Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

An integrated DBP for streams with (*m*, *k*)-firm real-time guarantee*

WANG Zhi (王 智)†, CHEN Ji-ming (陈积明), SUN You-xian (孙优贤)

(*National Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China*)

†E-mail: wangzhi@iipc.zju.edu.cn

Received Nov. 31, 2003; revision accepted Feb. 23, 2004

Abstract: (m, k)-firm real-time or weakly hard real-time (WHRT) guarantee is becoming attractive as it closes the gap between hard and soft (or probabilistic) real-time guarantee, and enables finer granularity of real-time QoS through adjusting m and k. For multiple streams with (m, k)-firm constraint sharing a single server, an on-line priority assignment policy based on the most recent k-length history of each stream called distance based priority (DBP) has been proposed to assign priority. In case of priority equality among these head-of-queue instances, Earliest Deadline First (EDF) is used. Under the context of WHRT schedule theory, DBP is the most popular, gets much attention and has many applications due to its straightforward priority assignment policy and easy implementation. However, DBP combined with EDF cannot always provide good performance, mainly because the initial DBP does not underline the rich information on deadline met/missed distribution, specially streams in various failure states which will travel different distances to restore success states. Considering how to effectively restore the success state of each individual stream from a failure state, an integrated DBP utilizing deadline met/missed distribution is proposed in this paper. Simulation results validated the performance improvement of this proposal.

Key words: (m, k)-firm, Weakly hard real-time, Real-time schedule, DBP, Quality of service
Document code: A **CLC number:** TP393

INTRODUCTION

Traditionally, in real-time schedule theory, real-time requirements are classified into two types, hard real-time (HRT) and soft real-time (SRT) (Giorgio, 1997). HRT requirement is mainly utilized in temporal safety-critical applications on the temporal aspect of processing a task. For applications with HRT requirement, such as process control or manufacture automation, deadline miss is not tolerated, i.e., each task of an HRT application must meet its deadline, and otherwise it will cause serious damage to the environment. The analysis of such systems is normally performed under worst-case assumption (considering a task with its maximum execution time, and minimum arrival interval and minimum deadline) to estimate an upper bound response time of the task using service curve approaches (Cruz, 1991) or classical worst-case response time analysis (Lehoczky, 1990). SRT requirement is normally utilized in most none temporal safety-critical applications, where some occasional deadline misses are permitted. However, the term "occasional" in SRT system is not precise; we often specify a probability to meet deadline requirement. For example, the statistical real-time

* Project supported by the National Natural Science Foundation of China (No. 60203030) and Advanced Research Program of France-China (Nos. PRA SI01-04, PRA SI03-02)

channel proposed by Chou and Shin (1997), is essentially a kind of QoS which provides a probability of real-time requirement over a point-to-point channel. In general, the analysis of such systems is made using stochastic approaches and queuing theory (Takagi, 1990; Rom and Sidi, 1990).

These two classes of real-time requirement or real-time QoS are insufficient for appropriately describing most real-time applications. Example of this type of applications is multimedia systems, such as video-on-demand or streamed audio. It is important that information is received and processed at an almost constant rate, such as 30 frames per second for video information. However, some packets comprising of video frame can be lost, resulting in little or no noticeable degradation in the QoS at the receiver. More concretely, consider the MPEG transmission where some frames containing control information (for example synchronization) are inserted into the packet stream in regular way. The video packet can tolerate a certain deadline miss rate p if the deadline misses are uniformly distributed, whereas a large number of consecutive deadline misses cannot be acceptable (Atiquzzaman and Hassan, 2001; Zhang *et al.*, 2001). Therefore, we should provide adaptive mechanism for controlling deadline miss distribution, to achieve graceful performance degradation (Kuo and Mok, 1991; Lee *et al.*, 1996; Nakajima, 1998). For SRT requirement, stochastic approaches give only a probability of deadline misses and cannot guarantee that these deadlines are missed in proper manner to achieve good behavior of real-time system. Even the popular statistical real-time channel (Chou and Shin, 1997) cannot deal with the problem. For example both "one deadline missed every 10 instances of the task" and "100 deadlines missed followed by 900 deadlines met" meet the requirement of statistical real-time channel "less than 10% of deadlines missed", but these two cases are obviously not the same. On the other hand, HRT requirement makes stringent assumption, where deadlines of all instances must be met. While there are still some pessimistic factors in HRT requirement, since having possibility to miss some of them adequately does not necessarily lead to system failure. In fact, take computer automatic control system as an example, where sampling (or task generating) period is regarded as deadline, and missing some of them can be tolerated (Ramamritham, 1996). Moreover when we consider a distributed system, take together the worst-case condition (task execution and message transmission) and hard deadline may be unfeasible for a large set of supporting systems (in terms of available computer power and network bandwidth).

