EDZL Scheduling for Large-Scale Cyber Service on Real-Time Cloud

Tseng-Yi Chen
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
tsengyi2005@gmail.com

Hsin-Wen Wei
Department of Information Management
TamKang University
Taipei, Taiwan
141131@mail.tku.edu.tw

Jenq-Shiou Leu
Department of Electronic Engineering
National Taiwan University of Science and Technology
Taipei. Taiwan
jsleu@mail.ntust.edu.tw

Wei-Kuan Shih
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
wshih@cs.nthu.edu.tw

Abstract—[+]**In recent years, physical sensor devices are used in many different applications, such as: Internet of Things (IoT), military, vehicle system, etc. Sensor devices are widely utilized in the real world so that a large-scale process becomes an important issue in such a cyber physical system. Large scale cyber physical system (LCPS) must have enough computing power for processing all real-time sensed data from sensor devices. LCPS can deploy a cloud platform to increase its computing power. The cloud based LCPS then serves these requests at the same time by well arranging sensor tasks based on the arrival time. Hence, cloud service can obtain better performance and more efficiency when good scheduling algorithms are implemented. In this paper, apply the early deadline first until zero laxity (EDZL) scheduling algorithm to optimize the cloud service scheduler for ensuring sensor jobs do not miss deadline in the LCPS.**

I. INTRODUCTION

Physical sensor devices are employed in a variety of applications, such as: smart building, military, habitat monitoring, etc. The application domain can be from monitoring to controlling devices. Hence, it is better to design a LCPS to manage the sensed data from physical sensor devices. LCPS must have enough computing power for processing all real-time data from these sensor devices. With assistance of a cloud platform, a cloud based LCPS can enhance its computing power.

Cloud computing provides flexible and scalable computing resources, such as computation power and data storages, to end-users. The concept of cloud service is also applied to consume computing power and hardware resource capacity in the air [1]. Dynamically dispatching these requests to computing entities in the cloud system becomes an important problem. In addition to the resource utilization, the real-time issue in the cloud service is also a major concern since some real-time applications and services have a deadline guarantee. For instance, LCPS needs a good scheduling algorithm on cloud to ensure sensor jobs do not miss deadline. Therefore, a good scheduling algorithm is highly demanded in cloud computing.

Task scheduling on cloud service has been studied for a long time. These scheduling solutions regard the computing platform as a parallel architecture. One approach is to employ the traditional parallel scheduling algorithm [2]. Tsai et al proposed a service-oriented solution concept [3]. In [4], [5], Shuo Liu et al also proposed preemptive and non-preemptive scheduling algorithms in the service-oriented architecture. In [6], authors introduced a data locality driven task scheduling algorithm. However all these algorithms mentioned above ignored the analysis of their utilization bound and schedulability.

This paper aims to show how to apply the early deadline first until zero laxity (EDZL) algorithm to the cloud service scheduling. The EDZL is a hybrid algorithm of Early Deadline First (EDF) and Least Laxity Algorithm (LLA) [7], [8]. EDZL schedules tasks based on both their deadlines and laxity values to obtain a better scheduling bound than EDF and a lower cost of context switching than LLA. This algorithm works based on dynamic promotion priorities. The laxity value of a task determines a task's promotion priority. Therefore, EDZL can be a good scheduling solution to a real-time cloud connected to all sensors and actuators. In [4], the authors compared the performance of scheduling algorithm to the EDF algorithm. Based on the result, the EDF and variant EDF scheduling algorithms are suitable for being executed in the cloud service architecture.

The rest of this paper is organized as follows. Section II shows the cloud task model and some scheduler in cloud computing. Section 3 presents some assumptions in the cloud service. The EDZL algorithm for the cloud scheduler and its fundament properties in the cloud service also are depicted in Section III. In Section IV, we discuss the future work about the schedulability of EDZL in the cloud environment. The brief conclusion is presented in Section V.

