International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:3, No:3, 2009

Research on Weakly Hard Real-Time Constraints and Their Boolean Combination to Support Adaptive QoS

Xiangbin Zhu

***Abstract*—**Advances in computing applications in recent years have prompted the demand for more flexible scheduling models for QoS demand. Moreover, in practical applications, partly violated temporal constraints can be tolerated if the violation meets certain distribution. So we need extend the traditional Liu and Lanland model to adapt to these circumstances. There are two extensions, which are the (m, k)-firm model and Window-Constrained model. This paper researches on weakly hard real-time constraints and their combination to support QoS. The fact that a practical application can tolerate some violations of temporal constraint under certain distribution is employed to support adaptive QoS on the open real-time system. The experiment results show these approaches are effective compared to traditional scheduling algorithms.

***Keywords*—**Weakly Hard Real-Time, Real-Time, Scheduling, Quality of Service.

I. Introduction

TRADITIONALLY, real-time systems are classified into two types[1]. One is hard real-time and another is soft real-time. The soft real-time systems can tolerate some deadlines missed occasionally. For hard real-time systems, no deadline miss is tolerated. The traditional real time schedule theories have served these systems well. But with the development of computer network and multimedia, lots of soft real-time systems, such as VOD, video conference and remote teaching and so on, are receiving increase attention. Specially, the quickly development in hardware makes many embedded systems be able to play video and audio streams. Many SRT(soft real-time) like these systems need the guarantee of QoS (quality of service). But the QoS of video play has its characteristics. For example, if there are ten deadlines missed in video play, it is different whether the ten deadlines missed continuously or not.

So these advances in computing applications have prompted the demand for more flexible scheduling models of real-time tasks. A key problem of flexible real-time system is how to guarantee the situation that deadlines missed with a permitted distribution over a finite range. The traditional real-time scheduling model cannot adapt these new real-time systems. So it is needed to extent to the Liu and Layland model[1][2][3] to allow flexibility in scheduling. There are two extensions, which are the (m, k)-firm model[4][5][6] and Window-Constrained model[7][8].

The weakly hard constraint model is one of the extensions to traditional real-time model. Weakly hard real-time systems is one of the task models proposed in recent years for scheduling periodic real-time tasks where partly violated temporal constraints can be tolerated if the violation meets certain distribution. Weakly hard real-time systems is developed from (n, m)-firm model by G.Bernat in 1997[9][10]. The weakly hard constraint is (n, m)-firm constraint and which consists of guaranteeing n out of m consecutive task executions or message transmissions, otherwise, the system is said to be in failure state.

Bernat and Burns summarize temporal properties of specifications available of weakly hard real-time systems, and point out the relations between various specifications. Further, they point out that a specification should be considered from two aspects:

(1) A task maybe is sensitive to the consecutiveness of deadline met while another is only sensitive to the number of deadline missed;

(2) A task maybe is sensitive to the consecutiveness of deadline missed while another is only sensitive to the number of deadline missed.

Concretely, they provide four types of basic specifications of weakly hard real-time constrains :(n, m), <n, m>, ($\overline{n}$, m) and <$\overline{n}$, m>.

But all schedule algorithms available about weakly hard real-time focus only (n, m)-firm constrain. There are no schedule algorithms on other weakly hard real-time constrains. Furthermore, we can define algebra of weakly hard constraint combining them with Boolean operators. For example, we can specify that a given task has to satisfy ((3,5) and (3,10)) or (8,10). In the same, there are no schedule algorithms on Boolean combination of weakly hard constraints.

On the other hand, as mentioned above, the multimedia task is a soft real-time task, which has some characteristics, such as its WCET cannot be known in advance and the task is usually aperiodic. Thus the traditional hard real-time scheduling algorithm is not adapted for soft real-time tasks. So the open real-time system[11][12][13][14] has been paid attention because of giving away to solve the problem and also brings

Xiangbin Zhu is with College of Mathematics, Physics and Information Science, Zhejiang Normal University, China (E-mail: zhuxb@zjnu.cn)

new ideas to scheduling theory and approach.

The open real-time scheduling is based on the bandwidth-preserving servers. The constant bandwidth server[15][16] is one of the best bandwidth-preserving servers. In an open real-time system, there are generally two level scheduling. The first level scheduler is usually EDF scheduling. But we select the weakly hard real-time scheduler as the first level scheduler because it is a good scheduler for scheduling multimedia tasks. Many computing applications have also prompted the demand for more flexible scheduling and quality of service(QoS) management[17]. For example, the periodic transmission of audio and video packets can usually tolerate a short blackout of the packet stream without significantly deteriorating the quality of the audio or video being transmitted. In other words, it is different in quality that ten video packets continuously or discretely miss their deadlines.