In order to go around the problem discussed above, it is necessary to investigate SRT and HRT guarantees in depth. It is interesting to find that these two guarantees just consider the criteria of real-time QoS from two extreme aspects, i.e., one underlines each individual and the other underlines the total from infinite individuals. Obviously, there is a gap between these two guarantees, and an alternative between single to infinite individual needs. To capture the situation that deadline misses with a permitted distribution over a finite range can be tolerated, a second parameter describing the length of finite individuals or time should be specified. Nagarajan *et al.*(1994) proposed two criteria called Interval QoS and Block QoS. Unlike statistical QoS criteria over infinite interval of time, these two types of QoS are measured over finite interval of time. Subsequently, two similar QoS criteria, (m, k)-firm constraint and windowed lost rate, were independently proposed in (Hamdaoui and Ramanathan, 1995; 1997; West *et al.*, 1999; West and Poellabauer, 2000). Furthermore, these QoS criteria were recently generalized under the name of WHRT schedule theory (Bernat and Burns, 2001) to deal with real-time system that can tolerate some degree of deadline misses provided that this number of missed deadlines is bounded and precisely distributed. (m, k)-firm constraint or WHRTC requires the guaranteeing of deadline of at least m out of k consecutive instances, otherwise the system is said to be in a failure state.

New strategies to schedule systems with (m, k)-firm constraint or WHRTC have been proposed (Koren and Shasha, 1995; Ramanathan, 1999; Quan and Hu, 2000; Bernat and Burns, 2001; Bernat and Cayssials, 2001; West and Poellabauer, 2000;

Striegel and Manimaran, 2003), and the criterion of comparison between these scheduling approaches is mainly the failure probability (Wang *et al.*, 2002). Among strategies, the main and simplest (m, k)-based scheduling policy is DBP proposed by Hamdaoui and Ramanathan (1995) which is used to schedule multiple input queues single server (MIQSS) model, each having its (m, k)-firm constraint. The basic idea of DBP is to assign higher priorities to instances from streams that are closer to failure state so as to improve their chance of meeting their deadline. In case of priority equality among these head-of-queue instances, EDF is used. It had been shown (Hamadaoui and Ramanathan, 1995) that when identical streams, i.e. with the same service time distribution, the same inter-arrival distribution and the same deadline, have different (m, k)-firm constraints, DBP is especially beneficial for decreasing the failure probability. Furthermore, there are many improvements on DBP since it was proposed, such as DBP-M and EDBP in (Lindsay and Ramanathan, 1997) and (Striegel and Mainmaran, 2000; 2002) respectively.

However, DBP combined with EDF cannot always provide good performance. One reason is that DBP only utilizes the distance to failure state of each individual stream. Just as shown in our previous research, DBP initially designed for identical streams is essentially a self-reference or local strategy since the relationship with other streams is neglected completely, and a remedy approach such as matrix-DBP (Poggi *et al.*, 2003; Song, 2003) for heterogeneous streams should be provided. The other limit is that DBP does not underline the rich information of deadline met/missed distribution, specially streams in failure states which will travel different distances to restore their success states. Considering how to effectively restore success state from failure state of each individual stream, an integrated DBP utilizing deadline met/missed distribution is proposed in this paper.

In what follows, we point out in Section 2 why WHRT schedule theory is interesting. This will give us an opportunity to state in Section 3 what is the essential advantage DBP and its two drawbacks under the framework of WHRT schedule theory. Then, an improved DBP integrating rich information of deadline met/missed distribution is proposed in Section 4. In Section 5, simulation results showed the performance improvement of this new proposal in terms of failure probability in overload scenarios. Finally in Section 6 we give conclusions and point out future work.

BASIC CONCEPT OF WHRT SCHEDULE THEORY

Application context: MIQSS

As briefly mentioned in the introduction section, we are interested in studying how to effectively serve multiple streams sharing a single server, referred to as MIQSS model. MIQSS can be used to study a large category of computer and telecommunication systems such as execution of multiple tasks in a CPU, transmission of tasks issued from multiple task sources sharing the same transmission medium or network interconnection equipment (switch and router).

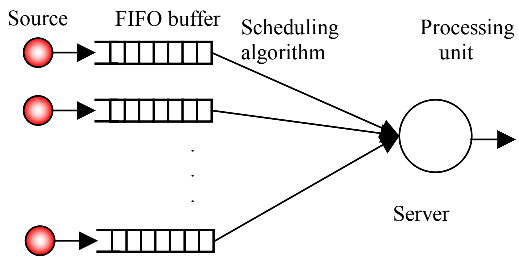

Fig.1 MIQSS model

The proposed model is made up of N sources generating N streams of tasks τ_i (i =1,2,…,N) to be served by a single server.

Each stream is formed by a source and a waiting queue (FIFO buffer), where an instance (a task consists of periodic or aperiodic instances) issued from the source waits until chosen by the server. The server chooses instances at the head of queues according to its scheduling policy.

Weakly hard real-time schedule theory

A good alternative between HRT and SRT

requirement is necessary in order to provide enough fine granularity for real-time requirement or real-time QoS, which is lacking at present just as briefly mentioned in the introduction section. To deal with the practical issues, a new criteria for real-time QoS and a correspondingly new schedule theory referred to as WHRT schedule theory are proposed.

Definition 1 WHRT schedule theory and WHRTC: A WHRT schedule theory is a conceptual framework, which investigates the characteristics of real-time systems that can tolerate certain deadline missed under a precise distribution over a finite time window. Correspondingly, the temporal constraint under the context of WHRT schedule theory refers to WHRTC.

