International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM)
ISSN: 2150-7988 Vol.1 (2009), pp.287-294
http://www.mirlabs.org/ijcisim

An Adaptive Fuzzy Scheduling and Fuzzy-PID Performance Control Model Suitable for Soft Real-time Systems

Xian-Bo He, Li Yang, Xian-hua Zeng, Gang-yuan Zhang, Xing-qiang Zhao

School of Computer Science, China West Normal University,

NanChong, SiChuan, China

hexianbo_sctc@163.com

***Abstract*:** In a soft real-time system, the deadline missing ratio is an important metric to evaluate system performance. In addition, considering the unsteadiness and unpredictability of a practical task running environment due to the unsteadiness of network communication and the estimation deviation, it is necessary to introduce fuzzy concept and theory to the scheduling and performance control of the soft real-time application systems oriented to communication and network fields. In this paper, a new adaptive soft real-time task scheduling model with the fuzzy-PID feedback controller was presented. In this model, we used the fuzzy scheduling algorithm in which the scheduling turn of a ready task is decided by the fuzzy inference result of its criticality degree and deadline distance. Meanwhile, in the adaptive control part, we used the fuzzy-PID controller that combined the fuzzy feedback controller and PID controller instead of traditional PID controller. The simulation test shows that our presented model can enable a soft real-time system with multi-level service characteristic reach steady state faster and has less miss ratio to its more important tasks.

***Keywords*:** Fuzzy feedback control, Deadline missing ratio, CPU utilization, PID.

I. INTRODUCTION

In a soft real-time system, it is tolerable that some task instances miss their deadlines. So the deadline missing ratio, which is defined as the number of deadline misses divided by the total number of task instances in a sampling period, is the most important performance metric to a soft real-time system [1][2][8]. Meanwhile, in many practical soft real-time systems, their workloads are commonly time varying due to the uncertain characteristics of tasks such as execution times and deadline distances, especially in the unpredictable environments such as the online trading and e-business server. In addition, some soft real-time applications take on the characteristic of multi-level service such as web services [3], multimedia [4], and systems that support imprecise computation [5]. When their workloads vary abruptly, the deadline miss ratios of these soft real-time systems may be adaptively maintained near the performance reference by upgrading or degrading their service levels of some tasks.

Considering the unsteadiness and unpredictability characteristics of the practical task running environment in a communication field due to the unsteadiness of network communication and the evaluation deviation, it is necessary to introduce fuzzy concepts and theory to the scheduling and performance control field of a soft real-time system. The characteristics of a soft real-time task, such as the criticality and the deadline distance, are more suitable to describe with fuzzy concepts, also. On the researches [1][3][4][6] about adaptive control of a soft real-time system performance, they seldom considered the following important factors:

1) The criticality or importance degrees of all tasks are not same. We should reduce the deadline miss ratios of more important tasks or let more important tasks run in the higher service level in order to maximize the value of the whole soft real-time system.

2) The fuzzy feedback control [9][10][11] and PID feedback control [1][2][8] have different advantages and disadvantages, respectively. Combining their advantages can achieve better performance control effect.

In our research, we present a new adaptive soft real-time task scheduling model with fuzzy-PID feedback controller. In this model, we adopt the fuzzy scheduling algorithm in which a ready task's scheduling turn is decided by the fuzzy inference result of its criticality and deadline distance. Meanwhile, in the adaptive control part, we introduce the fuzzy-PID controller instead of the traditional PID controller.

II. TASK MODEL

The soft real-time systems we study are mainly composed of periodic tasks that have several service levels. Every periodic task τ is described with a tuple *(E,D,P)*. *P* is the period of task τ; *D* is the relative deadline of task τ, assuming that *D* equals *P*; *E* is the execution time characteristic of task τ, and $E = \{(WCET_k, BCET_k, V_k, EET_k, AET_k) | 0 \leq k \leq m\}$,m(m$\geq$*1)* is the number of service levels task τ has. If *m* equals *1*, the task is a normal task that has two service levels (corresponding to the rejection and the admission, respectively). A higher service level of a task commonly

needs a longer (both estimated and actual) CPU time to execute, and contributes a higher value if it meets its deadline. $WCET_k$ is worst case execution time of the service level k of task τ; $BCET_k$ is the best case execution time; EET_k is the estimated average execution time $(BCET_k \leq EET_k \leq WCET_k)$; AET_k is the actual average execution time $(BCET_k \leq AET_k \leq WCET_k)$. A task's actual execution time is time varying and unknown to the scheduler. V_k is the value which contributes to system when task τ meet its deadline in its service level k.

