An Efficient Virtual CPU Scheduling Algorithm for Xen Hypervisor in Virtualized Environment

Chia-Ying Tseng and Po-Chun Huang

Department of Computer Science and Engineering, Tatung University
#40, Sec. 3, Zhongshan N. Rd., Taipei, Taiwan (R.O.C.)
1cytseng@ttu.edu.tw, 2g10006025@ms.ttu.edu.tw

Abstract. The core of virtualization is hypervisor which directly determines the platform performance. How to allocate resource effectively becomes an important problem. Xen is an open source hypervisor and used as a virtual machine monitor. In this paper, we designed an efficient virtual CPU scheduling algorithm that combined Deadline-Monotonic Scheduling with Simple Earliest Deadline First (EDF) scheduler which was implemented on Xen. Our experiment demonstrates that the proposed algorithm reduced kernel latency better than Simple EDF in overloaded condition.

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

1 Introduction

With the development of cloud computing, mobile device and industry automation, virtualization is widely adopted, include enterprise infrastructure, embedded virtualization and embedded systems. 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. 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, depend on operating system task scheduler. Some examples are VirtualBox and VMware Workstation.

2 Background

Xen has Credit and Simple Earliest Deadline First (SEDF) scheduler. SEDF, which is implemented by famous EDF scheduling algorithm, 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.

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.

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 over-loaded condition, it can guarantee the higher priority task met the deadline and let lower priority task miss deadline [1]. Therefore, in over-loaded condition, DM is better than EDF scheduling algorithm.

Devendra Thakor [2] proposed D_EDF scheduling algorithm which combines EDF with DM. Switch algorithm by recording the deadline miss count and deadline met count. 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.

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 is 40%, then we should switch to DM to guarantee high priority to meet the deadline. If 2 out of the 512 VCPUs that miss deadline mean miss ratio is 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 scheduling algorithm depend on miss ratio as threshold in Xen environment.

3 Improved D_EDF Scheduling Algorithm

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.

The scheduling model 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.

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 deadline met count, then calculate deadline miss ratio. Second, it will check the deadline ratio of past 256 VCPUs if it is achieving threshold or not. Define each 256 VCPUs as one period. Observe past period and decide which strategy should be used on next period. If deadline miss ratio greater than 6% means system overloaded, then switch to DM from SEDF. Otherwise,
if deadline miss ratio equal zero in past period, then switch to SEDF from DM. Finally, reset the miss_count and total every period.

![Fig. 1. SEDF_DM switching threshold](image)

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(), 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:

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

![Fig. 2. Pseudo code of partly update_queue()](image)

### 4 Evaluation

#### 4.1 Experiment Setup

The experiment hardware platform was on Intel® Core™ i7-920 2.66 GHz (Turbo 2.93 GHz) without hyper-threading, 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.

4.2 Comparison of SEDF, D_EDF and SEDF_DM

We use Cyclictest to measure kernel latency of SEDF, D_EDF, and SEDF_DM. 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 Fig. 3, x-axis means number of domain and y-axis means average kernel latency time in microseconds. In 100% load condition, the average kernel latency of D_EDF is 81% ~ 89k% of SEDF, and SEDF_DM is 82% ~ 85% of SEDF. In no load condition, D_EDF and SEDF_DM are approximately the same as SEDF.

![Fig. 3. The kernel latency of each scheduler](image)

5 Conclusion

In this paper, we implement SEDF_DM scheduler which based on improved D_EDF algorithm, and compare kernel latency of SEDF, D_EDF and SEDF_DM. 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 underloaded 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 89 percent of SEDF, and kernel latency is 86 percent of SEDF in overloaded condition.
An Efficient Virtual CPU Scheduling Algorithm for Xen Hypervisor in Virtualized Environment

References