Guaranteed dynamic priority assignment scheme for streams with (*m*, *k*)-firm deadlines

H. Cho, Y. Chung and D. Park

A guaranteed dynamic priority assignment scheme for multiple real-time streams with (*m*, *k*)-firm deadlines is presented. Analytical and experimental studies show that the proposed scheme provides assurance of timeliness performance and relatively high quality of service compared to existing schemes.

Introduction: A weakly–hard real-time system can afford to miss some deadlines during any time window, i.e. the occasional loss of some deadlines is usually acceptable [2–4]. One example is real-time video stream transmission, where a source generates a stream of video frames, which are transmitted and played back at the destination. Each frame has its own deadline by which it must arrive at the destination. In this system, a few occasional missed deadlines do not cause significant degradation in video quality, provided that there are only a limited number of consecutive deadline misses. To precisely specify the weakly–hard real-time requirement, Hamdaoui and Ramanathan have defined an (*m*, *k*)-*firm deadline* as when the quality of service is tolerable, provided at least *m* frames in any window of *k* consecutive frames meet their deadlines [1]. A stream that violates its own (*m*, *k*)-firm deadline, i.e. there are fewer than *m* occurrences of deadline satisfaction in a window of *k* consecutive frames, introduces a dynamic failure. Thus, the probability of a dynamic failure is used to measure how often the stream provides lower quality of service than is required.

For dealing with the problem of scheduling multiple real-time streams constrained by (*m*, *k*)-firm deadlines, Hamdaoui and Ramanathan proposed a dynamic priority assignment scheme, the distance-based priority scheme (DBP), that assigns priority based on the recent history of streams' dynamic failure occurrences. More specifically, DBP assigns a priority according to the minimum number of consecutive deadline-misses that are required for the stream to fall into the dynamic failure state. A higher priority is given to a stream with a shorter distance to its dynamic failure state. However in [2] it was pointed out that DBP, which is a best-effort online scheduling algorithm, has two major restrictions: 1. it provides non-guaranteed timeliness performance, and 2. it only considers homogeneous stream sets with the same execution and inter-arrival times, etc. In addition, the objective of stream scheduling is not only to provide guaranteed performance by avoiding dynamic failures of streams constrained by (*m*,*k*)-firm deadlines, but also to provide the highest possible quality of service, i.e. as many occurrences of deadline satisfaction of the stream as possible. To address these issues, we propose the guaranteed dynamic priority assignment scheme (GDPA) that schedules multiple streams constrained by (*m*, *k*)-firm deadlines. GDPA is designed to: 1. provide guaranteed real-time performance for multiple streams with (*m*, *k*)-firm deadlines when the system is under-loaded; 2. reduce the probability of dynamic failures, and 3. maximise the probability of deadline satisfactions.

Proposed scheme: We consider an application that consists of a set of streams, denoted $\{S_1, S_2, \ldots, S_n\}$. Each S_i has a number of frames which are released periodically or sporadically with a known inter-arrival time. The *j*th frame of stream S_i is denoted as $J_{i,j}$. The inter-arrival time of S_i is denoted as p_i and the worst-case service time of S_i is denoted as c_i. Each S_i has its own (m_i, k_i)-firm deadline constraint. Thus, each S_i is characterised by (c_i, p_i, m_i, k_i). We assume that each frame's deadline is the same as its period for the sake of simplicity, but we emphasise that GDPA is also designed to support frames where deadlines and periods differ.

A high-level description of GDPA is shown in Fig. 1. GDPA is invoked at both events of frame arrival and service completion. The ComputeDistance (ζ) function in line 2 calculates each stream's distance to its dynamic failure state. For example, suppose that the *i*th stream has (1, 3)-firm deadlines. When the stream misses, meets, and misses its deadline sequentially, the state of the stream is represented by (mMm), where m and M denote miss and meet, respectively, as in [1]. In this case, a dynamic failure will occur if two consecutive deadline-misses follow, which implies that the current distance of the stream is two. In line 4, GDPA sorts frames in order of the shortest distances first, which enables early processing of streams closer to a dynamic failure state. Each frame from head-to-tail in the sorted queue σ_1 is inserted into another temporal queue σ_2 in order of earliest deadline first in line 7. When inserting the frame in line 8, GDPA checks the feasibility of all frames in σ_2. If frame insertion results in infeasibility, the frame is removed. Note that all frames are said to be *feasible* when they all satisfy their deadlines. GDPA selects a frame at the head of σ_2.

```
Input:      Ready queue ζ that contains frames in ready state
Output:     Selected frame for service
Variables:  Temporal queue σ1, σ2={}

1:   For each frame Jk ∈ ζ
2:       ComputeDistance( ζ );
3:
4:   σ1 = SortByShortestDistanceFirst( ζ );
5:
6:   For each frame Jk ∈ σ1 from head to tail
7:       insertByEarliestDeadlineFirst( Jk, σ2 );
8:       If ( !feasible(σ2) ) remove( Jk, σ2 );
9:
10:  Return headOf( σ2 );
```

Fig. 1 *GDPA*

GDPA has a unique feature that it has no dynamic failure when the system is under-loaded, i.e. when the total utilisation demand of streams ($= \Sigma c_i/p_i$, *for all i*) is less than one. This is straightforward, since GDPA behaves exactly like earliest deadline first (EDF) that is known to be an optimal real-time scheduling algorithm satisfying all streams' deadlines when the system is under-loaded [5]. It is clear that satisfaction of all deadlines has no dynamic failure. Besides, GDPA considers both distances and deadlines of streams in order to reduce the probability of dynamic failures while maximising the probability of deadline satisfactions.