Hence, a real-time QoS criteria, i.e., tolerance to deadline missed is established within a finite time window. Actually, the finite time window can be taken with any feasible styles, and the consecutive instances of the task in (m, k)-firm is only a kind of finite time window. In fact (m, k)-firm constraint just means that at least m instances out of any windows of k consecutive instances of the task τ must meet the deadline. In the following parts of this paper, (m, k)-firm constraint is substituted by (m, k) WHRTC.

Definition 2 Failure state and success state: a task τ with (m, k) WHRTC is in a success state if m out of its last k consecutive instances meet the deadline, otherwise τ is in a failure state.

From the above description, we can see a task may experience different states. Take a task with (2, 3) WHRTC as example, the task will experience eight different states as indicated in Fig.2.

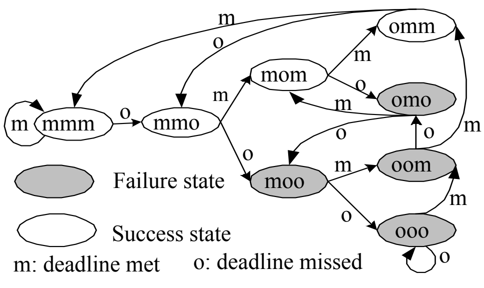

Fig.2 State transition diagram of task with (2,3) WHRTC

Fig.2 also suggests that the closer a state is to its failure state, the more easily the state suffers failure. Just consider "mmm" and "mom", the former can suffer at most two deadline misses, while the latter can suffer only one deadline miss. Therefore, for a task τ, each of its states corresponds to different distances to failure state, referred to as violating distance.

Definition 3 Violating distance: For a task τ with (m, k) WHRTC, the allowed number of consecutive deadline misses is referred to as violating distance, i.e. the distance to a failure state from current success state.

For a MIQSS model with (m, k) WHRTC, it is straightway to assign priority to each task τ based on its violating distance (This is just what DBP does, which will be discussed in depth in Section 3). The evaluation of this distance can be done exactly through considering the last k-length history of task τ, the pattern of deadline met/missed, referred to as μ-pattern.

μ-pattern and its violating distance

Definition 4 μ-pattern: For a task τ with (m, k) WHRTC, its μ-pattern is a word of k bits ordered from the most recent to the oldest instance in which each bit keeps memory of whether the deadline is missed (bit=0) or met (bit=1). Correspondingly, $\mu_1(\tau)$ and $\mu_2(\tau)$ are respectively the oldest and most recent instance during μ-pattern of task τ, i.e., the leftmost bit represents the oldest.

Each new instance causes a leftward shift of all the bits, the leftmost exits from the word and is no longer considered, while the rightmost will be a 1 if the task τ has met its deadline, i.e. it has been served within deadline or a 0 otherwise. Fig.3 gives an example with (4,5) WHRTC.

Therefore for a task τ with (m, k) WHRTC, the

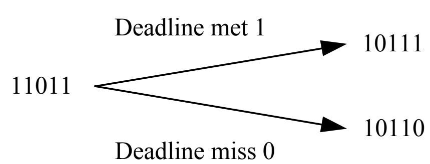

Fig.3 Possible evolution of the μ-pattern

priority of its instance at a given instant can be easily assigned with the help of its μ-pattern. Evolving the μ-pattern can determine the violating distance of the task. That is left shift the μ-pattern and adding in the right side 0s until the evolved μ-pattern violates (m, k) WHRTC of the task, and the number of added 0s is the priority. If a task stream is already in failure state (i.e., less than m 1s in the μ-pattern), the highest priority 0 is assigned. Just take a task τ with (4,6) WHRTC as example to show how to determine violating distance.

Example 1 For a state of (110011), τ falls into a failure state if its next deadline is missed, i.e., left shift μ-pattern of (110011) and add 0.

$$\overline{1}\;\underbrace{\overline{10011}\,\vdots\,\overbrace{0}^{1}}_{\text{Violating}(4,6)}$$

Example 2 For a state of (111111), τ can remain in a success state even if it consecutively suffers two deadline misses, i.e., left shift μ-pattern of (111111) two bits and add 0s, and will fall into a failure state if it further suffers deadline miss.

$$\overline{111}\;\underbrace{\overline{111}\,\vdots\,\overbrace{000}^{3}}_{\text{Violating}(4,6)}$$

Example 3 For a state of (101111), τ can at most suffer two consecutive deadline misses before falling into failure state.

$$\overline{10}\,\underbrace{\overline{1111}\,\vdots\,\overbrace{000}^{3}}_{\text{Violating}(4,6)}$$

In fact, Examples 1, 2 and 3 explain how to determine violating distance of a task. Normally, for a task τ with constraint $\beta=(m, k)$ WHRTC, let $VD^{\beta}(\tau)$ denote its violating distance, we get

$$VD^{\beta}(\tau)=\begin{cases} q \left| \sum_{i=q+1}^{k}(1-\mu_i(\tau))+q=k-m+1 \right. & \text{if } \sum_{i=1}^{k}\mu_i(\tau)\geq m \\ 0 & \text{if } \sum_{i=1}^{k}\mu_i(\tau)< m \end{cases} \tag{1}$$

Just consider the different μ-patterns task τ in Examples 1, 2 and 3, their violating distances are as follows,

$VD^{(4,6)}$ (110011)=1

$VD^{(4,6)}$ (101111)=3

$VD^{(4,6)}$ (111111)=3

MIQSS model under WHRT schedule theory

We further investigate the MIQSS model by characterizing its task τ_i as

$$\tau_i=(D_i, T_i, C_i, \beta_i) \tag{2}$$

where D_i denotes deadline, the maximum allowed time from the relaxation of its one instance to completion of the instance; T_i denotes period, the interval between release times of its two conescutive instances; C_i denotes execution time, i.e., the maximum time to complete its one instance without any interruption; β_i denotes its WHRTC enforced to task τ_i. In this paper, $\beta_i=(m_i, k_i)$ WHRTC.

