Journal of Systems Architecture 61 (2015) 210–226

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

Improving schedulability and energy efficiency for window-constrained real-time systems with reliability requirement

Linwei Niu [a,*], Jia Xu [b]

[a] California State University Bakersfield, Bakersfield, CA 93311, USA
[b] York University, Toronto, Ontario M3J1P3, Canada

ARTICLE INFO

Article history:
Received 1 October 2014
Received in revised form 11 March 2015
Accepted 20 April 2015
Available online 19 May 2015

Keywords:
Real-time embedded systems
Window-constraint
EDF
Window-pattern
Reliability

ABSTRACT

For real-time embedded systems, schedulability, energy efficiency, Quality of Service (QoS), and reliability are four highly co-related important design concerns. In this paper, we explore combining these four dimensions of design issues to achieve better schedulability and energy efficiency for real-time systems while satisfying the QoS and reliability requirements. The QoS requirements are deterministically quantified with the *window-constraints*, which require that at least m out of each non-overlapped window of k consecutive jobs of a real-time task meet their deadlines. Our contributions mainly consists of two parts: in the first part, we propose two off-line approaches for the management of appropriate mandatory/optional job partitioning with the purpose of improving the schedulability for real-time systems with window-constraints; based on it, in the second part, we proposed advanced techniques to reduce the energy consumption of the real-time systems while satisfying the reliability requirements. The results of extensive experiments demonstrate that our proposed techniques significantly outperform previous approaches in both schedulability and energy efficiency for real-time embedded systems while satisfying the window-constraints and reliability requirements.

1. Introduction

In traditional hard real-time embedded systems, all task instances are required to meet their deadlines and any deadline miss will cause the entire application or system to fail. However, few real-time applications are truly *hard*. For example, many real-time applications, such as multimedia processing and real-time communication systems, can often tolerate occasional deadline misses. Some other applications may have soft deadlines where tasks that do not finish by their deadlines can still be completed with a reduced value [26] or they can simply be dropped provided that the user's perceived Quality of Service (QoS) is satisfied.

To quantify the QoS requirements, some statistic information such as the average deadline miss rate is commonly used. Although the statistical deadline miss rate can ensure the QoS in a probabilistic manner, this metric can be problematic for some real-time applications because even a very low overall miss rate tolerance cannot prevent a large number of deadline misses from occurring consecutively in a short period of time, which could generate undesirable results. For example, even under the 5% average deadline miss rate, it is possible that 50 jobs out of 1000 jobs miss their deadlines consecutively while still satisfying the loss rate requirement. However, such kind of result might not be acceptable for certain applications in terms of satisfactory QoS levels or information integrity. To avoid possible severe consequences, one can always set the miss rate to be 0%, *i.e.*, treat the system as a hard real-time system. The problem, however, is that when the system is overloaded,[1] the approaches for hard real-time systems are no longer valid as deadline missing will become inevitable in such kind of scenarios. As a result, critical information that cannot be reconstructed may be lost, thus the service quality can be severely degraded from the user's perspective.

The concept of *weakly hard real-time system* is more suitable for modeling such kind of applications. In the weakly hard real-time model, the systems should not only support an overall guarantee of the QoS statistically, but also be able to provide a lower bounded, predictable level of QoS in each specified time interval. Two well known weakly hard real-time models are the (m,k)-model [12] and the *window-constrained* model [36]. The

[1] This is a common case when additional constraints such as energy constraints are imposed on the systems, as conserving energy often requires reducing the processor voltage/frequencies, which will increase the execution times of the tasks and thus easily cause the systems to be "overloaded".

* Corresponding author. Tel.: +1 6613322125.
E-mail addresses: lniu@csub.edu (L. Niu), jxu@cse.yorku.ca (J. Xu).

http://dx.doi.org/10.1016/j.sysarc.2015.04.006
1383-7621/

(m,k)-model requires that m jobs out of any *sliding* window of k consecutive jobs of the task meet their deadlines, whereas the *window-constrained* model requires that m jobs out of each *fixed* and *nonoverlapped* window of k consecutive jobs meet their deadlines. For example, for the particular case above, one only needs to specify m to be 19 and k to be 20, which allows at most one deadline miss out of each window of 20 jobs. Obviously if such kind of "deterministic" QoS constraint is satisfied, the required average deadline miss rate is guaranteed to be satisfied as well. And more importantly, the undesirable results above could be avoided.