,

III Basic Architecture

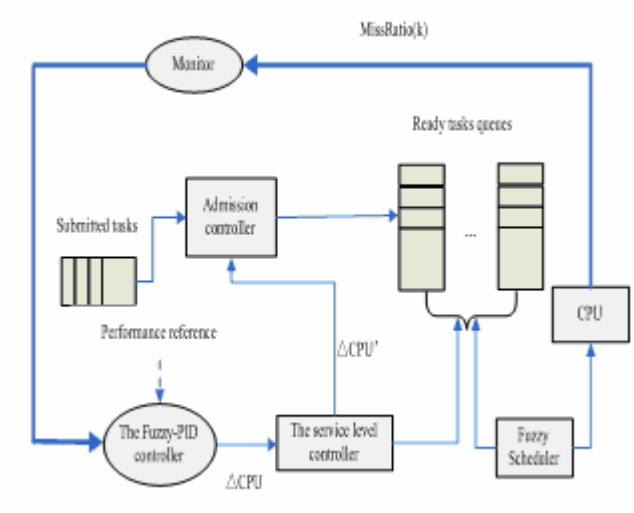

Fig.1. Architecture of the adaptive fuzzy-PID feedback control with fuzzy scheduler model

Our fuzzy scheduling and feedback control model (Figure 1) is mainly composed of five components: an admission controller, a fuzzy-PID controller, a monitor, a service level controller, and a fuzzy scheduler. The submitted new tasks enter the ready queue to wait to be scheduled by fuzzy scheduler. After monitor gets the deadline missing ratio in the sampling window and figures out the deviation from the performance reference of the deadline miss, the fuzzy- PID controller works out the CPU gain based on the performance deviation. Then, the task service level controller tunes the service level according to the CPU gain ΔCPU or accepts new tasks by admission controller according to the CPU gain $\Delta CPU'$ to adaptively tune system's workload to maintain the performance near the desirable system performance reference.

IV Fuzzy Scheduling Algorithm

A scheduling algorithm is a set of rules that determine the task to be executed at the scheduling moment. Traditional scheduling algorithms are commonly based on the precise quantity of unique task characteristics. For example, RMS algorithm (classic static scheduling algorithm) [6] is based on the period of a task, and EDF algorithm (classic dynamic scheduling algorithm) [6] is based on the deadline distance of a task. In a practical unpredictable soft-time system, it is not suitable to describe a task's characteristics by a precise quantity[13]. In our model, the scheduling turn of a task is decided by the fuzzy inference of the mixed characteristics of the criticality (static task characteristic) and the deadline distance (dynamic task characteristic) of a task. Figure 2 illustrates the inference architecture of the fuzzy scheduler.

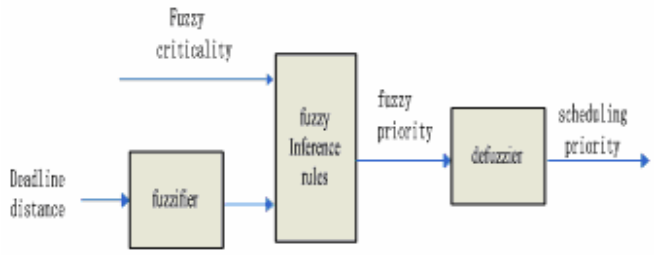

Fig. 2. The architecture of fuzzy inference

4.1 Fuzzy Inference

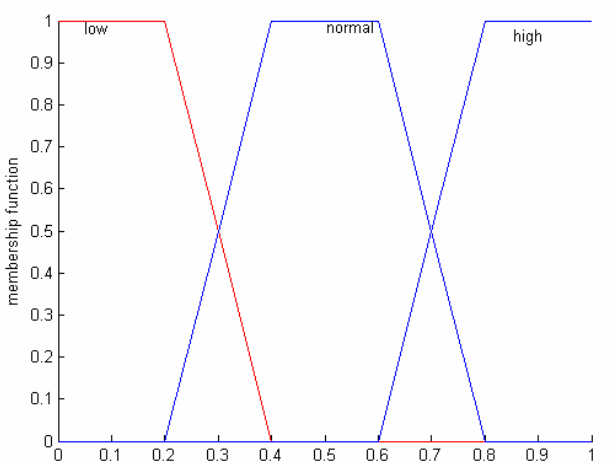

Fig.3. Membership function of criticality

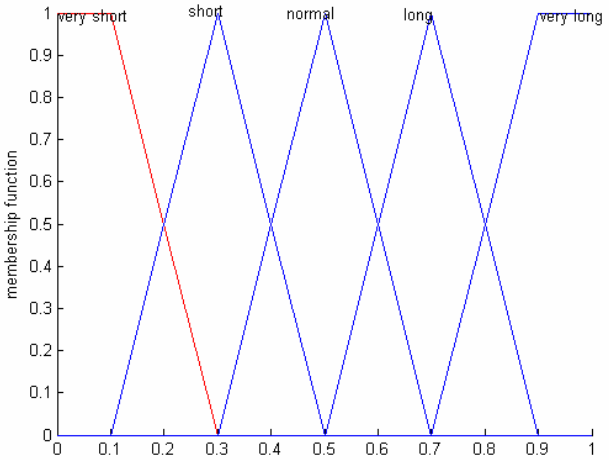

Fig. 4. Membership function of deadline distance

In our fuzzy scheduling inference model (Figure 2), we choose the deadline distance and criticality as the input fuzzy variable. In fuzzy inference, the fuzzy partition sets of the criticality $T(I_{cf}) = \{low, normal, high\}$ and figure 3 is its membership function. The fuzzy partition sets of the deadline distance $T(I_{df}) = \{very\ short, short, normal,$

long ,very long}, and figure 4 is its membership function. The fuzzy inference results mean the scheduling priority sub queues that a ready task may enter. Its fuzzy partition sets $T(O_p)$= *{high ,normal, low}* and figure 5 is its membership function.