Summarize, the weakly hard real-time scheduling algorithm is more adapted than the EDF algorithm for the situation, which is different kinds of real-time tasks coexist in one system. It can provide more flexibility in the system behavior. The flexibility gives a way for the system to be more resilient and provide different QoS. But all schedule algorithms available about weakly hard real-time focus only (n, m)-firm constrain. So we have research on the other constrain and their Boolean combination.

In this paper, a task is a sequence of invocations. So the entities being scheduler are invocations and tasks. Beginning from the start time of a task, invocations become ready at the start of fixed interval.

In this paper, we present these algorithms to schedule these weakly hard constraints and application on open real-time system. The remainder of this paper is organized as follows: The next section defines and explains weakly hard real-time systems. All scheduling algorithms on weakly hard constraints are presented in section 3. Section 4 introduces the combination constraints used in QoS model on open real-time system. Section 5 presents the experiment results and Section 6 summarizes our work.

II. Specification of Weakly Hard Real-time Schedule Theory

A. Weakly hard real-time schedule theory

The weakly hard real-time schedule theory is to solve the new situation that most real time applications can tolerate certain deadline missed, but the certain deadline missed must be under a precise distribution over a finite time window.

Definition 1: A weakly hard real-time system is a system in which the distribution of it's met and missed deadlines during a window of time *w* is precisely bounded.

Systems that must meet all of their deadlines are a particular case of our definition and will be referred to as strongly hard real-time systems whenever the distinction is relevant.

The theory shows the tolerance to deadline missed is established within a window of consecutive invocations of tasks.

B. Process model

In this paper, a task τ_i consists of a sequence of jobs or invocations, $J_{i,j}$, where $r_{i,j}$ denotes the arrival time of the jth job of taskτ_i.

The basic process model consists of periodic tasks. These periodic tasks have some attributes, such as period , deadline and so on. We characterize a system with a set of n independent periodic tasks, $\Gamma=\{\tau_1,\tau_2,\ldots,\tau_i,\ldots,\tau_n\}$.Task τ_i described as the following model:

$$\tau_i=(T_i,C_i,D_i,R_i) \quad (1)$$

where T_i denotes interval between release times of two consecutive instances ofτ_i ; C_i denotes maximum time needed to completeτ_i without any interruption; D_i denotes maximum time allowed from the release time to the completion time ofτ_i*'s* instance. R_i denotes the weakly hard constrains. The j^{th} instance of taskτ_i is denoted asτ_{ij}.

All tables and figures you insert in your document are only to help you gauge the size of your paper, for the convenience of the referees, and to make it easy for you to distribute preprints.

C. Weakly Hard Temporal Constraints

There are four types of constraints:

Definition 2. A task "meets any *n* in *m* deadlines"($m\geq1$ and $0 \leq n \leq m$) and it is denoted $\binom{n}{m}$,if in any window of *m* consecutive instances of the task, there are at least *n* instances that meet the deadline.

Definition 3. A task "meets consecutive *n* in *m* deadlines"($m \geq1$ and $0\leq n\leq m$) and it is denoted $<\frac{n}{m}>$,if in any window of *m* consecutive instances of the task, there are at least *n* consecutive instances that meet the deadline.

Definition 4. A task "not misses any *n* in *m* deadlines"($m\geq1$ and $0\leq n\leq m$) and it is denoted $\binom{\overline{n}}{m}$,if in any window of *m* consecutive instances of the task, there are no more than *n* instances that miss their deadlines.

Definition 5. A task "not misses consecutive *n* in *m* deadlines"($m\geq1$ and $0\leq n\leq m$) and it is denoted $<\frac{\overline{n}}{m}>$,if in any window of *m* consecutive instances of the task, it is never the case that *n* consecutive instances that miss their deadlines.

D. Weakly Hard Temporal Constraints properties

Most of weakly hard constraints properties can be found in literature [4]. In this paper, we propose some new properties as followed:

Theorem 1: $<\frac{\overline{m-n+1}}{m}>\prec\binom{n}{m}$

Proof.

If $<\frac{\overline{m-n+1}}{m}>\succ\binom{n}{m}$, there are m-n+1 consecutive violations. It is showed in Fig.1.. So there are some weakly hard real-time

windows which include the consecutive violations. Each window has only at most n-1 times to meet deadline. But this can't satisfy the constraint $\binom{n}{m}$.