In such a model, the choice of the scheduling algorithm is of prime importance for providing not only (m, k) WHRTC guarantee for each individual stream (end user point of view) but also good server utilization (system design point of view). When the system is overloaded, we, according to (m, k) WHRTC, best select instances, which will be discarded instead of being served.

Note the starting time of a task stream is not considered in our model, as in practice a periodic task stream can start at random time and often with jitter rather than strictly static. This simple model will help us focus on the main problem of MIQSS model with WHRTC.

Moreover, we decide to focus our attention only on the non pre-emptive and on-line schedule schemes as our main application will be the message schedule (so non pre-emptive) for providing real-time QoS guarantee in ATM, IP (Intserv, Diffserv) and in Industrial switch Ethernet (static off-line resources reservation is simply not feasible as such systems should be feasible and extensible for accepting new flow) (Song *et al.*, 2002). Of course, such a scheduling algorithm must also be of low complexity in order to be implemented on-line.

CONCEPT AND DRAWBACK OF DBP IN (HAMDAOUI AND RAMANATHAN, 1995)

DBP was firstly introduced by Hamdaoui and Ramanathan (1995), as a dynamic priority assignment mechanism in MIQSS model with (*m*, *k*) WHRTC. DBP gets lot focuses for its simplicity and easy implantation, and various improvement approaches are proposed, such as DBP-M (Lindsay and Ramanathan, 1997), EDBP (Striegel and Manimaram, 2000) and Matrix-DBP (Poggi *et al.*, 2003). However the rich information of deadline met/miss distribution is still neglected up to now, which leads to DBP and its variations providing low performance under heavy load. This section will deal with results of investigation on the essentials of DBP and the main reason for its drawback with the help of μ-patterns. Then a remedy approach called integrated DBP (IDBP) utilizing the information of deadline met/miss distribution is proposed in the next section.

Concept of DBP in (Hamdaoui and Ramanathan, 1995)

Just as shown in the above section that a task with (*m*, *k*) WHRTC will experience various states, and a failure state occurs when the task stream's (*m*, *k*) WHRTC is transgressed, i.e., there is more than *k*-*m* deadline misses within the last *k*-length window. Fig.1 also suggests that the closer a task is to its failure state, the easier is the task to suffer failure state. All these activate the idea of DBP schedule; it is quite simple and straightforward. i.e., the closer the stream is to the failure state the higher is its priority. As for a task with (*m*, *k*) WHRTC, DBP proposed in (Hamdaoui and Ramanathan,1995) designs its priority only according to the information of its last k-length consecutive instances, i.e. assign priority to an instance of a task though utilizing violating distance of the task.

Furthermore, Fig.1 also suggests that for a task with (*m*, *k*) WHRTC, the trend of its state to failure state is related to the position of *m*th 1 (position when *m*th deadline is met) from the last *k* instances. Examples 1, 2 and 3 further validate the argument. In fact, it will be found that DBP in (Hamdaoui and Ramanathan, 1995) just applies a smart approach to calculate violating distance by calculating the position of *m*th 1.

Formally, according to DBP (Hamdaoui and Ramanathan, 1995) the priority of a stream is evaluated as follows:

$$DBP^{\beta}(\tau_j) = \begin{cases} k - l(m, s_j) + 1 & |l(m,\tau) \le m \\ 0 & |l(m,\tau) \le m \end{cases} \tag{3}$$

where $s_j = (\delta^j_{i-k_j+1}, ..., \delta^j_{i1}, \delta^j_i)$ denotes the state of the previous *k* consecutive instances of τ_j, $l(n, s)$ denote the position (from the right) of *n*th met (or 1) in the τ_j, then the priority of the (*i*+1)th instance of τ_j is given by Eq.(3).

Fig.4 gives the implementation of DBP scheduling algorithm.

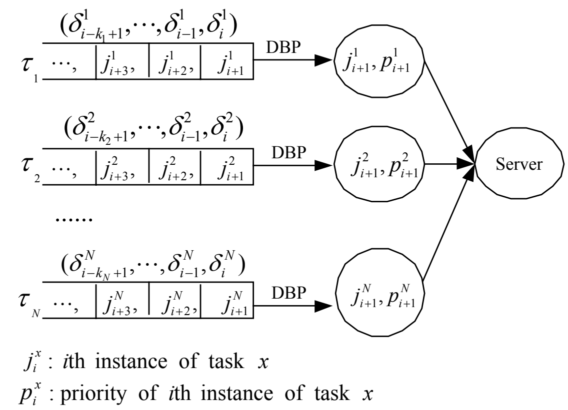

Fig.4 Implementation of DBP

In fact, the result of $DBP^{\beta}(\tau)$ and $VD^{\beta}(\tau)$ are the same, this can be validated by Examples 1, 2 and 3. That means we propose a more general framework for WHRT schedule theory since DBP can be treated as a special case here.