Table 1 is the fuzzy inference rules, which includes 15 fuzzy inference rules. The rules are as follow:

R1: if (the criticality is ***low****) and (the deadline distance is* ***very short*** *) then the scheduling fuzzy priority is* ***low****;*

…

R15: if (the criticality is ***high****) and (the deadline distance is* ***very short*** *) then the scheduling fuzzy priority is* ***high****.*

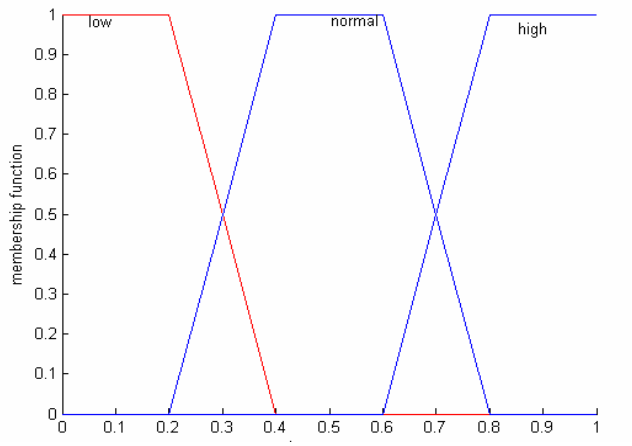

Fig. 5.. Membership function of scheduling priority

Table 1 Inference rules

$T(I_{df})$ / $T(o_p)$ / $T(I_{cf})$	very short	short	normal	long	very long
Low	low	low	low	low	Low
normal	low	normal	normal	normal	high
high	high	high	high	high	high

To fuzzify the deadline distance, we transform the deadline distance value (dd_τ denotes the value) of task τ to its fuzzy set according to following steps:

1) Domain transformation:

$rela\_deadline\_dist = dd_\tau / D_\tau$, D_τ is the period of task τ.

2) Fuzzification: if *rela_deadline_dist* is in *U*, we fuzzify it by singleton. For example, if *rela_deadline_dist* equals 0.2，its corresponding fuzzy set is:

$$\{\frac{0.0}{0.0},\frac{0.0}{0.1},\frac{1.0}{0.2},\frac{0.0}{0.3},\frac{0.0}{0.4},\frac{0.0}{0.5},\frac{0.0}{0.6},\frac{0.0}{0.7},\frac{0.0}{0.8},\frac{0.0}{0.9},\frac{0.0}{1.0}\}$$

Otherwise, we fuzzify it using linear proportion method. For example, if *rela_deadline_dist* equals 0.23，its corresponding fuzzy set is:

$$\{\frac{0.0}{0.0},\frac{0.0}{0.1},\frac{0.7}{0.2},\frac{0.3}{0.3},\frac{0.0}{0.4},\frac{0.0}{0.5},\frac{0.0}{0.6},\frac{0.0}{0.7},\frac{0.0}{0.8},\frac{0.0}{0.9},\frac{0.0}{1.0}\}$$

To defuzzify the output value and match the input fuzzy partition sets, we adopt the similarity nearness degree (SND) to decide which fuzzy set in fuzzy partition sets is selected corresponding to a special fuzzy set input or output. The similarity nearness degree (SND) is defined as follows:

$$SND(A,B)=1/2[A\cdot B+(1-A\odot B)] \quad (1)$$

In formula (1), "." is the operator to compute the inner product of two fuzzy sets A and B， and "⊙" is the operator to compute the exterior product of two fuzzy sets A and B.

In our fuzzy scheduling model, we get the fuzzy priority by applying fuzzy inference principle (the inference rules see table 1, the input fuzzy variable partitions is $T(I_{cf})$ and $T(I_{df})$, using SND to match the items of the fuzzy inference rules table) ; a task enter one of three corresponding fuzzy priority ready sub queues according to the similarity nearness degree to elements of $T_{(Op)}$.

4.2 Fuzzy Scheduling Policy

In our fuzzy scheduling model, the following is our scheduling policy:

1) To the tasks in different fuzzy priority ready sub-queues, the tasks which in higher fuzzy priority ready sub queue will be scheduled first.

2）In the same fuzzy priority sub queue, we adopt the EDF[6] scheduling policy, namely the task who has the shortest deadline distance will be scheduled first.

3）If there is a higher fuzzy priority task is ready, the scheduler preempts the current task' running right, namely we adopts the preemptive scheduling policy.

