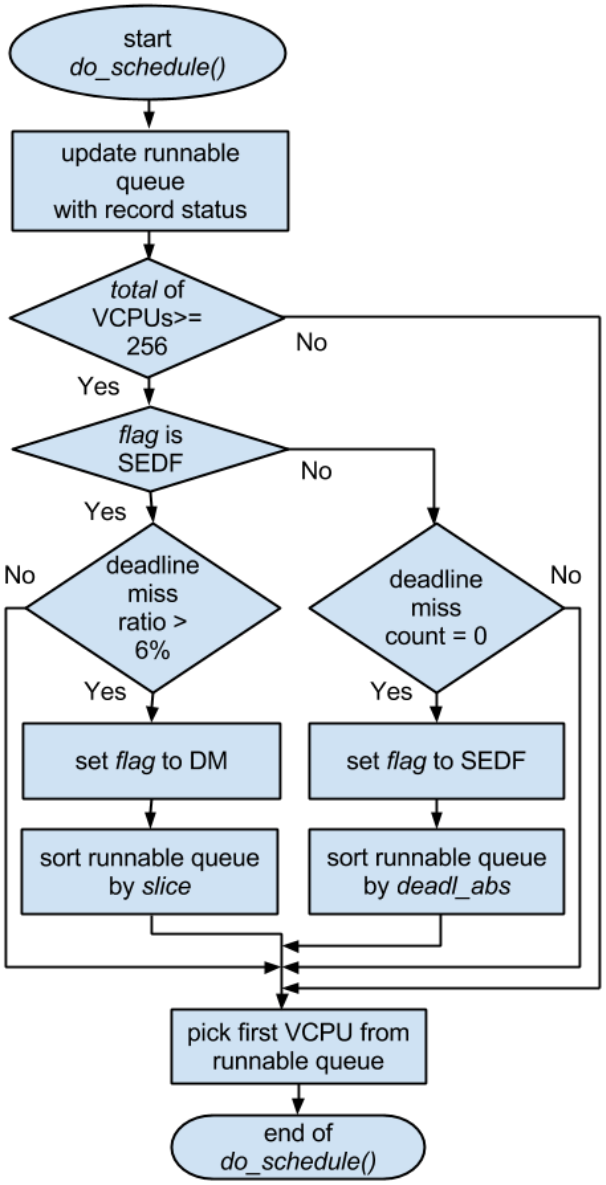

Figure 6. Define switching threshold in *do_schedule()*

Figure 6 shows the *do_schedule()* algorithm. If Deadline Miss Ratio is greater than 6 percent, then switches to DM. If Deadline Miss Ratio equals zero, then switches back to SEDF.

The sort algorithm used in sorting runnable queue is merge sort which is also used in Linux kernel. Because the runnable queue is implemented by link list, the merge sort is a solution for sorting the link list.

We implement D_EDF based on *sched_sedf.c* in Xen. Create two variables miss_count and met_count that record deadline miss and deadline met in *update_queues()*. And define the switching threshold in *do_schedule()*.


```
IF total >= 256 THEN
    IF current is SEDF
      AND miss_count > total x 6% THEN
        switch to DM
        sort runq by slice
    ELSE IF current is DM AND miss_count = 0 THEN
        switch back to SEDF
        sort runq by deadl_abs
    END IF
END IF
```

Figure 7. Pseudo code of partly *do_schedule()*

4. Evaluation

This section introduces experiment environment and benchmarks. First, measure performance of 6 domain-Us by running NPB. Second, we evaluate average kernel latency of 4 domain-Us with 100% CPU load using Cyclictest.

The experiment hardware platform was on Intel® Core™ i7-920 2.66 GHz (Turbo 2.93 GHz) disabled hyper-threading and Enhanced Intel SpeedStep technology, MSI X58 Pro-E motherboard, six 2 GB DDR3 SDRAM memory, and WD5000AALS 500 GB disk. Software platform was on Kernel 3.2.0-38, Ubuntu 12.04.2 LTS server 64-bit and Xen 4.2.1 version.

Domain-0 has a VCPU with pin to PCPU 1, 1 GB memory, and 40 GB LVM. All domain-U which paravirtualization machine each has a VCPU without pin, 1 GB memory, and 40 GB LVM. The six domain-U named Domain-1, Domain-2, …, and Domain-6. The values of slice were 50, 54, 58, 62, 66 and 70. And the value *extra* was zero.

We use Numerical Aerodynamic Simulation Parallel Benchmarks (NAS Parallel Benchmarks, NPB) and Cyclictest to measure SEDF, D_EDF, and SEDF_DM performance and kernel latency. Several domain-Us run ep.A which a parallel program in NPB. The "A" class means problem size. Each scheduler test 5 times and calculate average benchmark time. Figure 8 shows the average CPU time with different scheduler which x-axis means number of domain and y-axis means CPU time in seconds. When a domain-U is running, the average CPU time of D_EDF and SED_DM are approximately the same as SEDF. A domain-U in underloaded condition, D_EDF and SEDF_DM doesn't switch to DM, so the overhead increased by recording deadline miss count and total or met count can be negligible. When 4 domain-Us were running, due to overloaded condition, the benchmark time of SEDF_DM is about 84% that of SEDF. When 6 domain-Us were running, due to overloaded condition, the benchmark time of SEDF_DM is about 84% that of SEDF. The benchmark time of SEDF_DM is about 87% that of D_EDF.