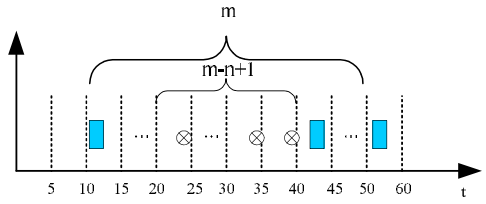

Fig. 1 A case of $<\overline{\frac{m-n+1}{m}}>\subseteq\binom{n}{m}$

Theorem 2: $<\overline{\frac{p}{m}}>\prec\binom{n}{m}$ $\quad p > m-n$

Proof.

If $<\overline{\frac{p}{m}}>\succ\binom{n}{m}$, there are p consecutive violations. It is showed in Fig.1.. So there are some weakly hard real-time windows which include the consecutive violations. Each window has only at most m-p times to meet deadline. But this can't satisfy the constraint $\binom{n}{m}$ because $n>m$-p.

Theorem 3: $<\overline{\frac{p}{m}}>\equiv\binom{1}{p}$

Proof.

$<\overline{\frac{p}{m}}>$ means there are not p consecutive violations. It is showed in Fig.2.. So for any weakly hard real-time windows which have p invocations, a deadline should be met. This means it satisfies the constraint $\binom{1}{p}$.

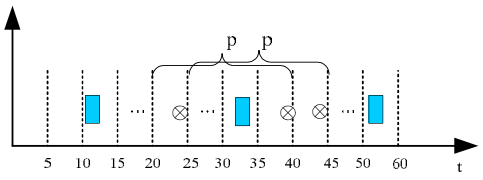

Fig. 2 A case of $<\overline{\frac{p}{m}}>$

III. DBP (Distance Based Priority) Schedule

The idea of DBP is that the priority of each task is assigned based on the number of deadline missed that task can still stand before violating its (n, m) requirement. The allowing number of deadline misses is referred to as distance. For the specification (n, m), the window of m consecutive instances has 2^n states. So we can draw a state transition diagram based on the history of each task. Thus, at every moment, the current window must be a state of the 2^n states. After missing some deadlines, the state can transmit to the failure state. The distance is the least number of transitions from current state to failure state. For example, Fig.3 indicates the state transition diagram of the specification (2, 3).

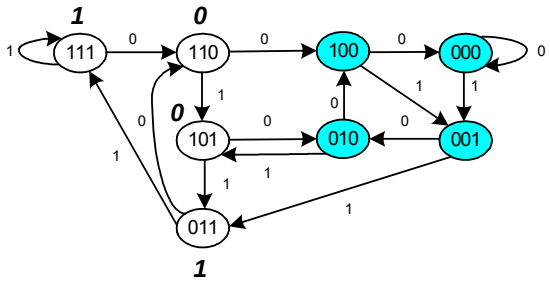

Fig. 3 The state transition diagram of (2, 3)

The DBP scheme can be implemented in hardware or in software. The current state can be kept in a m-bit shift register. Let 0 and 1 represent a deadline miss and a deadline meet, respectively. When the next customer is serviced, a 0 or a 1 is shifted in (from the right) depending on whether the customer missed or met its deadline. It is showed in Fig.4.. The register is initialized to all 1. The current state is denoted as s. We get the next priority from the parameter s. First, there are three function need to be defined:

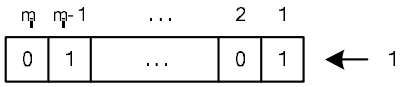

Fig.4 Shift register

1) $l_i(k,s)$ denotes the positon(from the right) of the kth meet(or 1) in the state s of task τ_i. If there are less than k 1's in s, then $l_i(k,s) = m_i + 1$. For example, suppose task τ_i has weak hard real-time restriction $\binom{2}{3}$. Then, $l_i(2,011) = 2$ and $l_i(3,011) = m_i + 1 = 4$.

2) $r_i(n_i,s)$ denotes the positon(from the right) of the first consecutive n_ith meet(or 1) in the state s of task τ_i.. For example, suppose task τ_i has weak hard real-time restriction $\binom{2}{5}$ .Then $r_i(3,01111) = 3$ and $r_i(3,01110) = 4$.

3) $t_i(s)$ denotes the 0's number at the tail in the state s of task τ_i.. For example, $t_i(010) = 1$ and $t_i(100) = 2$.