V FUZZY FEEDBACK CONTROLLERS

In a soft real-time system that has no adaptive control mechanism, the research on task scheduling is commonly based on the worst case of tasks' characteristics, such as the worst execution time. These pessimistic scheduling algorithms [6][7] may cause the system resources underutilization. On the research of adaptive control of a soft real-time system performance, papers [1][2][8] introduce traditional automatic control theory ,especially PID feedback control theory, to soft real-time tasks scheduling field; papers [9][10][11] introduce fuzzy control theory to scheduling field of soft real-time tasks. In our model, we combine the fuzzy feedback control and PID control to adaptively control the deadline miss ratio near the desirable system performance reference. Meanwhile, we introduce the fuzzy control decision table instead of fuzzy reference computation [9][10][11] to reduce CPU computation overhead.

To apply automatic control theory to soft real-time tasks scheduling, we need to decide what are the controlled variable, performance reference and the manipulated variable. In our feedback control architecture model, we choose the deadline miss ratio *MissRatio(k)* as the controlled variable. The miss ratio *MissRatio(k)* is defined as the number of deadline misses divided by the total number of completed and aborted tasks in a sampling

window *((k-1)W, kW)* ,where *W* is the sampling period and *k* is called the sampling instant. Performance references represent the desired system performances in terms of the controlled variables. In our feedback control architecture model, it is the desired deadline miss ratio M_S. In consideration of better CPU utilization and more tasks running in higher service level. The performance reference value of a soft real-time is commonly greater than zero. In the simulation test, we choose 0.15 as the performance reference. Manipulated variables are the system attributes that can be dynamically changed by the scheduler to affect the values of the controlled variables. In our architecture model, the manipulated variable is the total average estimated utilization $B(k) = \sum_i U_i[l_i(k)]$ of all tasks in ready queues of a soft real-time system, where $U_i[l_i(k)]$ is the average estimated CPU utilization of task *i* with the service level *k* in the *kth* sampling window, i.e. $U_i[l_i(k)] = \frac{EET_k}{P}$ (EET_k is the average estimated execution time of task *i*, and *P* is its period). The rationale for choosing the total average estimated utilization as a manipulated variable is that deadline miss ratio increases or decreases as the system load increases or decrease when the CPU is overloaded. However, the actual total CPU utilization is often different from the total average estimated utilization *B(k)*, which is due to the estimation error of execution times when workload is unpredictable and time varying.

5.1 Fuzzy Feedback Controller

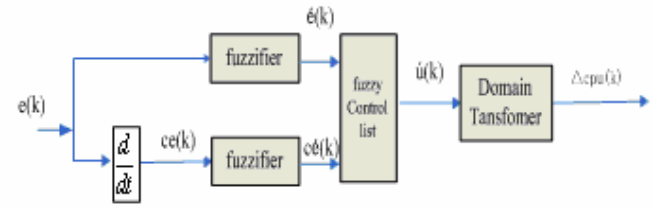

Fig. 6. Basic architecture of fuzzy controller.

Figure 6 is the architecture model of the fuzzy controller in our fuzzy-PID controller. In this figure, *e(k)* and *ce(k)* are the deviation and deviation differential which are gotten in the *kth* sampling window *((k-1)W , kW)*. The formula of *e(k) and ce(k)* in discrete form is as follow:

$$e(k) = MissRatio(k) - M_s,$$

$$ce(k) = \frac{(e(k)-e(k-1))-(e(k-1)-e(k-2))}{W}$$

By a fuzzifier, *e(k)* and *ce(k)* become *é(k)* and *cé(k)* which are used to search in offline fuzzy control decision table to get the output *ú(k)* quickly.

5.2. Fuzzifier

To use offline fuzzy control decision list to reduce fuzzy computing overhead, *e(k)* and *ce(k)* adopt singleton fuzzifier after domain transform. The discrete fuzzy domain set we choose is *U = {-6,-5,-4,-3,-2,-1,,0,1,,2,3,4,5,6}*.Table 2-1 and table 2-2 are the domain transform and fuzzifying method.

Table 2-1. The fuzzifier of *e(k)* and *ce(k)/2*

≥0.5	[0.2, 0.5)	[0.1, 0.2)	[0.05, 0.1)	[0.03, 0.05)	[0.01, 0.03)	[-0.01, 0.01)
-6	-5	-4	-3	-2	-1	0

Table 2-2. The fuzzifier of *e(k)* and *ce(k)/2*

[-0.03, -0.01)	[-0.06, -0.03)	[-0.1, -0.06)	[-0.15, -0.10)	[-0.30, -0.15)	< -0.30
1	2	3	4	5	6

5.3 Fuzzy Control Table

Table 5 is the fuzzy inference rules table in which the input fuzzy variables are *é* and *cé*, and the output fuzzy variable is *ú*. In the inference rules, *T(é) = {NB ,NM, NS, NZ, PZ, PS, PM, PB}* is the fuzzy partition sets of fuzzy variable *é*,and table 6-1 and 6-2 is its membership function. *T(cé) =T(ú) = {NB,NM,NS,ZE,PS,PM,PB}* are the fuzzy partitions of the fuzzy variable *cé* and *ú*, and table 3-1and 3-2 are their membership function. In these fuzzy partition sets, *NB*, *NM*, *NS*, *NZ*, *ZE, PZ*, *PS*, *PM* and *PB* denote the fuzzy sets negative big, negative middle, negative small, negative zero(-0), zero(0), positive zero(+0), positive small, positive middle and positive big, respectively. According to the membership function table 3-1,3-2,6-1,6-2 and the fuzzy inference rules table 5, we get the final offline fuzzy control decision table (table 7-1 and 7-2) by using Mamdani fuzzy inference principle and weighted average [13] to defuzzify them.