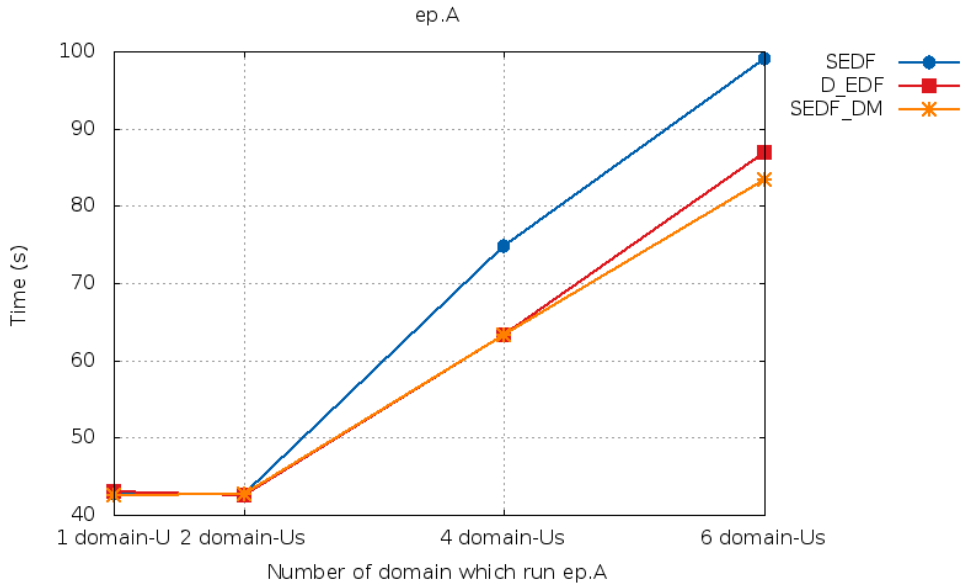

Figure 8. Measure CPU Time of each scheduler

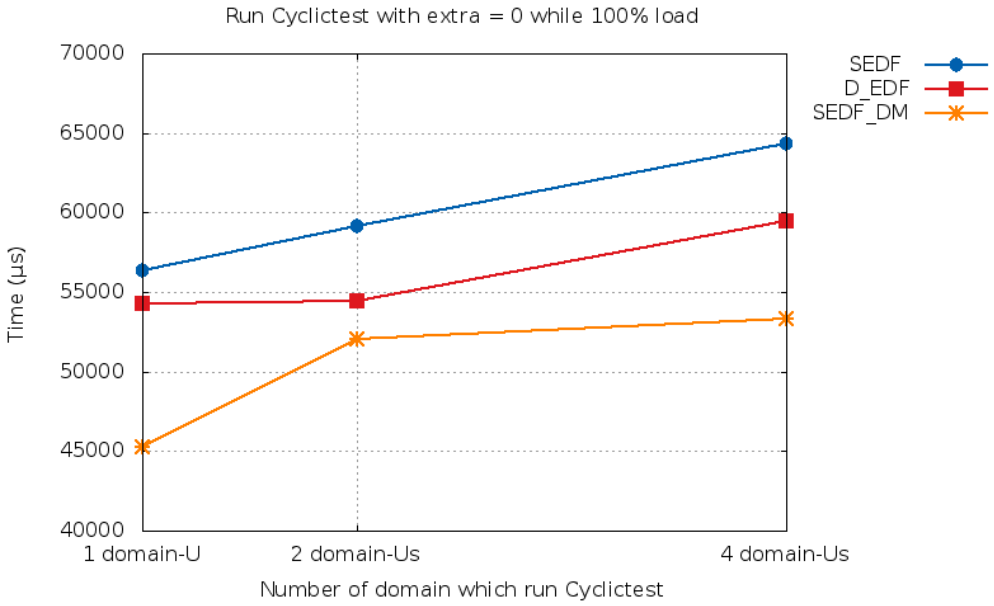

Figure 9. Measure kernel latency of each scheduler with *extra* = 0

Cyclictest is used to evaluate kernel latency. To make the 100% CPU load environment, Domain-5 and Domain-6 both run CPU-intensive work. Domain-1~Domain-4 run simultaneously the Cyclictest with 150,000 times. In Figure 9, x-axis shows number of domain and y-axis means kernel latency in microseconds. In 100% load condition without sharing remaining time using RR, the average kernel latency of SEDF_DM is about 83 percent of SEDF, and SEDF_DM is 89 percent of D_EDF.

In Figure 9, x-axis shows number of domain and y-axis means kernel latency in microseconds. The result shows that the kernel latency of SEDF_DM is 82 percent of SEDF, and SEDF_DM is 90 percent of D_EDF.

On the other way, we measure kernel latency of a para-virtualization virtual machine without load. The result shows D_EDF and SEDF_DM were approximately the same as SEDF.