For different weakly real-time restrictions, we have:

1) For$\tau_i \vdash \binom{n_i}{m_i}$ the priority assigned to customer j+1 from task τ_i is given by:

$$priority^i_{j+1} = m_i - l_i(n_i,s) \qquad (2)$$

2) For $\tau_i \vdash < \frac{n_i}{m_i} >$ the priority assigned to customer j+1 from task τ_i is given by:

$$priority^{i}_{j+1} = m_i - r(n_i, s) - n_i + 1 \quad (3)$$

3) For $\tau_i \vdash \left(\frac{\overline{n}_i}{m_i}\right)$ the priority assigned to customer j+1 from task τ_i is given by:

$$priority^{i}_{j+1} = m_i - l(m_i - n_i, s) \quad (4)$$

4) For $\tau_i \vdash < \frac{\overline{n}_i}{m_i} >$ the priority assigned to customer j+1 from task τ_i is given by:

$$priority^{i}_{j+1} = n_i - t_i(s) - 1 \quad (5)$$

With these equations, we can get the priority of next state and which state is not safe or miss the weakly real-time deadline. For the weakly real-time $< \frac{2}{4} >$, although 1100 meets $< \frac{2}{4} >$ it is not safe because the next state 1001 or 1000 all cannot meet $< \frac{2}{4} >$ By the above equations, we can get the priority is the highest priority 0 for the previous *110.

Thus we solve the problem which is not solved in the literature [2], which is about weakly hard constraints and their Boolean combination. Suppose there are r weakly hard real-time constraints for a task. The algorithm is followed by:

(1)Compute all priorities of each constraints at current state, we have $\{R_1, R_2 \ldots, R_r\}$.

(2)Select the highest priority as the current priority.

For example, for a common combination constraint $\left(\frac{n}{m}\right) \wedge < \frac{\overline{h}}{m} >$, it means the task should meet any n in m deadlines and not misses consecutive h in m deadlines. We can employ the equation 2 and equation 5 to get priorities , and select the smaller one. For $\left(\frac{1}{3}\right) \wedge < \frac{\overline{2}}{3} >$ the assignment of priority is showed in Fig.5..

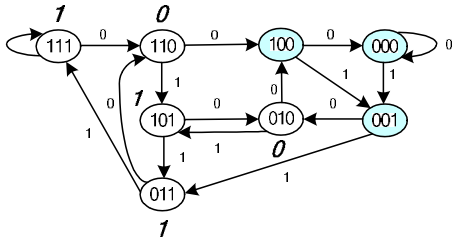

Fig. 5 The combination constraints

IV. The Combination Constraints Used in QoS Model on Open Real-Time System

A. Weakly Hard Real-Time Constant Bandwidth Server(WHRTCBS)

A soft real-time task is characterized by the parameters(C_i,T_i), however the timing constraints are more relaxed. In particular, for a soft task, C_i represents the mean execution time of each job, whereas T_i represents the desired activation period between successive jobs. So if C_i is adopted in real-time scheduling, CPU bandwidth will be wasted. Since mean values are used for the computation time and minimum inter-arrival times are not known, soft tasks cannot be guaranteed a priori. The deadline of each soft job $J_{i,j}$ is $d_{i,j}=r_{i,j}+T_i$.But the deadline missed only means the degrade of QoS, specially for multimedia applications. Thus for such real-time systems, the goal of scheduling is to be the best effort to reduce the lag of finish time, without jeopardizing the schedulability of the hard tasks[12].

For the advantage of EDF algorithm, we need the deadline for soft real-time task and aperiodic task. Thus we can use EDF scheduler to schedule soft real-time task and aperiodic task. For doing this, we need a mechanism, called server, to assign an absolute deadline to a job. The arriving jobs are enqueued in a queue of pending requests according to FIFO discipline. The first job will be allocated a deadline which can be used in EDF scheduling. The server is said to be active when the first job is executing.

A server S_s can server a job $J_{i,j}$ dividing it in smaller blocks, each of them will be assigned a fixed absolute deadline, named chunks. So the kth chunk is characterized by a release time $r_{i,j,k}$, a deadline $d_{i,j,k}$, a finish time $f_{i,j,k}$ and a computation time $c_{i,j,k}$[14].

For these multimedia applications such as VOD, video conference, digital library and online game, WHRTS can be the low-level scheduler in open real-time systems. Thus the advantages of WHRTS and CBS, for example CBS doesn't need know WCET in advance and WHRTS is a very flexible scheduler and so on, can be used to improve the schedulability in the open real-time system. But CBS must be changed.