5.4 CPU Utilization Gain of the Fuzzy Controller

When we get the output by look up in the fuzzy control decision table by inputs *é(k)* and *cé(k)*, the output is commonly not used to directly control the object system. We have to transform it to object domain. The CPU utilization gain is acquired after transforming from the output $\acute{u}(k)$ by the table 4-1 and 4-2.

Table 3-1. Membership function of *T(cé)* and *T(ú)*

U / μ / T(cé)	-6	-5	-4	-3	-2	-1
NB	1.0	0.7	0.3	0.0	0.0	0.0
NM	0.3	0.7	1.0	0.7	0.3	0.0
NS	0.0	0.0	0.3	0.7	1.0	0.7
ZE	0.0	0.0	0.0	0.0	0.3	0.7
PS	0.0	0.0	0.0	0.0	0.0	0.0
PM	0.0	0.0	0.0	0.0	0.0	0.0
PB	0.0	0.0	0.0	0.0	0.0	0.0

Table 3-2. Membership function of *T(cé)* and *T(ú)*

U / μ / T(cé)	0	1	2	3	4	5	6
NB	0.0	0.0	0.0	0.0	0.0	0.0	0.0
NM	0.0	0.0	0.0	0.0	0.0	0.0	0.0
NS	0.3	0.0	0.0	0.0	0.0	0.0	0.0
ZE	1.0	0.7	0.3	0.0	0.0	0.0	0.0
PS	0.3	0.7	1.0	0.7	0.3	0.0	0.0
PM	0.0	0.0	0.3	0.7	1.0	0.7	0.3
PB	0.0	0.0	0.0	0.1	0.3	0.7	1.0

Table 4-1 CPU utilization gain transform table

≤−4	(−4, −2]	(−2, −1]	(−1, −0.5]	(−0.5, 0]
−0.8	−0.6	−0.3	−0.05	−0.01

Table 4-2 CPU utilization gain transform table

(0, 0.5]	(0.5, 1]	(1, 3]	(3, 4]	> 4
0.01	0.05	0.10	0.30	0.50

Table 5. Fuzzy inference rules

T(cé) / T(ú) / T(é)	NB	NM	NS	ZE	PS	PM	PB
NB	NB	NB	NB	NB	NM	ZE	ZE
NM	NB	NB	NB	NB	NM	ZE	ZE
NS	NM	NM	NM	NM	ZE	PS	PS
NZ	NM	NM	NS	ZE	PS	PM	PM
PZ	NM	NM	NS	ZE	PS	PM	PM
PS	NS	NS	ZE	PM	PM	PM	PM
PM	ZE	ZE	PM	PB	PB	PB	PB
PB	ZE	ZE	PM	PB	PB	PB	PB

Table 6-1 Membership function of *T(é)*

U / μ / T(é)	-6	-5	-4	-3	-2	-1
NB	1.0	0.8	0.7	0.4	0.1	0.0
NM	0.2	0.7	1.0	0.7	0.3	0.0
NS	0.0	0.1	0.3	0.7	1.0	0.7
NZ	0.0	0.0	0.0	0.0	0.1	0.6
PZ	0.0	0.0	0.0	0.0	0.0	0.0
PS	0.0	0.0	0.0	0.0	0.0	0.0
PM	0.0	0.0	0.0	0.0	0.0	0.0
PB	0.0	0.0	0.0	0.0	0.0	0.0

Table 6-2 Membership function of *T(é)*

U / μ / T(é)	0	1	2	3	4	5	6
NB	0.0	0.0	0.0	0.0	0.0	0.0	0.0
NM	0.0	0.0	0.0	0.0	0.0	0.0	0.0
NS	0.2	0.0	0.0	0.0	0.0	0.0	0.0
NZ	1.0	0.0	0.0	0.0	0.0	0.0	0.0
PZ	1.0	0.6	01	0.0	0.0	0.0	0.0
PS	0.2	0.7	1.0	0.7	0.3	0.1	0.0
PM	0.0	0.0	0.2	0.7	1.0	0.7	0.3
PB	0.0	0.0	0.1	0.4	0.7	0.8	1.0

Table 7-1. Fuzzy control decision table

cé / ú / é	-6	-5	-4	-3	-2	-1
-6	−5.35	−5.23	−5.35	−5.23	−5.35	−5.23
-5	−5.00	−4.95	−5.00	−4.95	−5.00	−4.95
-4	−4.69	−4.52	−4.69	−4.52	−4.69	−4.52
-3	−4.26	−4.26	−4.26	−4.26	−4.26	−4.26
-2	−4.00	−4.00	−3.78	−3.76	−3.47	−3.42
-1	−4.00	−4.00	−3.63	−3.08	−2.47	−2.12
0	−3.59	−3.56	−2.93	−2.60	−0.96	−0.51
1	−2.92	−2.92	−2.33	−1.91	−0.26	1.04
2	−1.81	−1.79	−0.57	−0.31	0.44	1.79
3	1.00	1.00	−0.25	0.94	1.42	2.29
4	−0.58	−0.64	0.69	1.42	1.94	2.93
5	0.23	0.24	1.12	1.79	2.36	3.71
6	0.0	0.0	1.29	2.00	2.71	4.26