$DBP^{(4,6)}$ (110011)=1

$DBP^{(4,6)}$ (101111)=1

$DBP^{(4,6)}$ (111111)=1

Drawback of DBP in (Hamdaoui and Ramanathan, 1995)

It is also important to note that DBP underlines choosing priority based on the history of the stream's μ-pattern, and in case of priority equality DBP (Hamdaoui and Ramanathan, 1995) utilizes the simplest and common way to overcome these problems; that is, to assign DBP-based priority to the instances and use another scheduling algorithm among the already known ones. In their paper, Hamdaoui and Ramanathan (1995) combined DBP with EDF. However, the DBP suffers two drawbacks: one is that it does not take into account any specific information on the actual attributes of the stream like its length C_j, its minimum inter-arrival time T_j, and its deadline D_j; the other is rich information on deadline met/miss distribution is neglected completely.

The first limit leads to self-reference or local strategy. The solution of DBP in (Hamdaoui and Ramanathan, 1995) gives to parameters of other streams (such as deadline) less importance than that given to the μ-pattern, since EDF would be used only when μ-pattern is not sufficient, i.e. when two streams get the same DBP priority. In general, according to our earlier simulation study, using DBP with a dynamic sub-algorithm to choose in ambiguous cases may be quite disappointing. Sometimes, underestimating information of other streams will lead DBP to a local optimal or self-reference, instead of global optimal, and lead to poor performance. We call this self-reference or local strategy. Fortunately, this problem was solved in recent research (Poggi *et al.*, 2003).

The second problem of DBP is that it only uses the distance of the μ-pattern to its failure state whereas the richer information on the whole μ-pattern contains are neglected. To understand this point, it is enough to consider two μ-patterns under (2,5)-WHRTC: 11100 and 11001. They are both the same distance (=2). If they are issued from two streams with the same characteristics (T_i, C_i, D_i) and the instances arrived at the same time, which one should be chosen?

It appears that 11100 is less robust than 11001. For example, after a successful service of both, the two μ-patterns become respectively:

11100→11001 (DBP=2)

11001→10011 (DBP=4)

But this does not necessarily mean that we should first serve the first μ-pattern since both $DBP^{\beta}(\tau)$ and $VD^{\beta}(\tau)$ had set the same priority (=2) for both patterns.

Always with (2,5)-WHRTC, DBP gives the same priority (=0) to the two following different μ-patterns in failure state: 00001 and 10000. But to exit a failure state, 00001 will need just one more 1 whereas 10000 needs two 1s.

It is obvious that DBP in (Hamdaoui and Ramanathan, 1995) lacks remedy approach. Consider $DBP^{\beta}(\tau)$ and $VD^{\beta}(\tau)$, where the violating distance only guarantees a task, which is prone to a failure state be assigned higher priority. However, there are no differences to tasks which are already in failure states. Furthermore, consider the following examples.

Example 4 $DBP^{(4,6)}(100011)=0$

Example 5 $DBP^{(4,6)}(111000)=0$

Example 6 $DBP^{(4,6)}(000111)=0$

For these three failure states, there exist different distances to restore their success states. This argument is validated by evolving these failure states in Examples 4, 5, and 6 as follows:

Example 4: $\overline{1}\,\overline{0}\underbrace{\overline{0}\,\overline{0}\,\overline{1}\,\overline{1}}_{\text{Meet}(4,6)}:\overbrace{\overline{1}\,\overline{1}}^{2}$

Example 5: $\overline{1}\,\overline{1}\,\overline{1}\,\overline{0}\underbrace{\overline{0}\,\overline{0}:\overline{1}\,\overline{1}}_{\text{Meet}(4,6)}\overbrace{}^{4}$

Example 6: $\overline{0}\underbrace{\overline{0}\,\overline{0}\,\overline{1}\,\overline{1}\,\overline{1}:\overline{1}}_{\text{Meet}(4,6)}\overbrace{}^{1}$

To distinguish task streams in failure state, we propose to extend the notion of violating distance to failure state by introducing the restoring distance to success state, which will be normally defined in the next section.

Adding remedy approach: an integrated DBP

In fact, how to use such information of deadline met/missed, which can be exploited in several ways, depends on what we want to optimize and on what complexity we want to introduce. Always keeping violating distance as basic priority assignment function, one of the possible ways is to focus interest in failure states.

Definition 6 Critical success state: For a task τ with (m, k) WHRTC, critical success state is a success state, which only has m 1s in its last k-length state.

For example, for a task τ with (4,6) WHRTC, the following success state or μ-pattern is in critical success state, (111100), (110011) and (001111).

Definition 7 Restoring distance: For a task τ with (m, k) WHRTC, the minimum number of deadline met is referred to as restoring distance, i.e. the distance to critical success state from a current failure state.