For arrival time and execution time of the task in CBS are random, the main thought of weakly hard real-time constant bandwidth server should be the object scheduled by WHRTS, not be the task in CBS. The goal of WHRTS is best effort to meet the deadline of CBS because the object scheduled by WHRTS is CBS.

First, let Z denotes the set of weakly hard real-time combination constraints: $\{R_1, R_2 \ldots, R_r\}$

The new CBS is defined as follows:

1. Every CBS is characterized by a budget c_s and by a ordered triple (Q_s, T_s,Z_s), where Q_s is the maximum budget, T_s is the period of the server, c_s is the budget of the server and Z_s is the weakly hard real-time parameter.
2. At each instant, a fixed deadline $d_{s,k}$ is associated with the server. At the beginning $d_{s,0}$=0.
3. Each served job $J_{i,j}$ is assigned a dynamic deadline $d_{i,j}$

equal to the current server deadline $d_{s,k}$.

4. Whenever a served job executes, the budget c_s is decreased by the same amount.
5. When c_s=0, the server budget is recharged to the maximum value Q_s and a new server deadline is generated as $d_{s,k+1} = d_{s,k}+T_s$.
6. When a job $J_{i,j}$ arrives and the serve is idle, if $c_s \geq (d_{s,k}-r_{i,j})U_s$ the server generates a new deadline $d_{s,k+1} = d_{s,k}+T_s$ and c_s is recharged to the maximum value Q_s, otherwise the job is served with the last server deadline $d_{s,k}$ using the current budget.
7. When a job finishes, the next pending job, if any, is served using the current budget and deadline. If there are no pending jobs, the server becomes idle.
8. At any instant, a job is assigned the last deadline generated by the server.
9. **CBS violation:** if at the $d_{i,j}$ time, the job $J_{i,j}$ doesn't finish, we say a CBS violation occurs.
10. **CBS weakly hard real-time violation:** If a real-time task cannot meet constraints of Z_s , we say a CBS weakly hard real-time violation occurs.

The CBS violation is an important parameter because WHRTS scheduler uses the CBS violations to schedule all tasks.

But to the WHRTS algorithm, it need know the time when CBS server meets its deadline. For CBS server is scheduled, it should to judge whether meet the deadline of this server when the budget of the server is exhausted. From the above definition, the deadline needs to be postponed T_i time. Although the arrival of a new job can change the deadline, the arrival of the new job cannot be as a reason for scheduling. This is because the scheduling is for the CBS server.

So, when a weakly hard real-time CBS server meets its deadline, it needs not to change the Z parameter of the CBS server. It is the time to change the Z parameter of the CBS server when the budget of the CBS server is exhausted. At every scheduling instant, it needs to judge whether the other servers miss their deadlines. The deference in judge whether the real-time systems occurs misdeadlines between WHRTS scheduling of periodic tasks and that of CBS servers is listed below:

1. When the WHRTCBS is idle, although the time exceeds the deadline of the WHRTCBS, it cannot be as a violation occurs.
2. The tasks of the WHRTCBS are aperiodic tasks and their deadlines are random. When the violation occurs, the deadlines are not changed. Thus a violation doesn't mean next period of the WHRTCBS. A deadline only causes one violation.

B. WHRTCBS properties

The main property of WHRTCBS is the isolation property. But there are some differences. There are two situations: one is the task is general aperiodic task. We cannot know the arrival time of tasks and the execute time is varied. Thus it is difficult to decide the minimum bandwidth of WHRTCBS. We can only know the maximum bandwidth. Obviously, the maximum bandwidth is decided by the parameters Q_s and T_s. The maximum bandwidth is equal to Q_s /T_s because the window-constrain is at least m_s out of k_s deadlines should be met; another one is the task of WHRTCBS is periodic task. If the periodic taskτ_i is characterized by the ordered pair (C_i, T_i)and $Q_s \geq C_i, T_s=T_i$, we can have:

Definition 6: Let denotes μ_R as the value of weakly hard real-time constraints R and:

1) For $R=\binom{n}{m}$ the value μ_R is given by:

$$\mu_R = \frac{n}{m} \tag{6}$$

2) For $R=<\frac{n}{m}>$ the value μ_R is given by::

$$\mu_R = \frac{n}{m} \tag{7}$$

3) For $R=\vdash\binom{\bar{n}}{m}$ the value μ_R is given by::

$$\mu_R = 1-\frac{n}{m} \tag{8}$$

4) For $R=<\frac{\bar{n}}{m}>$ the value μ_R is given by::

$$\mu_R = 1-\frac{n}{m} \tag{9}$$

Definition 7: Let denotes μ_z as the value of weakly hard real-time constraints set Z and:

A. $\neg R : \mu_{\neg R} = 1-\mu_R$

B. $R_1 \vee R_2 : \mu_{R_1 \vee R_2} = Min\{\mu_{R_1}, \mu_{R_2}\}$

C. $R_1 \wedge R_2 : \mu_{R_1 \vee R_2} = Max\{\mu_{R_1}, \mu_{R_2}\}$

Theorem 4 A periodic taskτ_i is characterized by the ordered pair (C_i, T_i) and it runs on a WHRTCBS server which is characterized by $Q_s \geq C_i$, $T_s=T_i$ and Z_s. So the minimum bandwidth is $U_s = \frac{\mu_{Z_s} Q_s}{T_s}$.

Proof.

For a job $J_{i,j}$ of periodic taskτ_i, we can have $r_{i,j+1}-r_{i,j}=T_i$ and $c_{i,j} \leq Q_s$.So based on the definition, each job $J_{i,j}$ has a deadline $d_{i,j}=r_{i,j}+T_i$ and the server's budget $Q_s \geq c_{i,j}$. Moreover, for $c_{i,j} \leq Q_i$, each job would finish before the budget is exhausted. So the deadline is not changed and the WHRTCBS can be as a weakly hard real-time periodic taskτ_i which parameters is (C_s, T_s, Z_s). Thus the minimum bandwidth is $U_s = \frac{Z_s Q_s}{T_s}$.

Theorem 5 A periodic taskτ_i is characterized by the ordered pair (C_i, T_i) and it runs on a WHRTCBS server which is characterized by $Q_s \geq C_i, T_s=T_i$ and $R_s = R_i$. The necessary and sufficient condition for it can be scheduled is taskτ_i can be

scheduled by WHRTCBS algorithm.

Proof. See the above proof.

C. WHRTCBS to support QoS in open real-time system

We employ weakly hard real-time constraints to support QoS adaption in open real-time system. Fig.6 shows that the task of QoS is mainly finished by QoS manage module on open real-time system. The QoS manage module manages QoS of all servers and periodic task, including QoS parameter map and feedback control mechanism. The main work include two parts: one is to make the real-time system scheduling be feasible by control the bandwidth, the other is to allow user to adjust QoS parameters dynamically and dynamically negotiate between QoS modules, thus each task has a minimum QoS to guarantee. For example, for media application, QoS parameters include fps(frame percent second) and weakly hard real-time constraints $\{R_1,R_2\ldots,R_r\}$.

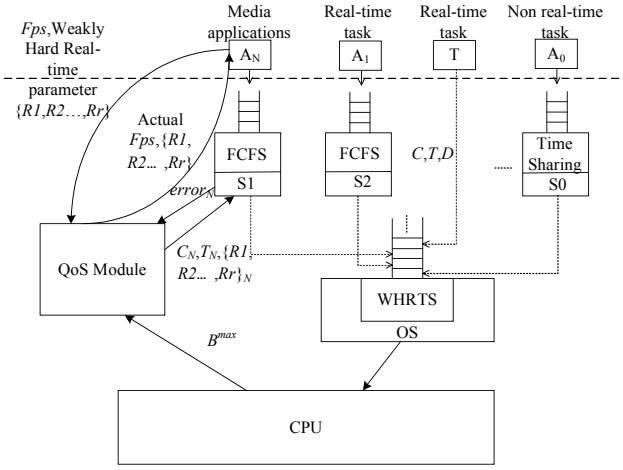

Fig. 6 QoS model

Application program can also to manage module send the corresponding parameter, which can be turn into the server parameter by manage module. For example to play video, can use Fps parameter, manage module can know their period from it or to appoint the weakly hard real-time parameter, it can be adjusted according to the situation of the bandwidth.

There are many kinds of QoS parameters, especially in multimedia application, there are frame speed, compressing rate, size, form, etc. Changes of these parameters will influence C, T, $\{R_1,R_2\ldots,R_r\}$ parameter directly. For example the frame relates to T rapidly, the compressing rate relates to C etc. It is generally speaking these parameters reflect the need bandwidth of real-time tasks, they can be written as B_i^{req} .