Table 7-2. Fuzzy control decision table

cé / ú / é	0	1	2	3	4	5	6
-6	−4.69	−4.26	−2.71	−2.00	−1.29	0.0	0.0
-5	−3.86	−3.71	−2.36	−1.79	−1.12	0.24	0.23
-4	−3.05	−2.93	−1.93	−1.42	−0.69	0.64	0.58
-3	−2.93	−2.90	−1.42	−0.94	0. 26	1.00	1.00
-2	−2.43	−1.79	0.44	0.004	0.16	1.60	1.63
-1	−250	−.1.05	0.26	1.91	2.33	2.92	2.92
0	0	0.51	0.96	2.60	2.93	3.55	3.59
1	1.50	2.12	2.47	3.01	3.64	4.00	4.00
2	2.43	3.42	3.47	3.76	3.78	4.00	4.00
3	2.93	4.26	4.26	4.26	4.26	4.26	4.26
4	3.05	4.52	4.69	4.52	4.69	4.52	4.69
5	3.86	4.95	5.00	4.95	5.00	4.95	5.00
6	4.69	5.11	5.24	5.11	5.24	5.11	5.24

5.5 PID controller

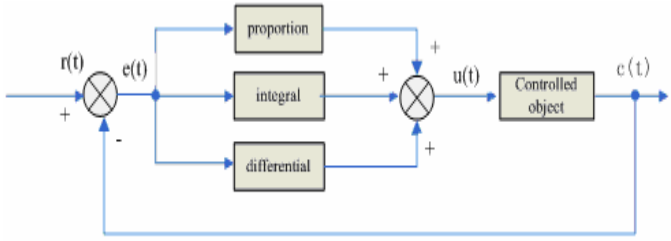

Fig. 7. The control architecture model of PID

In the auto-control field, the PID control architecture is illustrated in figure 7. In this figure, *e(t)* is the differential of reference value r(t) and actual output value c(t), i.e. e(t) = r(t)-c(t). By the linear combination of proportion, integral and differential of deviation *e(t)*, the PID controller produces the control output value to controlled object system. A basic PID control formula is as the follow:

$$u(t) = K_P e(t) + K_I \int_0^t e(t)dt + K_D \frac{de(t)}{dt} \quad (2)$$

In our model (figure 1) the discrete digital PID formula [8] is as the follow:

$$\triangle CPU(k) = K_P e(k) + K_I \sum_{IW} e(i) + K_D \frac{e(k) - e(k-1)}{W} \quad (3)$$

In the formula (2) (3), K_P is proportion coefficient; K_I is the integral time coefficient; K_D is the differential time coefficient. *IW* is the time window for integral (in the practical control system. to avoid counting overflow, using the most recent sampling data instead of integral of whole running time); *e(i)* in formula (3) is the i^{th} sampling deviation in the integral window; *W* is the sampling period, and the PID performance control does once per *W* time units.

5.6 Fuzzy-PID Controller

When the workload of a real-time system abruptly changes in a large step, it needs a longer time to reach steady sate using PID feedback control. On the contrary, when the workload changes little, it easily causes to overshoot. So, the fuzzy-PID control, which is the combination of the fuzzy feedback control and PID feedback control, should be a better feedback control technology to adaptively adjust a soft real-time system's performance. When the deviation of the current performance value is greater than the set threshold value, fuzzy feedback control is chosen, or PID feedback does the same work. The following is the pseudo code of the fuzzy-PID feedback controller，which is called in every sampling period *W*.

```
void Fuzzy_PID()
{
 Get MissRatio(k) during last sampling period kW;
 e(k) = MissRatio (k) - M_S ;
 ce(k) = (e(k) - e(k-1)) / W
 if ((abs(e(k)) < M_threshold)
 {
   /*PID control function, abs(e(k)) is the absolute value of
   e(k) */
   ΔCPU(k) = K_P e(k) + K_I Σ_IW e(i) + K_D ce(k);
 } els e {
  /* fuzzy control */
  Get the é(k) and cé(k) by fuzzier;
   Get the output value ú(k) by looking up in the fuzzy
   decision control table using the inputs é(k) and cé(k);
   Get the ΔCPU(k) by domain transformer using the
   input ú(k);
   }
Call the service level controller and the admission
controller to tune the manipulated variable;
}
```

In the simulation test, the values of K_P, K_I , K_D , *W* and *IW* are *0.5,0.05,0.1,2400*(time unit) and *100*2400*(time unit) which mainly refer to paper [8]; the M_S and $M_{threshold}$ are set to *0.15* and *0.05*, respectively.