Similarly, restoring distance of a task at a given instant can be achieved as the restoring distance of the task can be evaluated by evolving its μ-pattern by left shifting of the μ-pattern and adding in the right side 1s until the evolved μ-pattern meet (m, k) WHRTC of the task (the number of added 1s refers to restoring distance). If a stream is already in a success state (i.e., more than m 1s in the μ-pattern), the restoring distance is 0.

Example 7 For a state of (101110), τ can be restored to critical success state if it consecutively meets 5 deadlines.

$$\overline{1}\,\overline{0}\,\overline{1}\,\overline{1}\,\overline{1}\,\underbrace{\overline{0}:\overbrace{11111}^{5}}_{\text{Meet}(5,6)}.$$

Example 8 For a state of (101101), τ can be restored to critical success state if it consecutively meets two deadlines.

$$\overline{1}\,\overline{0}\,\underbrace{\overline{1}\,\overline{1}\,\overline{0}\,\overline{1}:\overbrace{11}^{2}}_{\text{Meet}(5,6)}$$

Example 9 For a state of (100111), τ can be restored to critical success state if it consecutively meets two deadlines.

$$\overline{1}\,\overline{0}\,\underbrace{\overline{0}\,\overline{1}\,\overline{1}\,\overline{1}:\overbrace{11}^{2}}_{\text{Meet}(5,6)}.$$

From the above three examples, it is similarly easy to get the normal method for calculating restoring distance for a task τ with constraint $\beta=(m, k)$ WHRCT.

Let $RD^{\beta}(\tau)$ denote its restoring distance,

$$RD^{\beta}(\tau)=\begin{cases} q\left|\displaystyle\sum_{i=k-q+1}^{k}\mu_i(\tau)+q=m\right|\displaystyle\sum_{i=1}^{k}\mu_i(\tau)<m \\ 0 \qquad\qquad\qquad\qquad\quad \left|\displaystyle\sum_{i=1}^{k}\mu_i(\tau)\geq m\right. \end{cases} \tag{4}$$

For these three states in Examples 7, 8, 9, their restoring distances can be easily obtained as follows,

$RD^{(5,6)}$ (101110)=5

$RD^{(5,6)}$ (101101)=2

$RD^{(5,6)}$ (100111)=2

Note μ-pattern of task τ cannot be in a success sate or a failure state simultaneously, so we just use $VR(\tau)$ to judge the state of μ-pattern of this task.

$$VR(\tau)=\begin{cases} 1 & \left|\displaystyle\sum_{j=1}^{k_i}\mu_j(\tau)\geq m\right. \\ 0 & \quad\text{elsewise} \end{cases} \tag{5}$$

After considering the above factors, an integrated DBP (IDBP) for designing priority of task τ is proposed as follows,

$$P(\tau)=VD^{\beta}(\tau)VR(\tau)+RD^{\beta}(\tau)(1-VR(\tau)) \tag{6}$$

where $RD^{\beta}(\tau)$ has the same meaning as $VD^{\beta}(\tau)$.

SIMULATION RESULTS

To evaluate the performance of the IDBP we proposed and to compare it with classic DBP, the following scenario was simulated.

Two adjunctive scheduling policies: IDBP+EDF and DBP+EDF were evaluated through simulation examples given in (Hamdaoui and Ramanathan, 1995) and take the probability of dynamic failure (the number of violating (m, k)-WHRTC divided by total state) as optimized objective function. In the following results, two generation patterns of instances were considered: Possion stream and bursty stream. In a Possion stream, instances of inter-arrival times are exponentially distributed. A bursty source alternates between ON and OFF states. When in the ON state, instances are generated periodically. No instances are generated when the source is in OFF state. The durations of the ON and the OFF states are exponentially distributed, with averages ON_{ave} and OFF_{ave} respectively. Such a stream is often used to model a stream of voice samples in a conversation. We firstly consider the case where all streams in the system have the same timing requirements. We also assume that only the instances that meet their deadline are serviced, which means that drop policy is enabled. The simulation adopted software OPNET8.0. Time of all simulation was 20000 s in the following three examples, and the average value was taken as the final result.

The server load and load (m, k) are defined as follows:

$$Load = \sum_{j=1}^{n} \frac{C_j}{E(T_j)} \tag{7}$$

$$Load(m,k) = \sum_{j=1}^{n} \frac{m_j C_j}{k_j E(T_j)} \tag{8}$$

where C_j is service time, $E(T_j)$ is the mean period of inter-arrival time.

Poisson streams

The plots in Fig.5 show the probability of dynamic failure in a system with (3,4)-WHRTC. The system consisted of five streams. All instances required a constant service time. Service deadline were set equal to five times the instance service time. The instance inter-arrival time was exponentially distributed and the overall average load varied from 0.8–2.0. Fig.4 of the result shows that the adjunctive scheduling policy IDBP can reduce the probability of dynamic failure, especially when the average load is heavy.

The above system considered that all streams had the same deadline requirement with (3,4)-WHRTC. Fig.6 compares results for the heterogeneous system of streams with different deadline requirement. The system consisted of five streams: (9,10)-WHRTC, (3,4)-WHRTC, (1,2)-WHRTC, (1,3)-WHRTC, (1,4)-WHRTC, respectively. The instance service time, arrival pattern, and the deadline in this system were like those for the streams in Fig.5. The arrival rates of the stream were adjusted to get average load of 1.0–2.3. This figure shows that even at load 1.5–1.7, there was a little abnormal behavior in that DBP+EDF was better than IDBP+EDF. But there was also a trend that IDBP can reduce the probability of dynamic failure at heavy load.