Compared with *window-constrained* model, one problem of the (m,k)-model is that it adopts very strong constraints which may over-specify the QoS requirements for some applications. For example, some packet streams over the network can be logically divided into segments, and in each segment a minimum number of jobs must be completed in time. In this case, the (m,k)-model imposes a minimum requirement of job completions on each and every overlapped window of k jobs, regardless of whether these jobs in different windows are logically related or not, which can be over stronger than necessary [1]. The *window-constrained* model, on the other hand, is more appropriate for such kind of applications. In [1], Mok et al. proved that the general *window-constrained* scheduling problem for arbitrary service times and request periods is NP-hard in the strong sense. So far, deterministic assurance with this model can only be guaranteed for a very limited range of systems, such as those with all tasks having the same request periods and unit size execution times [1]. Note that the scheduling problem with (m,k)-constraint is also NP-hard in the strong sense [32]. To guarantee the (m,k)-constraints, Ramanathan et al. [33] proposed to partition the jobs into *mandatory* and *optional* jobs. The mandatory jobs are jobs that must meet their deadlines in order to satisfy the (m,k)-constraints, while the optional jobs can be executed to further improve the QoS or simply be dropped. In [28] a similar partitioning strategy was adopted to deal with the problem of scheduling with *window-constraints* guarantee for task sets with arbitrary execution times and periods.

For real-time embedded systems, energy minimization has also come to be recognized as one of the primary design goals. Dynamic voltage/frequency scaling (DVFS), *e.g.* [37], which dynamically varies the supply voltage and frequency of the processing unit, has proven to be an effective way to reduce energy consumption. Moreover, since energy consumption is a convex function of the processor speed, which is proportional to the supply voltage [7], the energy efficiency of periodic real-time embedded systems is closely related to the schedulability of the system. In other words, if the schedulability of the periodic real-time task sets can be improved, the running speeds of the processor to execute the jobs can be lowered correspondingly and thus the energy consumption can be reduced.

Recently, reliability and fault tolerance have also been studied for the design of dependable computer systems. Generally, computing system faults can be classified into transient and permanent faults [15]. Since transient faults occur more frequently than permanent faults [6], in this paper we focus our research on dealing with the transient faults. Due to the effects of hardware defects, electromagnetic interferences and/or cosmic ray radiations, transient faults may occur at anytime during the execution of real-time jobs [46]. On the other hand, it has been shown that DVFS has a direct and negative effect on system reliability due to increased number of transient faults at lower supply voltages [9,34,46]. Therefore, for safety–critical real-time systems such as avionics and industrial controls, catastrophical consequences may occur if system faults are not handled in a timely manner.

In this paper, we explore how to improve the schedulability and energy efficiency for window-constrained real-time systems while satisfying the system reliability requirements. Since compared to the fixed-priority scheme, the earliest deadline first (EDF) scheme allows the task set to be executed under higher CPU utilizations and thus has better potential for improving energy efficiency through voltage scaling, we assume that the real-time tasks are scheduled according to the EDF scheduling policy.

Our contributions in this paper consists of two parts. In the first part (Section 4), we propose two off-line approaches for the management of appropriate mandatory/optional job partitioning with the purpose of improving the schedulability of real-time systems with given QoS constraints. Specifically, the first approach tries to enhance the previous work in [33] in which the mandatory jobs of each task are evenly distributed. Due to the inherent properties of some real world applications (for example, the task periods of some practical systems are often close to harmonic), strictly evenly distributed patterns might not always yield good results. To take into account these scenarios, we also propose a second approach in which the mandatory jobs are determined more flexibly based on the simulated annealing (SA) algorithm. In the second part (Section 5), by considering the system reliability under the context of window-constraints, we propose new approaches to reduce the energy consumption of the real-time task sets systematically. Through our extensive experiments, the results show that our proposed approaches can significantly improve the schedulability and energy efficiency for real-time systems while satisfying the window-constraints and reliability requirements.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 presents the system models. Section 4 introduces our approaches in enhancing the previous mandatory/optional job partitioning approaches for improving the schedulability of the systems. Section 5 introduces our approaches for reducing the systems energy under the reliability requirement. The effectiveness of our approach is demonstrated using experimental results in Section 6. In Section 7, we offer some conclusions.

2. Related work

Due to its intuitiveness and capability of capturing not only statistical but also deterministic QoS requirements, the weakly hard real-time models have been widely studied. West et al. [36] proposed a dynamic window-constrained scheduling algorithm for providing window-constrained service to real-time streams consisting of non-preemptable unit size packets. In [1], Mok et al. proved that with *window-constraints*, the guaranteed scheduling problem is NP-hard in the strong sense. They also proposed a *P-fairness* scheduling algorithm for real-time systems with *dynamic window-constraints*. However, the deterministic assurance with this approach can be guaranteed only for tasks with the same unit size execution times and with resources utilization no more than 50%. In [39], Zhang et al. introduced a virtual deadline scheduler to schedule task sets with higher resource utilizations. However, it can only schedule m out of k consecutive jobs by their "virtual deadlines" which are relaxed from the real deadlines. As a result, it cannot guarantee that at least m jobs out of k jobs meet their real deadlines.

When considering (m,k)-constraints, Ramanathan et al. [33] proposed to partition the jobs into *mandatory* and *optional* jobs. The mandatory jobs are the jobs that must meet their deadlines in order to satisfy the (m,k)-constraints, while the optional jobs can be executed to further improve the quality of the service or simply be dropped to save computing resources. In [28], the same