5. Conclusion and Future Work

In this paper, we implement SEDF_DM scheduler which based on improved D_EDF algorithm, and compare SEDF, D_EDF and SEDF_DM's performance in virtualized environment. The experiment result was shown that the overhead of records which are deadline miss count, total or met count is very little. We evaluate the kernel latency in under-loaded and overloaded condition. The result shows D_EDF and SEDF_DM are approximate the same as SEDF in under-loaded condition. The CPU time of SEDF_DM is 64~93 percent of SEDF, and kernel latency is 80~87 percent of SEDF in overloaded condition.

With the development of mobile device and industry automation, embedded virtualization is more and more important today. Xen is a lightweight native hypervisor, but it does not good enough to support SMP architecture and Real-time environment. In the default situation, all the domain's VCPU were used PCPU 0. If we want to let SEDF work on multiprocessor, then use "xm vcpu-pin" command that make a VCPU pin to other PCPU core. Although SEDF can work on multiprocessor, it does not support global load balancing and VCPU migration scheme.

References

[1] S. Yoo, M. Park and C. Yoo, "A step to support real-time in virtual machine", 2009 IEEE 6th Consumer Communications and Networking Conference (CCNC), **(2009)**, pp. 1-7.
[2] L. Lee, K. Liu, H. Huang and C. Tseng, "A Dynamic Resource Management with Energy Saving Mechanism for Supporting Cloud Computing", International Journal of Grid and Distributed Computing (IJGDC), vol. 6, no. 1, **(2013)**, pp. 67-76.
[3] K. Ye, X. Jiang, S. Chen, D. Huang and B. Wang, "Analyzing and modeling the performance in xen-based virtual cluster environment", 2010 IEEE 12th International Conference on High Performance Computing and Communications (HPCC), **(2010)**, pp. 273-280.
[4] X. Zhang and D. Yin, "Real-time improvement of VCPU scheduling algorithm on Xen", 2011 International Conference on Computer Science and Service System (CSSS), **(2011)**, pp. 1506-1509.
[5] P. Yu, M. Xia, Q.Lin, M. Zhu, S. Gao, Z. Qi, K. Chen and H. Guan, "Real-time Enhancement for Xen hypervisor", 2010 IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC), **(2010)**, pp. 23-30.
[6] H. Kim, H. Lim, J. Jeong, H. Jo and J. Lee, "Task-aware virtual machine scheduling for I/O performance", Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments, **(2009)**, pp. 101-110.
[7] C. Tseng, S. Lin, L. Chen and H. Chung, "The Performance Improvement and Evaluation of an Enhanced CPU Scheduler in Virtualized Environment", Proceedings of the 2011 International Conference on e-CASE & e-Tech, **(2011)**, pp. 3107-3123.
[8] A. Masrur, S. Drössler, T. Pfeuffer and S. Chakraborty, "VM-based real-time services for automotive control applications", 2010 IEEE 16th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), **(2010)**, pp. 218-223.
[9] S. Xi, J. Wilson, C. Lu and C. Gill, "RT-Xen: Towards Real-Time Hypervisor Scheduling in Xen", 2011 Proceedings of the International Conference on Embedded Software (EMSOFT), **(2011)**, pp. 39-48.
[10] D. Thakor and A. Shah, "D_EDF, An efficient scheduling algorithm for real-time multiprocessor system", 2011 World Congress on Information and Communication Technologies (WICT), **(2011)**, pp. 1044-1049.
[11] Z. Zheng, Y. Zhang, Z. Wang, X. Zhang and L. Lu, "The Multi-Processor Load Balance Scheduler Based on XEN", 2012 International Conference on Systems and Informatics (ICSAI), **(2012)**, pp. 905-909.

[12] C. Tseng and K. Liu, "A Modified Priority Based CPU Scheduling Scheme for Virtualized Environment", International Journal of Hybrid Information Technology, vol. 6, no. 2, **(2013)**, pp. 39-49.
[13] N. Guerreiro and O. Belo, "Predicting the Performance of a GRID Environment: An Initial Effort to Increase Scheduling Efficiency", 2010 International Journal of Grid and Distributed Computing (IJGDC), vol. 3, no. 1, **(2010)**, pp. 31-38.

Authors

Chia-Ying Tseng received a bachelor degree in industrial education from Taiwan Normal University in 1979 and a master degree in computer engineering from the Electrical and Computer Engineering Department of Arizona State University, USA, in 1985. In 1990, he was selected by Ministry of Education to enter the PhD program in the Computer Science and Engineering Department of Arizona State University for one year research. Currently, he is an Assistant Professor in the Computer Science and Engineering Department of Tatung University, Taiwan. His research interests include Embedded System Design, Embedded Real Time Operating System, Multiprocessor System-on-Chip (MPSoC), Cloud Computing and Virtualization.

Po-Chun Huang received the B.S. degree in Department of Computer Science and Engineering from Tatung University in 2011, and his M.S. degree in Computer Science and Engineering from Tatung University in 2013. His research interest is Cloud Computing and Virtualization.