5.7 Service Level Controller and Admission Controller

The service level controller changes the requested utilization in the system by adjusting the service levels of tasks in ready queue. When gotten the CPU utilization gain $\triangle CPU(k)$ in the sampling instant *KW* by the fuzzy-PID controller, the next step is to tune the requested utilization by service level controller and the admission controller. If $\triangle CPU(k) > 0$, The service level controller and the admission controller upgrade the service levels of accept tasks service level or admit the tasks in submitted task queue (if all accept tasks running in the highest service level); 2) degrade the service level if $\triangle CPU < 0$. Pseudo code of service level controller is as follow [8]:

```
double SLC(ΔCPU（k）)
{
 tmpCPU=ΔCPU（k）;
 if (tmpCPU < 0)
{
   While (tmpCPU <0 && Exist Degradable Task) {
  τ = Select_Degraded_Task();
 ChangeSeviceLevel( τ,current_level,new_level);
 tmpCPU = tmpCPU – EET_current_level + EET_new_level ;
 } else {
     While (tmpCPU > 0 && Exist Upgradable Task){
     τ = Select_Upgraded_Task();
     ChangeSeviceLevel( τ,current_level,new_level);
     tmpCPU = tmpCPU - EET_new_level + EET_current_level ;
     }
    if (tmpCPU >0 ){
      while (tmpCPU > 0 && Exist tasks in submitted
      queue ){
```

```
            τ = Select_Task_From_Submitted_Queue(level);
            AddToReadyQue( τ );
            tmpCPU = tmpCPU - EET_level ;
        }
    }
}
```

VI Experiment and Conclusion

6.1Workload Model

To compare the result of the PID adaptive control model and our fuzzy-PID adaptive control model, our simulation test environment refers to that of paper [8]. Every test period task has three service levels corresponding to its rejection, low service level and high service level, respectively. In the tuple *(S，E， D，T)* of every task , *E* is defined as *{(*$WCET_k$*,*$BCET_k$*,* V_k*,*EET_k*,*AET_k*)| 0⩽ k ⩽ 2}. *$WCET_1$ *= Random(8,16), P=T=* $WCET_1$ ** Random(10,15),* $WCET_2$ *=* $WCET_1$** 0.5;* $BCET_k$ *=* $WCET_k$ ** 0.25 ,* EET_k *= (*$WCET_k$ *+* $BCET_k$*) * 0.5 ,*AET_k *=* EET_k ** etf (k=1,2)*. The execution time factor *etf* can be tuned to change the accuracy of the estimation. The actual execution time of each instance (which is unknown to the scheduler) is computed as a uniform random variable in interval *[*AET_k $WCET_k$*] or [*$BCET_k$ AET_k*] (k=1, 2)* depending on a random Bernoulli trial with the probability *(*AET_k*-*$BCET_k$*) / (*$WCET_k$*-*$BCET_k$*)*. In our test experiment, we change *etf* to simulate the time varying characteristic of the actual workload of a soft real-time system. In addition, the tasks in test environment have these ideal characteristics:

1) The tasks are independent in that the request of a task does not depend on the initiation or the completion of the requests of other tasks;

2) Except the CPU, the resource of the environment is sufficient;

3) The overhead of scheduling time, the switching time is omitted;

4) When a task instance misses its deadline, it is aborted immediately.

Our test tasks fall into two categories: the mandatory tasks simulating critical tasks that must be accept and optional tasks simulating the tasks that can be accept by the admission or rejected by the service level controller when the actual workload changes. In the initial state of our test experiment, the total average estimated execution time workload of all mandatory tasks is 1 and that of the optional tasks is 2.

In our simulation test experiment, the initial value of the execution factor *etf* is *0.5*. Then, in the 100^{th} 200^{th} *and* 300^{th} sampling instants, *etf* changes to 1.6, 1.3 and 0.9, respectively.

6.2 Experiment Result and Conclusion

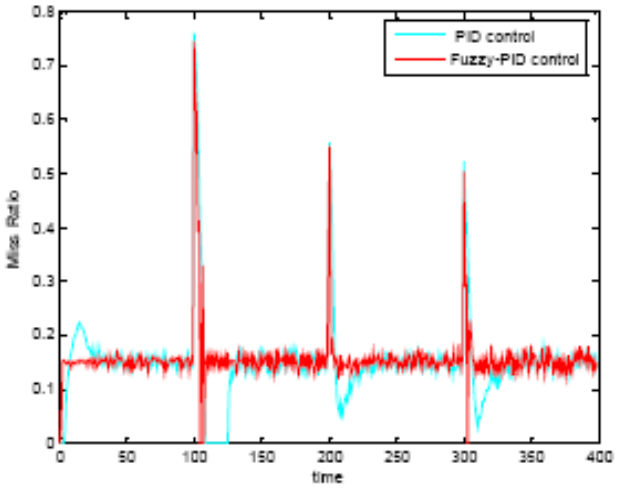

Fig. 8. Experiment result contrast

Table 8. Deadline miss ratio comparison of Fuzzy-PID controller and PID controller

	high	normal	low
Fuzzy-PID controller (Fuzzy scheduling)	0.0016	0.0041	0.376
PID controller (EDF scheduling)	0.097	0.106	0.103