V. Experimental Results

In this section, we present some experimental results to verify the performance of our approaches. We performed some simulation on Linux. The most interesting part is the QoS performance of WHRTCBS on an open real-time system.

The frame rate is adopted as the performance benchmark on the same task set with different soft loads. The weakly hard real-time constraints is (7,10).

In order to handle the budget exhausted event, the budget of the running task must be decreased by the system while the task executes. The simplest solution is to divide the time in ticks and assign each tick to a CBS, which budget is decreased by 1.But all the times in the system must be multiple of a system tick. We changed the Mplayer to test the performance of the server at varied loads.

When a CBS server is running in Linux, several Mplayers are playing with recording the frame rate (fps: frames per second).

A. The effect on frame rate by varied overloads

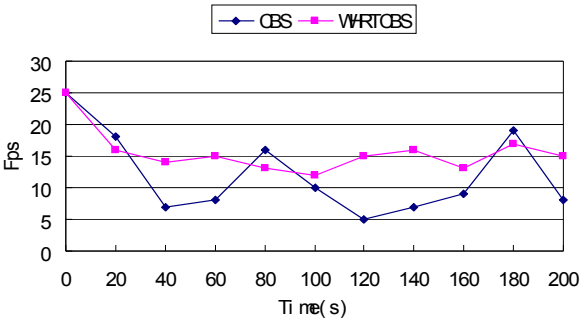

Fig. 7 Heavy overloads

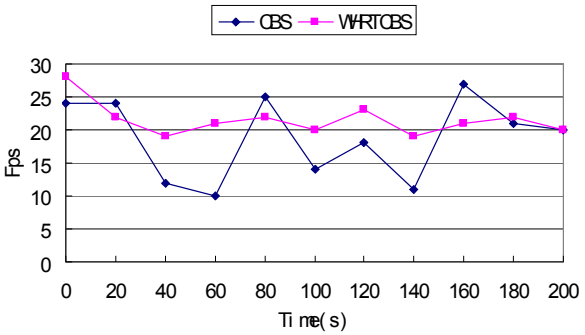

Fig. 8 Middle overloads

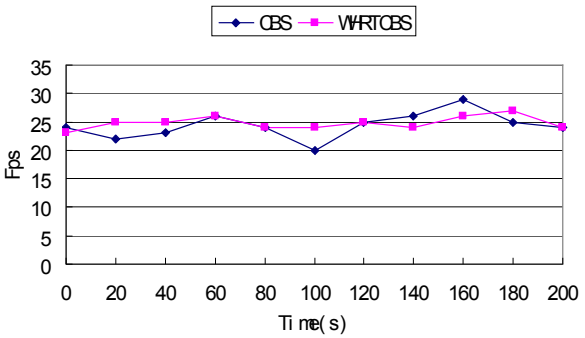

Fig. 9 Light overloads

Fig.7 shows the result by the heavy overloads. The frame rate is averagely dropped and stable on new CBS. But the one on the old CBS is more variety and instable. So the QoS of WHRTCBS is better than the one of original CBS.

Fig.8 shows the effect on fps by the middle overloads. It

should be clear in Fig. 8 that the result is almost the same as that at the heavy overload. The QoS of the old CBS has larger wave motion. We can also see from the Fig.8 that the frame rates of the old CBS have larger variance than those of the WHRTCBS. Thus Fig.8 shows that the WHRTCBS has better performance than the CBS server.

Fig.9 shows the effect on fps by the light overloads. We can observe that the performance of the two CBS is almost the same.

From the above analysis, we can conclude that the difference between the CBS server is increased when CPU utilization is increasing. This is because the new CBS server uses the weakly hard real-time scheduler. The weakly hard real-time scheduler has the ability to adjust the distribution of violations by effectively utilizing the fact that a practical application can tolerate some violations of temporal constraint under certain distribution. Thus, tasks may have stable degradation in quality of service when the real-time system is overloaded.

B. The adaptive QoS experiment

The weakly hard real-time constrained parameter is modified when the real-time system is running. The real-time system is middle overloads. The experiment is that the weakly hard real-time constrained parameter is modified from (0,10) to (10,10). The result is showed at Fig. 10. It is easy to see that the frame rates is quickly modified when the parameter is changed. The frame rate is changed from 12fps to 24 fps. The experiment show the weakly hard real-time constrained parameter is a important parameter to adjust the QoS of a Mplayer.