Burst streams

Fig.7 shows the plots of probability of dynamic

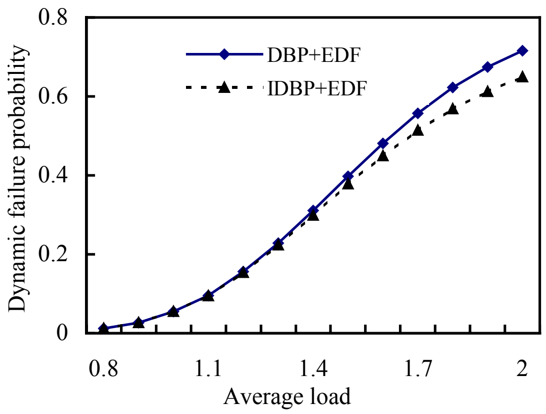

Fig.5 Probability of dynamic failure of IDBP and DBP for Poisson streams with (3,4) WHRTC

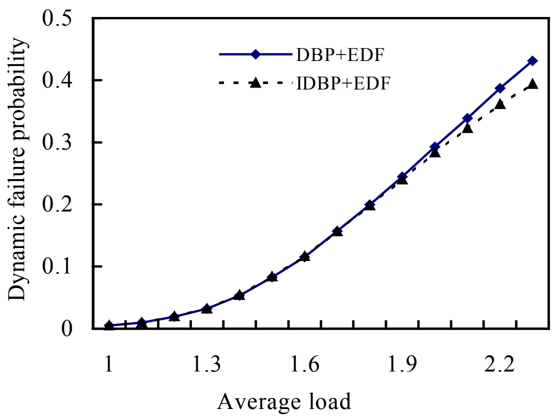

Fig.6 Probability of dynamic failure of IDBP and DBP for heterogeneous system

failure in a system with five burst streams. The ON and OFF periods of each stream are exponentially distributed with ON_{ave}=50 and OFF_{ave} =100. The peak load of a stream is therefore three times the average load. When in the ON state, the period of a stream generating one instance is 5, and exponentially distributed. The deadlines were set to twice the average generation period. Overall load was varied by changing instance service time. Fig.7 shows that IDBP was better than DBP for guaranteeing the (*m*, *k*) WHRTC and reducing probabilities of dynamic failure obviously with load varied; from 0.5−1.5, and that IDBP can performed better under heavy load.

All the above figures show obviously that IDBP can more effectively reduce the probability of dynamic failure under the condition of load larger than 1 (Load(*m*, *k*) 0.9, 0.984, 0.6 according to the above three examples). Because at heavy load, there are more streams in failure states, and more same priorities (*DBP*=0), IDBP can more embody its effect.

CONCLUSION

The main idea behind DBP is the proposed integrated approach DBP based on utilizing the rich information of deadline met/missed, which has been neglected up to now. The algorithm proposed here can guarantee the priority assignment according to both violating distance and restoring distance. The new approach effectively enhances initial DBP's capability to deal with failure states; and was validated by various simulation cases.

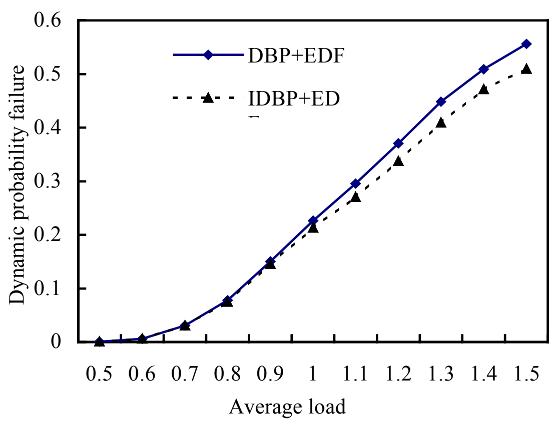

Fig.7 Probability of dynamic failure of IDBP and DBP for burst-stream (3,4) WHRTC

This improvement involved very low computing cost or complexity since it is achieved by checking the right position which need adding minimum 1's. Furthermore, this new computing is needed for a task only then the task is in failure state. In this sense, the implementation of our algorithm in an admission control mechanism for providing (*m*, *k*) WHRTC in a network should be interesting.

Furthermore, (*m*, *k*)-firm or WHRTC is just the beginning; how to utilize this new fundamental frame to unify HRT and SRT, how to provide finer QoS with WHRTC concept, and how to adjust available algorithms such as WFQ and RED after more effectively utilising the practical real-time requirement by the help of WHRTC, all needs further investigation in detail.

References

Atiquzzaman, M., Hassan, M., 2001. Guest editorial: Adaptive real-time multimedia transmission over packet switching networks. *Real-Time Imaging*, **7**(3): 219-220.

Bernat, G., Burns, A., 2001. Weakly hard real-time systems. *IEEE Transactions on Computers*, **50**(4):308-321.