Figure 8 and table 8 are our test experiment result. In figure 8, the red curve denotes the deadline miss ratio variances of our adaptive fuzzy-PID performance control and the blue curve denotes that of the adaptive PID performance control which is presented in paper [8]. In the initial state (*etf =0.5)*, the actual workload is smaller than the estimation; in the 100^{th} sampling instant, the actual workload changes greater than the estimation abruptly (*etf* changes from *0.5* to *1.6)*; then in the 200^{th} *and* 300^{th} sampling instants, the workload varies abruptly due to *etf* change. The simulation test experiment result of figure 8 shows that our adaptive fuzzy-PID performance control model needs much less time to reach steady state than that of the adaptive PID performance control model when the actual workload of a soft real-time system varies abruptly. Table 8 also shows that the more important a task is, the less deadline miss ratio a task has in our Fuzzy-PID performance controller adopting fuzzy scheduling policy; but in the PID controller adopting EDF scheduling policy, the deadline miss ratios of all three kind tasks are very close. Of course, the fuzzy control needs CPU computation overhead. The use of the fuzzy control decision table in our model can greatly reduce the time overhead.

References

[1] Stankovic JA, Lu CY, Son SH, Tao G. “The case for feedback control real-time scheduling”, In: Proc. of

the 11th Euromicro Conf. on Real Time Systems, York, pp. 11-20, 1999.
[2] Lu C, Stankovica J A,Tao G,an Son S H. “Feedback control real-time scheduling: framework,modeling,and algorithms”, Real-Time Systems Journal, Special Issue on Control-Theoretical Approaches to Real-Time Computing, pp.58-66, 2001(1).
[3] T.F.Abdelzaher and N. Bhatti. “Web Server QoS Management by Adaptive Content Delivery”, International Workshop on Quality of Service, pp. 216-225, 1999.
[4] S. Brandt and G. Nutt. “A Dynamic Quality of Service Middleware Agent for Mediating Application Resource Usage”, IEEE Real-Time Systems Symposium, pp.307-317, December 1998.
[5] J. W. S. Liu, et. al., “Algorithms for Scheduling Imprecise Computations”, IEEE Computer, pp.58-68,24(1), No.5,May 1991.
[6] C.L.Liu, J.W.Layland. “Scheduling algorithms for multiprogramming in a hard-real-time environment”, Journal of ACM, pp. 46-61: 1973, 20(1).
[7] Lehoczky JP, Sha L, Ding Y. “The rate monotonic scheduling algorithm: Exact characterization and average case behavior”, In:Proc. of the 10th IEEE Real-Time Systems Symp, Santa Monica: IEEE Computer Society Press, pp. 166-171, 1989.
[8] Chenyang Lu John A. Stankovic Gang Taoÿ Sang H. Son. “Design and Evaluation of a Feedback Control EDF Scheduling Algorithm”,20th IEEE Real-Time Systems Symposium, pp.56-67, 1999.
[9] Lee J, Tiao A, Yen J. “A fuzzy rule-based approach to real-time scheduling”, In: Yen J, ed. Proc. of The 3rd IEEE Int’l Conf. on Fuzzy Systems, Vol 2. Piscataway: IEEE Computer Society, pp. 1394-1399, 1994.
[10] Terrier F, Rioux L, Chen Z. “Real time scheduling under uncertainty”, In: Nakanishi, S. ed. Proc. of the 4th IEEE Int’l Conf. on Fuzzy Systems, Vol 3. Piscataway: IEEE Computer Society, pp. 1177-1184, 1995.
[11] Litoiu M, Tadei R. “Real-Time task scheduling with fuzzy deadlines and processing times”, Fuzzy Set and Systems, pp. 35-45, 2001,117(1).
[12] Mojtaba Sabeghi, and Mahmoud Naghibzadeh. “A Fuzzy Algorithm for Real-Time Scheduling of Soft Periodic Tasks”, IJCSNS International Journal of Computer Science and Network Security, VOL.6 No.2A, February 2006 pp. 227~235, 2001,117(1).
[13] Li SY. “Fuzzy Control, Neurocontrol and Intelligent Cybernetics. 2th ed”, Harbin: Harbin Institute of Technology Press, pp. 254~280, 1998. (in Chinese).
[14]Xian-Bo He, Zhi-Shu Li, Ning-jiu Tang. “A New adaptive performance Feedback Control scheduling Model oriented to Embedded soft real-time System”, 2008 International Conference on embedded Software and Systems Symposia, pp.369-374,2008.

Acknowledgements

Supported by the scientific research fund of Sichuan Provincial Education Department (project No:08ZA015) and China West Normal University Startup Foundation for doctor (project NO: 08B078, 08A020).

Biographies

Xian-Bo He was born in 1971. He received his doctor degree from Sichuan University, China. Now, he is a vice professor of China West Normal University. His current research area is the embedded real-time system and communication networks.