II. BACK GROUND

A. Cloud Service Task Model

In the cloud service architecture, a real-time architecture with *m* homogeneous computing units is assumed to have *n* real-time tasks sorted by their arrival times. The sorted task sequence is denoted as $\Gamma = \{ \tau_1, \tau_2, \ldots, \tau_n \}$. Each task τ_i has their own parameters value of c_i and d_i, such as $\tau_i = (c_i, d_i)$. The c_i is worst case execution time and d_i is relative deadline for ensuring QoS in cloud service.

[+] Supported in part by National Science Council (Taiwan) Grants NSC 99-2220-E-007-015, NSC 99-2220-E-007-021

A cloud service generates a sequence of tasks. A ready task running on a cloud computing platform is called the current task. During the run time, a job of task τ_i can be characterized by J_i, with J_i defined by the same parameters in [8]. The laxity value of a job J_i at time t can be rewritten as $L_i(t)=t_i(t) - e_i(t)$. The parameters of $e_i(t)$ and $t_i(t)$ are the remaining execution time and the remaining time to deadline at time t. The laxity of a job provides a measure of *urgency* for this job. A negative laxity indicates that the job can never meet its deadline, while a zero laxity indicates that the job must be started immediately otherwise it would miss deadline.

B. Scheduler in Cloud Computing

Scheduling in the cloud system has been widely studied in recent years. In [2], Moschakis and Karatza employed gang scheduling in a cloud computing system. They evaluated the performance and cost of gang scheduling in a cloud service platform. They have implemented two scheduling strategies - adaptive first com first serve (AFCFS) and largest task first served (LJFS) in their scheduling scheme. These two strategies do not consider the deadline constrains so that they cannot ensure the quality of service for cloud services.

The service-oriented scheduling concept is introduced by Tsai et al [3]. They employed task duplication and database partitioning and used a simplest scalable scheduling algorithm. They focused on the scalability of scheduling algorithm. There is also no deadline constrain in their proposed model. Some other on-line scheduling algorithms have been proposed for cloud services [4] [5]. In [4], Liu proposed a non-preemptive scheduling algorithm based on the service-oriented cloud architecture. However, for cloud services that need to be executed with a real-time constrain, the preemption supported method is more realistic and necessary. Thus, Liu proposed another preemptive on-line scheduling solution in [5]. They have presented a novel preemptive utility accrued scheduling algorithm. Their algorithm chooses highly profitable tasks to be executed first. Moreover, the algorithm cannot ensure the schedulability for the cloud tasks.

Another scheduling solution was proposed in [6]. Jin et al proposed a data locality driven task scheduling algorithm that schedules tasks by adjusting the data locality of the task. Jin's solution only improved the data locality in the scheduling algorithm. However, they did not analyze the schedulability of cloud tasks.

In this paper, we focus on the schedulability of cloud tasks. If there is a cloud tasks set $\Gamma = \{ \tau_1, \tau_2,\ldots, \tau_n\}$ and each task's properties are as mentioned earlier, we develop an on-line preemptive scheduling method for the cloud task set Γ to ensure the schedulability for the task set. Our solution has introduced two useful concepts including the *dynamic priority promotion scheduling* and the *laxity value for cloud service*.

III. Earliest Deadline Frist until Zero Laxity (EDZL) for Cloud Computing

A. Cloud Service Assumption

We make some assumptions and constrains in our cloud service model.

Assumption:

1) Tasks are preemptive: each task can perform content switching on the cloud platform.

2) Cloud platform consisted of homogeneous process units: The computing units have the same computing power, archtiecture, system software and system configuration.

We design our EDZL algorithm for cloud services based on these assumptions.

B. EDZL Algorithm in Cloud Environment

EDZL is a novel scheduling algorithm with a better scheduling capability than EDF and also has a lower context switching overhead than LLF. EDZL uses EDF as long as there is no urgent task in the cloud service tasks. When the laxity of a task becomes zero, a current task with positive laxity and has the latest deadline among all current tasks is preempted by this task. If the system is overloaded the scheduler will discard this task. When ties occur among current tasks, our algorithm chooses the task with the smallest computation time. It is clear to see that priority of a task in EDZL is not fixed.

