Dynamic Sporadic Server

Moiz Zaheer Malik
B.Eng. Electronic Engineering
Hochschule Hamm-Lippstadt
Lippstadt, Germany
moiz-zaheer.malik@stud.hshl.de

***Abstract*—In real-time systems, where tasks have different levels of critical importance, it is essential to serve aperiodic (irregular, event-driven) tasks while ensuring that the deadlines of high-priority periodic tasks are not violated. The Dynamic Sporadic Server (DSS) is a scheduling method designed for Earliest Deadline First (EDF) systems that addresses this problem.**

DSS is defined by a period T_s and a budget C_s, but unlike traditional sporadic servers, it does not restore the full budget at every period. Instead, when an aperiodic task arrives, DSS assigns it a deadline and restores only the amount of budget that was actually used. This approach allows the processor to reach full (100 percent) utilization while ensuring that all deadlines are still met. DSS improves the response time of aperiodic tasks without compromising the guarantees of periodic tasks.

Originally introduced by Spuri and Buttazzo[3], DSS has been widely studied for its efficiency in managing mixed task sets under EDF scheduling. This report reviews the theory behind DSS, describes its operation (including budget management), highlights its advantages over traditional methods, and discusses potential application areas.

I. Introduction

Real-time systems often mix periodic tasks (recurring jobs with fixed periods and deadlines) and aperiodic/sporadic tasks (jobs which arrive unpredictably, e.g. interrupts or user requests). Ensuring that aperiodic tasks receive timely service without causing any periodic task to miss its deadline is a major challeng[1].

A common solution is to use a server abstraction a dedicated "server task" is given a share of the processor to handle aperiodic jobs. Examples include Polling Servers, Deferrable Servers, and Sporadic Servers. The Sporadic Server (SS), introduced by Sprunt et al[2], assigns a fixed priority (typically via Rate Monotonic) to the server and provides a budget C_s every period T_s. This allows aperiodic tasks to be executed with limited interference to hard real-time tasks. However, under static-priority scheduling the processor often cannot be fully utilized, unused server budget might be wasted or cause complex analysis.

With dynamic scheduling (EDF) one can achieve higher utilization. Dynamic priority servers extend the SS concept to EDF. In particular, Spuri and Buttazzo proposed the Dynamic Sporadic Server (DSS)[3]. DSS is characterized by a server period T_s and capacity C_s (the maximum aperiodic budget). Unlike SS, the DSS does not replenish C_s fully at each period start. Instead, the server's deadline and replenishment events are set dynamically whenever aperiodic work arrives or is consumed[3]. This allows DSS to adaptively use idle time for aperiodic tasks and to achieve 100 Percent CPU utilization if needed. The key property is that under the EDF schedule, the combined utilization of periodic tasks (U_p) plus the utilization of DSS ($U_s = C_s/T_s$) must satisfy

$$U_p + U_s \leq 1$$

In order to meet all deadlines[3]. This EDF schedulability bound (100 Percent utilization) for DSS is analogous to the standard EDF limit for purely periodic tasks, which means that DSS does not reduce the processor's capacity for periodic jobs. In contrast, a fixed-priority SS can only guarantee less than full utilization without more complex analysis[3].

II. Background: Periodic and Aperiodic Tasks; EDF

Periodic:A periodic task τ_i generates an infinite sequence of jobs; each job requires execution time C_i and must finish by a relative deadline D_i after its arrival. Commonly, $D_i = T_i$ (deadline equals period). The total utilization of periodic tasks is

$$U_p = \sum_i \left(\frac{C_i}{T_i}\right)$$

By the seminal result of Liu and Layland, EDF schedules any set of periodic tasks (with deadlines equal to periods) if and only if

$$U_p \leq 1$$

Sporadic:A sporadic task also has a minimum inter-arrival time (period) T_s and execution time C_s, but its jobs arrive irregularly (only the spacing is bounded). Sporadic tasks are often treated like a subset of periodic tasks where job activations are constrained by a minimum interval.

Aperiodic tasks: More general than sporadic, aperiodic tasks have no fixed period; they arrive unpredictably. Often aperiodic jobs have soft deadlines (misses are tolerable) or only request best-effort service. To quantify their demand, one may consider the average (or maximum) aperiodic load U_a. The challenge is to service aperiodic jobs promptly without compromising the hard deadlines of periodic tasks.

EDF scheduling: Under EDF, jobs are assigned dynamic priorities equal to their absolute deadlines (earliest deadline = highest priority). The EDF schedulability criterion for mixed periodic and aperiodic (when aperiodic are treated as sporadic with deadlines) remains

$$U_p + U_a \leq 1$$

(assuming job deadlines do not exceed their periods)[3]. However, simply queueing all aperiodic jobs under EDF can cause spikes in demand that violate periodic tasks' guarantees. Therefore, server algorithms are used to police how much CPU the aperiodic workload can use, effectively reserving bandwidth

$$U_s = \frac{C_s}{T_s}$$

for aperiodic service[3]. DSS is one such server, optimized for EDF-based systems.

III. Scheduling Theory Behind DSS

In an EDF-scheduled system, tasks are selected by earliest absolute deadline. A server can be treated as a periodic task with period $T_s$and execution time C_s; however, DSS refines this by using dynamic deadlines. Spuri and Buttazzo showed that DSS behaves (in the worst case) like a periodic EDF task with execution C_s every T_s[3]. This is the same bound as for pure EDF, so DSS preserves full utilization.

Whenever the server is active (has pending aperiodic requests), it is assigned the earliest deadline among all jobs, so that it will not preempt any higher-priority (earlier-deadline) periodic job. This ensures that, from the perspective of EDF analysis, the server simply looks like a periodic task with relative deadline T_s The rigorous proof given in [3] confirms that as long as $U_p + U_s \leq 1$, no deadline misses occur. In summary, scheduling theory tells us that DSS achieves the EDF utilization bound and effectively isolates periodic and aperiodic loads. DSS thus extends the classic Sporadic Server theory into the dynamic-priority (EDF) domain [3].

IV.

V.

VI.

VII. Conclusion

References

[1] Philip A. Laplante. *Real-Time Systems Design and Analysis: Tools for the Practitioner*. 4th. Wiley-IEEE Press, 2011.

[2] Barry Sprunt, Lui Sha, and John P. Lehoczky. “Aperiodic Task Scheduling for Hard-Real-Time Systems”. In: *The Journal of Real-Time Systems* 1.1 (1989), pp. 27–60.

[3] Marco Spuri and Giorgio C. Buttazzo. “Efficient Aperiodic Service under Earliest Deadline Scheduling”. In: *Proceedings of the IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 1994, pp. 2–11.