Graceful Overload Management in Firm Real-Time Systems

Maryline Chetto*
IRCCyN (Institut de Recherche en Communications et Cybernétique de Nantes), University of Nantes, Nantes, France

Abstract

Real-time embedded systems have to provide the highest Quality of Service despite possible processing overloads. In such systems, programs are characterized by upper bounds on finishing times and the QoS is assessed by the ratio of successful deadlines. In this article, we deal with this issue. We focus on single processor architectures in the framework of firm real-time applications that accept deadline missing under some specified limits. Tasks are assumed to be periodic. We present a novel model for tasks which is called BGW model. It is drawn from two approaches respectively known as the skip-over model and the Deadline Mechanism. We propose specific dynamic priority schedulers based on EDF (Earliest Deadline First) for BGW task sets and we briefly report the results of a simulation study.

Introduction to Real-Time systems

In real-time environments, tasks have to complete by their deadlines. As we restrict our attention to mono-processor systems, a scheduling algorithm aims to determine which task is to execute on the processor [1,2]. On-line scheduling algorithms have been designed under non-overloaded conditions. We say that a real-time system is overloaded when it is impossible to schedule it on the processor such that all the tasks meet their deadlines. Classical online priority driven schedulers perform poorly as compared to offline clairvoyant ones in overloaded conditions.

In that article, we consider the firm real-time systems where timeliness properties normally guaranteed may admit irregularities in occasional situations mainly caused by prohibitive execution times. A framework should allow a real-time system to gracefully adapt to these exceptions that manifest as deadline missing. The approach presented in this article is a new task model, namely BGW (Black Grey White), developed as a means to provide flexibility in scheduling time-constrained tasks when the processor is overloaded [3].

Scheduling in under-loaded real-time systems

A number of authors have studied the problem of devising algorithms for scheduling time critical jobs on a single processor computing system with no energy consideration. The most popular online scheduling algorithm was introduced by Liu and Layland [4]. According to Earliest Deadline First (EDF) which is preemptive and dynamic priority driven, a ready job with the earliest deadline is executed first. Dertouzos [5] proved that EDF is optimal among all scheduling algorithms on a uniprocessor machine. Consequently, if a set of jobs cannot be scheduled by EDF, then this set cannot be scheduled by any other algorithm. Liu and Layland stated a very simple necessary and sufficient schedulability condition for EDF under the assumption that jobs are the instances of periodic tasks with relative deadlines equal to periods. The EDF strategy is consequently a very desirable approach for scheduling independent jobs preemptively when there is no energy limitation and no processing overload. A survey on EDF scheduling can be found in the work of Buttazzo [6].

Overload management

When the timing constraints cannot be met with computation-quality, one way of maintaining an acceptable Quality of Service in overloaded conditions is to trade computation-quality for timeliness. The Deadline Mechanism offers such possibility with software redundancy [7,8]. In the Deadline Mechanism, two versions of programs are provided for each task: primary and alternate. The primary version is the normal program that produces good quality results whereas the alternate version produces less precise results with very low processor demand. The Deadline Mechanism is integrated in the BGW mechanism.

Another way to reduce the load in transient overloaded conditions is by discarding some jobs that cannot be completed in time under certain conditions. The effectiveness of the so-called Skip-Over approach has been demonstrated especially in multimedia applications and is integrated in the BGW model. In the Skip-Over model, every periodic task $t_i$ is characterized by its basic timing parameters such as deadline and period and a skip parameter $s$ [9]. This parameter represents tolerance for the periodic task to miss some of its deadlines. The distance between two consecutive skips must be at least $s$ periods. When $s$ equals to infinity, no skips are allowed and $t_i$ is called a hard periodic task.

Motivations for the BGW model

The design of the BGW model is based on the following assumptions. First, we do not assume a priori knowledge of effective processing times but worst case processing times needed by the application. It can be observed that the amount of processing time needed has high variations as a result of changes in multimedia contents for example. Second, we assume that the application tolerates occasional deadline violations. The BGW mechanism ensures that the tasks execute timely by enforcing certain timing distance between two consecutive successful executions. Third, the application gracefully adapts to overloads by reducing its processing requirements. There are many applications where tasks have stochastic processing times. Such tasks can undergo changes in period too due to application dependencies which result in processing overload. The BGW model provides a systematic approach to specify QoS (Quality of Service) requirements of firm real-time systems wherein timing requirements can be violated during exceptions up to an acceptable known bound.