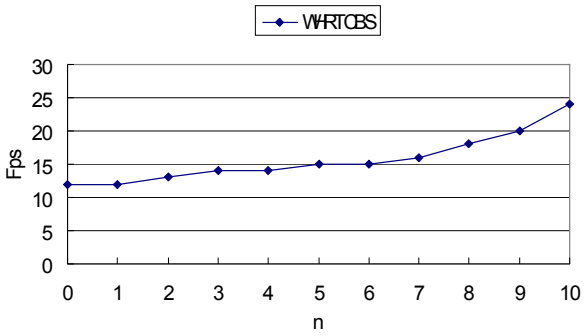

Fig. 10 Adaptive QoS experiment

VI. Conclusion

In this paper we have presented weakly hard real-time combination constrains and proposed a new CBS algorithm, which uses weakly hard real-time to reduce the variance in all tasks in open real-time systems. We analyzed some properties of combination constrains and WHRTCBS. We also presented an evaluation of our method that demonstrates its effectiveness in QoS support. The experiments show the new CBS algorithm has a good performance when CPU utilization is larger than 1. Moreover, when CPU utilization is changed, the performance of the WHRTCBS is also good because it has some adaptive ability.

Recently, there has been increasing interest that incorporates the Dynamic Voltage Scaling (DVS) techniques in real-time scheduling to deal with the power/energy conservation with regard to QoS constraints. This method can be generalized to apply to these scheduling algorithms.

References

[1] C L Liu and J W Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment," *Journal of the ACM* ,20(1), 1973: 40-61

[2] Liu J W S, *Real-Time Systems*.Upper Saddle River: Prentice Hall, 2000

[3] Nimal Nissanke, *Real-time System*. Prentice Hall, 1997

[4] M Hamdaoui and P Ramnathan, "A Dynamic Priority Assignment Technique for Streams with (m,k)-Firm Deadlines," *IEEE Transactions on Computers*, 1995, 44(12): 1443-1451

[5] G. Bernat and A. Burns, "Combining (n,m)-hard deadlines and dual priority scheduling," In *RTSS*, Dec 1997.

[6] G. Bernat and R. Cayssials, " Guarantted on-line weakly-hard real-time systems," In *RTSS*, 2001.

[7] Richard West and Karsten Schwan, "Dynamic Window-Constrained Scheduling for Multimedia Applications," In *6th Proceedings of the IEEE International Conference on Multimedia Computing and Systems (ICMCS)*, Florence, Italy: IEEE.1999.87~91

[8] Richard West and Christian Poellabauer, "Analysis of a Window-Constrained Scheduler for Real-Time and Best-Effort Packet Streams," In: *Proc of the 21st IEEE Real-Time Systems Symposium (RTSS)*, Orlando: IEEE Computer Society, 2000.239~248

[9] G Bernat,A Burns and A Liamosi, "Weakly Hard Real-Time Systems," *IEEE Transactions on Computers*, 2001 50(4): 308~321

[10] G Bernat, "Specification and Analysis of Weakly Hard Real-Time Systems," PhD thesis, Universitat de les Illes Balears, Spain, 1998

[11] Deng Z, Liu JWS, Sun J, "A scheme for scheduling hard-real-time applications in open environment," In: *Proceedings of the 9th Euromicro Workshop on Real-Time Systems*, Los Alamitos, CA: IEEE Computer Society Press, 1997: 155-185.

[12] Z. Deng and J. W. S. Liu, "Scheduling real-time applications in open envirovment," In *IEEE Real- Time Systems Symposium*, San Francisco, December 1997.

[13] G. Lipari and G.C. Buttazzo, " Scheduling real-time multi-task applications in an open system," In *proceeding of the 11th Euromicro Workshop on Real-Time Systems*, York, UK, June 1999

[14] Giuseppe Lipari and Sanjoy Baruah, "Efficient scheduling of real-time multi-task applications in dynamic systems," *Proceedings of the Real-Time Technology and Applications Symposium*, pp 166-175, Washington, DC. May 2000. IEEE Computer Society Press.

[15] Luca Abeni and Giorgio Buttazzo, "Integrating multimedia applications in hard real-time systems,"In *proceedings of the IEEE Real-Time Systems Symposim*, Madrid, Spain, December 1998.

[16] Luca Marzario ,Giuseppe Lipari Patricia ,Balbastre and Alfons Crespo, " IRIS: A new reclaiming algorithm for server-based real-time systems," *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium*, Toronto: IEEE Computer Society Press ,May 2004:211-218

[17] A.K.Mok and W.Wang, "Window-constraint real-time periodic task scheduling," In *RTSS*, 2001.