Bernat, G., Cayssials, R., 2001. Guaranteed On-Line weakly Hard Real-Time Systems. Proceedings of IEEE Conference on Real-Time Systems, p.25-35.

Giorgio, C.B., 1997. Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Application. Kluwer Academic Publisher.

Chou, C.C., Shin, K.G., 1997. Statistical real-time channels on multi-access bus networks. *IEEE Transaction on Parallel and Distributed Systems*, **7**(8):769-780.

Cruz, R.L., 1991. A calculus for network delay, Part I. *IEEE Transactions on Information Theory*, **37**(1):114-131.

Hamdaoui, M., Ramanathan, P., 1995. A dynamic priority assignment technique for streams with (*m*, *k*)-firm deadlines. *IEEE Transactions on Computers*, **44**(4): 1443-1451.

Hamdaoui, M., Ramanathan, P., 1997. Evaluating dynamic failure probability for streams with (*m*, *k*)-firm deadline. *IEEE Transactions on Computers*, **46**(12):1325-1337.

Koren, G., Shasha, D., 1995. Skip-over: Algorithms and Complexity for Overloaded Systems that Allow Skips. Proceedings of Real-Time Systems Symposium,

p.110-117.
Kuo, T.W., Mok, A.K., 1991. Load Adjustment in Adaptive Real-time. Proceedings of 12th IEEE Real-Time System Symposium, p.160-170.
Lee, C., Rajkumar, R., Mercer, C., 1996. Experiences with Processor Reservation and Dynamic QoS in Real-Time. Proceedings of Multimedia.
Lehoczky, J.P., 1990. Fixed Priority Scheduling of Periodic Task Sets with Arbitrary Deadlines. Proceedings of IEEE Real-Time Systems Symposium, p.201-209.
Lindsay, W., Ramanathan, P., 1997. DBP-M, A Technique for Meeting End-to-end (*m*, *k*)-Firm Guarantee Requirements in Point-to-point Networks. Proceedings of IEEE Conference on Local Networks, p.294-303.
Nagarajan, R., Kurose, J., Towsley. D., 1994. Finite-horizon statistical quality of service measure for high-speed networks. *Journal of High Speed Network*, **3**(4):351-373.
Nakajima, T., 1998. Resource Reservation for Adaptive QoS Mapping in Real-time Mach. Proceedings of 6th International Workshop on Parallel and Distributed Real-time System.
Poggi, E.M., Song, Y.Q., Koubâa. A., Wang, Z., 2003. Matrix-DBP for (*m*, *k*)-firm Real-time Guarantee. Proceedings of Real-Time and Embedded System, Paris, France, p.457-482.
Quan, G., Hu, X., 2000. Enhanced Fixed-priority Scheduling with (*m*, *k*)-firm Guarantee. Proceedings of IEEE Real-Time Systems Symposium, p.79-88.
Ramamritham, K., 1996. Where do time constraints come from and where do they go? *International Journal of Database Management*, **7**(2):4-10.
Ramanathan, P., 1999. Overload management in real-time control applications using (*m*, *k*)-firm guarantee. *IEEE Transactions on Parallel and Distributed Systems*, **10**(6):549-559.
Rom, R., Sidi, M., 1990. Multiple Access Protocols Performance and Analysis. Springer-Verlag, New York.
Song, Y.Q., Koubâa, A., Simonot, F., 2002. Switched Ethernet for Real-time Industrial Communication: Modelling and Message Buffering Delay Evaluation. Proceedings of 4th IEEE WFCS, p.27-30.
Song, Y.Q., 2003. Gestion Dynamique de la QdS Temps-réel Selon (*m*, *k*)-firm. Report in Ecole Temps Réel, Toulouse, France, ETR'03.
Striegel, A., Manimaran, G., 2000. Best-effort scheduling of (*m*, *k*)-firm real-time streams in multihop networks. *Computer Communications*, **23**(13):1292-1300.
Striegel, A., Manimaran, G., 2002. Packet scheduling with delay and loss differentiation. *Computer Communications*, **25**(1):21-31.
Striegel, A., Manimaran, G., 2003. Dynamic class-based queue banagement for scalable media servers. *Journal of Systems & Software*, **66**:119-128.
Takagi, H., 1990. Queuing Analysis of Polling System: An Update in Stochastic Analysis of Computer and Communication System. Elsevier Science North Holland, Amsterdam, p.267-318.
Wang, Z., Song, Y.Q., Poggi, E.M., Sun, Y.X., 2002. Survey of Weakly Hard Real-time Schedule Theory and its Application. Proceedings of International Symposium on Distributed Computing and Applications to Business, Engineering and Science, p.429-437.
West, R., Schwan, K., Poellabauer, C., 1999. Scalable Scheduling Support for Loss and Delay Constrained Media Streams. Proceedings of IEEE Real-Time Technology and Applications Symposium.
West, R., Poellabauer, C., 2000. Analysis of A Window-con-strained Scheduler for Real-time and Best-effort Packet streams. Proceeding of 21st IEEE Real-Time Systems.
Zhang, Z.L., Nelakuditi, S., Aggarwal, R., Tsang, R.P., 2001. Efficient selective frame discard algorithms for stored video delivery across resource constrained networks. *Real-Time Imaging*, 7(3):255-273.