*Corresponding author: Maryline Chetto, IRCCyN (Institut de Recherche en Communications et Cybernétique de Nantes), University of Nantes, Nantes, France, Tel: +33 (0) 6 10 20 35 94; E-mail: maryline.chetto@univ-nantes.fr

Received October 24, 2015; Accepted October 30, 2015; Published November 30, 2015


Copyright: © 2015 Chetto M. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.
Description of the BGW model

The BGW model integrates two approaches:

- The Deadline Mechanism where each real-time periodic task uses two independent versions for the purpose of meeting timing constraints with variable QoS. The BGW model is similar to the Deadline Mechanism, in that both consider any task to be divided into a primary part and an alternate part. However, the Deadline Mechanism imposes to execute at least one of the two algorithms.

- The Skip-Over model where each real-time periodic task has a skip parameter.

Formally, a BGW periodic task set $\tau$ is composed of $n$ tasks where each task $\tau_i$ is characterized by five timing parameters: $r, C_\alpha, C_p, D$, and $T_i$. $C_\alpha$ refers to the worst case execution time of the alternate respectively the primary, with $C_\alpha \leq C_p$, $D$ respectively $T_i$ refers to the deadline respectively the period of task $\tau_i$.

Two QoS parameters are further defined: $n_i$ and $l_i$. At least one primary version over $n_i$ successive requests has to execute timely, as well as at least one alternate version over $l_i$ successive requests. The term "distance" can be defined as a number of requests. The parameter $n_i$ expresses the maximum distance allowed between two consecutive successful executions of the primary version. The parameter $l_i$ is the maximum distance allowed between two consecutive successful executions of a job (whatever primary or alternate).

The motivation for the BGW model is to guarantee the stability of computer-based control systems. As the processor may sometimes be overloaded, this model will permit us to determine online the algorithm (primary or alternate) that has to be executed for every task. Control may be realized through two different algorithms: one which returns very precise results but requires high processor utilization and recover processing time since the result of the alternate becomes no longer necessary. In this strategy, the scheduler has to suspend any running primary whenever an alternate requires to be executed so as to meet its deadline.

A scheduling framework for the BGW model is specified by a combination of:

- a scheduling rule for the primaries,
- a scheduling rule for the alternates,
- a scheduling rule for white jobs
- and a hierarchy between the three schedulers.

A job can be in the first list only if it is a black job, in the two first lists if it is a grey job and in the third list if it is a white job. Moreover, this framework is hierarchical in that sense that the first two schedulers must be organized under either the First Chance approach or the Last Chance approach. Finally, the list of white jobs has to be served when no jobs are present in the first two lists i.e. as a background scheduler.

Performance Evaluation

Several combinations were studied in simulations. We have implemented the following scheduling strategies: EDF, LCJ and LCP. Under EDF, every task respects the classical Liu and Layland model without QoS parameters. Under LCP, primary versions of grey jobs and primary versions of white jobs are executed as soon as possible. In contrast, primary versions of black jobs and alternate versions of grey jobs are scheduled as late as possible using the Last Chance technique. Under LCJ, primary versions of black jobs and alternate versions of grey jobs are scheduled as soon as possible and alternate versions of white jobs are executed in the remaining idle times. The simulations show that LCP outperforms all other schedulers regarding the metric NPP and LCJ outperforms all other schedulers regarding the metric NP. The ratio of deadline misses for the EDF scheduler is never higher.
than the ratios for LCP and LCJ. This outlines why both the BGW model and specific scheduling strategies improve significantly the resulting Quality of Service under transient processor overload.

Conclusion

Many applications with content-dependent execution times such as video-processing tolerate bounded non-satisfaction of their timeliness properties. The computing system considered here uses a single-processor machine to run multiple software jobs issued from periodic tasks. We described a new approach for modeling Quality of Service requirements of periodic task systems which may be the object of both transient faults and processor overloads. Several scheduling schemes for the so-called BGW-task model have been implemented and compared. The experiments brought to light the usefulness of this new task model for managing overload in firm real-time systems in comparison to the classical task model.

References