Advanced Science and Technology Letters
Vol.85 (Information Technology and Computer Science 2015), pp.38-41
http://dx.doi.org/10.14257/astl.2015.85.09

A Study on the Priority Scheduling Technique for Smart Virtual Machine

Yunsik Son[1], Jaehyun Kim[2], Yangsun Lee[2*]

[1] Dept. of Computer Engineering, Dongguk University
26 3-Ga Phil-Dong, Jung-Gu, Seoul 100-715, KOREA
sonbug @dongguk.edu
[2] Dept. of Computer Engineering, Seokyeong University
16-1 Jungneung-Dong, Sungbuk-Ku, Seoul 136-704, KOREA
{statsr, yslee}@skuniv.ac.kr
[*]Corresponding Author

Abstract. This paper presents the priority scheduling for the thread model of Smart Virtual Machine (SVM). The existing SVM have assured the execution time to all created threads equally, so that it was not able to distribute time in accordance with the importance of threads. We designed scheduling technique which suits to the characteristics of SVM through analyzing the existing priority scheduling.

Keywords: Priority scheduling, Multithread, Smart Virtual Machine

1 Introduction

The thread model of existing SVM have assigned equal weight to the all created threads, so that it was not able to allocate priority and execution time separately in accordance with the importance of threads [1, 2, 3].

In this paper, the priority scheduling is designed for the thread of SVM based on the priority scheduling which operating systems service including Windows and Linux and the priority interface which programming languages provide including as C/C++, Java. Through this, we composed the programming and execution environment to adopt the thread priority with various languages provided by SVM.

2 Priority scheduling

In general, a process or a thread under multi process or multi thread environment has the attribute of priority, and it varies the occupancy of CPU. The execution time of a process or a thread can be allocated separately in accordance with the priority which is assigned depending on the importance of task and it allows a particular thread to be allocated with more working hours [4,5].

ISSN: 2287-1233 ASTL

Priority scheduling is mainly categorized into preemptive Scheduling and nonpreemptive Scheduling. First, preemptive scheduling is to allocate higher priority threads on CPU ahead. Preemptive scheduling allows the process to forcibly quit the other process on CPU currently and to occupy CPU instead. It is good for the time sharing system which is favored when the higher priority process executes ahead and requires fast response time [6].

Then under nonpreemptive scheduling, once a process is allocated to CPU, CPU cannot be get out of the process until it finishes. Although it handles a request from all of processes fairly and it is good for predicting the response time, it has weakness which makes the work with short execution time to wait for long time.

3 Priority Scheduling System for Smart Virtual Machine

The proposed priority scheduling algorithm has simple structure to reduce the overheads, and maintain the thread ready queue which designed by list per each priority. Figure 1 is details of the proposing algorithm.