Figure 1 shows an example by using the EDZL algorithm. Given a set of cloud service tasks $\{\tau_1 = (2,3), \tau_2 = (2,3), \tau_3 = (2,3)\}$, the task set needs more than two process units in the cloud service when the scheduler employs EDF (See Figure 1.a). This task set is schedulable using EDZL and the numbers of process units needed is equal to two. The details of EDZL scheduling algorithm are presented in [8].

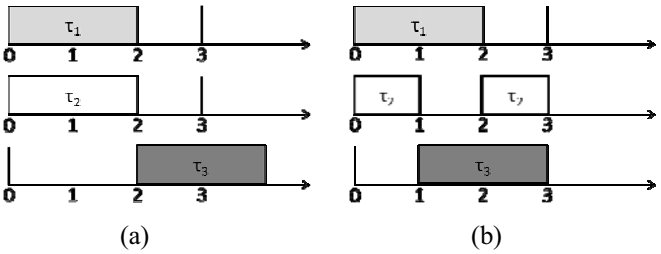

Figure 1. (a) EDF schedule and (b) EDZL schedule with process units = 2

C. Properties of EDZL Algorithm

We describe some general properties of EDZL in cloud service systems.

Definition 1. A schedule of a task set Γ by using the EDZL algorithm is called an EDZL schedule.

EDZL algorithm never leaves any process unit that is rent by a user in an idle state if there are user tasks waiting to be executed.

Lemma 1. In an *m*-process unit rent by a user using the EDZL scheduling algorithm, if a job *J* misses its deadline at time *t*, the cloud side must have at least *m* +1 user services with zero-laxity simultaneously occurring at some time before *t*.

Note that, EDZL is a hybrid algorithm of EDF and LLF. In the following, we introduce some properties related to EDF and EDZL.

Definition 2. *Priority promotion* in EDZL schedule. In an EDZL schedule, job J_i has a higher priority than job J_j, but $d_i > d_j$ is called priority promotion.

Lemma 2. An EDZL schedule is the same as an EDF schedule if and only if there is no priority promotion in the EDZL schedule.

Lemma 3. For multiple process units, if a task set Γ can be scheduled by the EDF algorithm, Γ can be scheduled by the EDZL algorithm.

The details of above lemma proof are shown in [8].

IV. FUTUREWORK: SCHEDULABILITY OF EDZL ON CLOUD ENVIRONMENT

In this section we state the schedulability of EDZL on *m*-processors system by considering task sets with different largest utilization u_{max}. We try to apply these results to cloud scheduling in the future.

For the general case, the number of tasks is not related to the number of processors in EZDL algorithm.

Theorem 1. A set of periodic task is schedulable by EDZL scheduling algorithm on *m* processor systems, if the total utilization is less than or equal to $(m+1)/2$

Theorem 2. The utilization upper bound of EDZL is $m(1-1/e)$

There are some differences in the task set in the cloud platform. In the future, we will prove the EDZL algorithm that still can work on a cloud service task and we will try to give the utilization bound of EDZL for a certain amount of cloud process units. Then, we will employ the EDZL scheduling method onto the LCPS cloud platform. The system architecture is shown in Figure 2.