Experimental results: To validate the above-mentioned features and evaluate performance, simulation-based experimental studies were conducted. EDF and DBP were selected as counterparts to GDPA. Note that in all schemes, a frame that already missed its deadline is aborted. While varying the total utilisation demand from 0.6 to 1.8, we generated streams with c_i and p_i, both of which were randomly generated with a uniform distribution in the range $[1, 0.8p_i]$ and [2, 30], respectively. Three different (*m*, *k*)-firm deadlines including (2,3), (2,4), and (1,2) were randomly assigned to the generated streams. While varying the total utilisation demand from 0.6 to 1.8, the probability of deadline satisfaction (PDS) and probability of dynamic failure (PDF) were measured. In Figs. 2 and 3, the error bar around each data point represents a 95% confidence interval. Fig. 2 shows that both EDF and GDPA support 100% PDS when the total utilisation is less than or equal to one, i.e. when the system is under-loaded. DBP, however, shows the lowest PDS for all measured total utilisation demands. Fig. 3 shows that both EDF and GDPA provide 0% PDF when the total utilisation is less than or equal to one. On the contrary, DBP does not always provide 0% PDF, even when the system is under-loaded. In terms of PDF, GDPA is fairly comparable to DBP for all measured total utilisation demands.

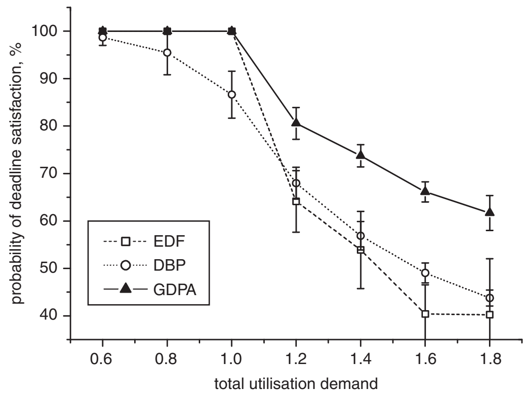

Fig. 2 *Probability of deadline satisfaction with varying total utilisation demand*

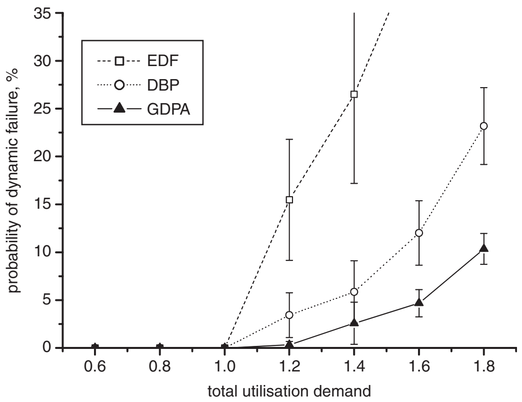

Fig. 3 *Probability of dynamic failures with varying total utilisation demand*

Conclusion: We have presented GDPA, which is a guaranteed dynamic priority assignment scheme for multiple streams with (m, k)-firm deadlines. Analytical and experimental studies established that GDPA provides assurance of no dynamic failure in an under-loaded system, and reduces the probability of both dynamic failures and deadline-misses. Although the computational complexity of GDPA is $O(n^2)$, where n is the number of streams, which is a little higher than that of EDF and DBP, this cost is justifiable for multimedia streaming applications where inter-arrival times of frames are of the order of milliseconds.

Acknowledgment: This work was supported by a Korea University grant.

1 September 2009
doi: 10.1049/el.2010.2497

H. Cho, Y. Chung and D. Park (*Department of Computer and Information Science, Korea University, Jochiwon-eup, Yeongi-gun, Chungnam 339-700, South Korea*)

E-mail: raycho@korea.ac.kr

References

1 Hamdaoui, M., and Ramanathan, P.: 'A dynamic priority assignment technique for streams with (m,k)-firm deadlines', *IEEE Trans. Comput.*, 1995, **44**, (12), pp. 1443–1451
2 Bernat, G., and Cayssials, R.: 'Guaranteed on-line weakly–hard real-time systems'. IEEE Real-Time System Symp., 2001
3 Flavia, F., Ning, J., Simonot-Lion, F., and YeQiong, S.: 'Optimal on-line (m,k)-firm constraint assignment for real-time control tasks based on plant state information'. IEEE Int. Conf. on Emerging Technologies and Factory Automation, 2008, pp. 908–915
4 Wu, T., and Jin, S.: 'Weakly hard real-time scheduling algorithm for multimedia embedded system on multiprocessor platform'. IEEE Int. Conf. on Ubi-Media Computing, 2008
5 Liu, C.L., and Layland, J.W.: 'Scheduling algorithms for multiprogramming in a hard-real-time environment', *J. Assoc. Comput. Mach.*, 1973, **20**, (1)