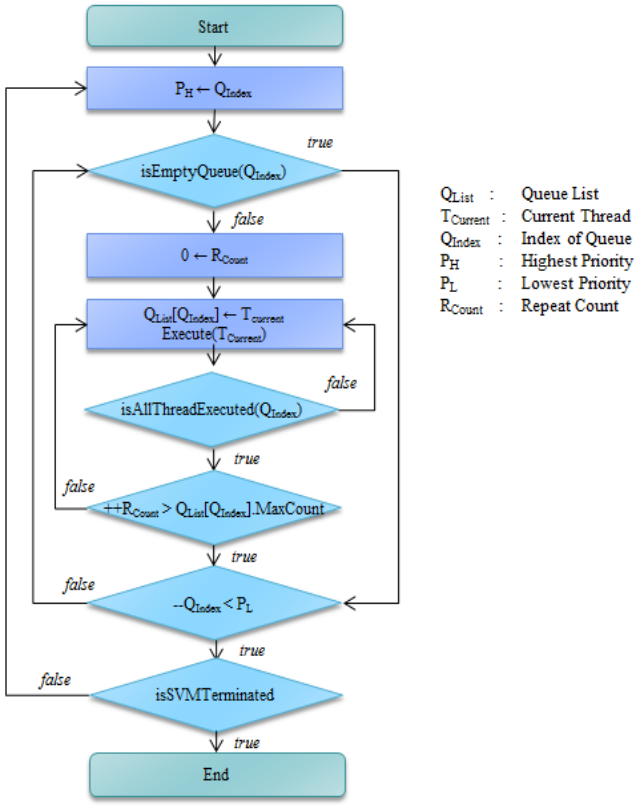

Fig. 1. Proposed priority scheduling algorithm for Smart Virtual Machine

While the priority scheduling of SVM adopts the nonpreemptive basically, it allocates the different number of time slot repetition in grade of priority to allocate working hours for all threads and to allocate more time for higher priority threads simultaneously unlike the existing nonpreemptive techniques. The priority of thread indicates by the level of 1 to 10, and the number of time slot repetition for each priority is on the table 1.

Table 1. Priority levels and repetition counts of time slot unit for each thread.

Priority level	Repetition Counts
Lowest (1~3)	1
Normal (4~7)	5
Highest (8~10)	10

4 Experimental Results

The following section is about the performance analysis of priority scheduling algorithm. It tested the queueing model with the proposed algorithm in chapter 3, and the generation frequency of thread in order of priority is assumed with the normal distribution. The testing simplified the level of priority from 10 to 3 and the priority and the values of cumulative probability for each level are on table 2.

Table 2. Priority levels and repetition counts of time slot unit for each thread.

Priority level	Cumulative probability value
Lowest (1~3)	0.28613
Normal (4~7)	0.49628
Highest (8~10)	0.1873

The working hours for all the generating threads are fixed to 40ms, and the delay is planned to take place by fixing the number of thread per second to 30. Figure 2 is the result for the testing.

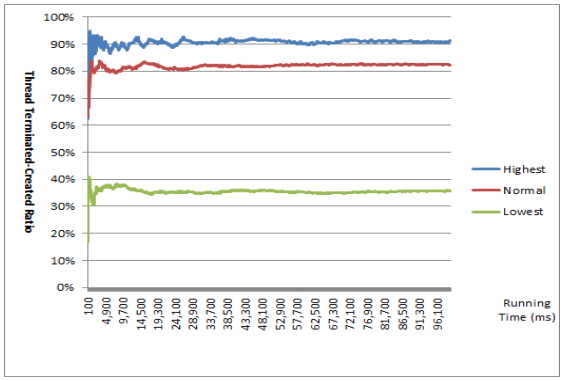

Fig. 2. Test result of proposed priority scheduling algorithm

5 Conclusions and further researches

This paper adds the priority to the thread model of SVM to assure discriminate execution time for each thread. It is confirmed that despite the thread with lower priority shows more delay time compared to the thread with higher priority, it still maintains certain level of service time. The way of minimizing the delay time of entire threads by controlling the duration of time slot or the number of repetition per thread dynamically based on the generation information of thread is remained for the future work.

Acknowledgments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (No.2013R1A2A2A01067205).

References

1. Lee, Y.S., Son, Y.S.: A Study on the Smart Virtual Machine for Executing Virtual Machine Codes on Smart Platforms. International Journal of Smart Home, SERSC, Vol. 6, No. 4, 2012, Australia, pp. 93-105.
2. Lee, Y.S., Son, Y.S.: A Study on the Smart Virtual Machine for Smart Devices. Information -an International Interdisciplinary Journal, Vol. 16, No. 2, International Information Institute, 2013, Japan, pp.1465-1472.
3. Lee, Y.S., Oh, S.M., Son, Y.S.: Design and Implementation of HTML5 based SVM for Integrating Runtime of Smart Devices and Web Environments. International Journal of Smart Home, SERSC, Vol.8, No.3, 2014, Australia, pp.223-234.
4. Baruah, S.K.: Dynamic and Static-priority Scheduling of Recurring Real-time Tasks. Journal of Real-Time Systems, Vol 24, No. 1, 2003, USA, pp. 93-128.
5. Ramamritham, K.: Scheduling Algorithms and Operating Systems Support for Real-time Systems. Proceedings of the IEEE, Vol. 82, No. 1, USA, 1994, pp. 55-67.
6. Stallings, W.: Operating Systems: Internals and Design Principles, Pearson Educacion, 2011, USA.