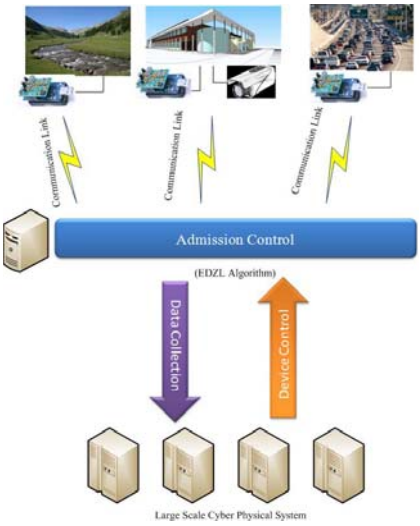

Figure 2. System architecture

There are many end sensor devices in the cyber physical system. The sensed data size of cyber physical devices is larger than the LCPS capacity, although the cyber physical system is deployed on a cloud platform. Thus, the admission control is an important component in the LCPS architecture. We use the EDZL algorithm to be our admission control algorithm in LCPS. The LCPS schedules tasks of information collection and device control on the cloud platform. According to the laxity values of sensor jobs, the EDZL algorithm gives every task a dynamic priority promotion. We can assign different task periods to different cyber physical applications or cyber physical groups. Different task periods and sensing information delivering time determine different job's laxity values. Hence, we use EDZL to control these cyber physical devices for information delivering in LCPS. The EDZL scheduler can efficiently arrange jobs in the cyber physical system for ensuring sensor jobs do not miss deadline.

V. CONCLUSION

In this paper, we try to apply the EDZL scheduling algorithm to the cloud service platform. EDZL integrates EDF and LLA, and has a low context switching overhead and a low deadline miss ratio. EDZL is appropriate for cloud on-line scheduling algorithm. The flexibility of priority promotion allows system resources to be better utilized for meeting deadlines. The EDZL can ensure the quality of service in the cloud platform.

We show that EDZL scheduling algorithm is suitable to be employed on the cloud platform and plan to study the EDZL utilization bound for the cloud service platform in the future. The utilization bound of schedulability of cloud service task set can suggest users to rent minimum process units from cloud computing supplier.

REFERENCES

[1] Michael Armbrust , Armando Fox , Rean Griffith , Anthony D. Joseph , Randy Katz , Andy Konwinski , Gunho Lee , David Patterson , Ariel Rabkin , Ion Stoica , Matei Zaharia, "A view of cloud computing," Comunications of the ACM, Vol. 53, No. 4, April 2010.

[2] Ioannis A. Moschakis, Helen D. Karatza, "Evaluation of gang scheduling performance and cost in a cloud computing system," The Jounral of Supercomputing, 16 Sep. 2010.

[3] Wei-Tek Tsai, Qihong Shao, Xin Sun, Jay Elston, "Real-Time Service-Oriented Cloud Computing," in IEEE 6th World Congress on Services, Miami, FL, 2010, pp. 473-478.

[4] Shuo Liu, Gang Quan, Shangping Ren, "On-line Scheduling of Real-time Services for Cloud Computing," in IEEE 6th World Congress on Services, Miami, FL, 2010, pp. 459-464

[5] Shuo Liu, Gang Quan, Shangping Ren, "On-line Preemptive Scheduling of Real-Time Services with Profit and Penalty," in Southeastcon, 2011 Proceedings of IEEE, Nashville, TN, 2011, pp. 287-292

[6] Jiahui Jin, Junzhou Luo, Aibo Song, Fang Dong, Runqun Xiong, "BAR: An Efficient Data Locality Driven Task Scheduling Algorithm for Cloud Computing," in 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Newport Beach, CA, 2011, pp. 295-304

[7] Lee, Suk Kyoon, "On-line Multiprocessor Scheduling Algorithms for Real-Time Tasks," TENCON'94. IEEE Region 10's Ninth Annual International Conference, (Aug. 1994) 607-611.

[8] Hsin-Wen Wei , Yi-Hsiung Chao , Shun-Shii Lin , Kwei-Jay Lin , Wei-Kuan Shih, "Current Results on EDZL Scheduling for Multiprocessor Real-Time Systems," in Proceedings of the 13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, p.120-130, August 21-24, 2007

[9] Yi-Hsiung Chao, Shun-Shii Lin, Kwei-Jay Lin, "Schedulability Issues for EDZL Scheduling on Real-Time Multiprocessor Systems", Journal Information Processing Letters, Vol 107, Issue 5, Aug., 2008.