An Adaptive Fuzzy Scheduling and Fuzzy-PID Performance Control

Model Being Suitable to Soft Real-time Systems

Xian-Bo He, Li Yang, Xian-hua Zeng, Gang-yuan Zhang, Xing-qiang Zhao
College of Computer Science
China West Normal University
NanChong, SiChuan, China
E-mail: hexianbo_sctc@163.com

Abstract—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 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 and deadline distance. Meanwhile, in the adaptive control part, we used the fuzzy-PID controller that combines 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 systems which has the multi-level service characteristic reach the steady state faster and has less miss ratio to more important tasks.

Keywords: fuzzy feedback control; deadline missing ratio; CPU utilization; PID

1 INTRODUCTION

In a soft real-time system, it is tolerable that some task instances miss their deadlines. So the deadline miss 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. Meanwhile, in many practical soft real-time systems, the workload is commonly time varying due to the uncertain characteristics of a task such as execution time and deadline distance, especially in the unpredictable environments such as the online trading and e-business server. In addition, some soft real-time systems take on the characteristic of multi-level service such as web services, multimedia, and imprecise computation. When the workload of a soft system varies abruptly, its deadline miss ratio may be adaptively maintained near the performance reference by upgrading or degrading the service levels of some tasks.

Considering the unsteadiness and unpredictability of the practical running environment in communication field 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 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 concept, also. On the researches of adaptive control of a soft real-time system performance[1][3][6], they seldom considered the following important factors:

1) The criticality or importance degree of all tasks is not the same. We should reduce the deadline miss ratios of more important tasks or let more important tasks run in the higher service level.

2) The fuzzy feedback control[3][4] and PID feedback control[1] have different advantages and disadvantages, respectively. Combining their advantages can achieve better performance control effect.

So we presented a new adaptive soft real-time task scheduling model with the 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, we introduced the fuzzy-PID controller instead of the traditional PID controller.

2 TASK MODEL

The soft real-time systems we study are mainly composed of periodic tasks that have several service levels. Every periodic task $\tau$ is described with a tuple $(E,D,P)$. $P$ is the period of task $\tau$; $D$ is the relative deadline of task $\tau$, assuming that $D$ equals $P$; $E$ is the execution time of task $\tau$, and $E = \{(WCET_k,BCET_k, E_k, AET_k) | 0 \leq k \leq m\}$, $m (m \geq 1)$ is the number of service levels task $\tau$ 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 has a longer (both estimated) CPU execution time, and contributes a higher value if it meets its deadline. WCET$_k$ is worst case execution time of the service level $k$ of task $\tau$; BCET$_k$ is the best case execution time; EET$_k$ is the estimated execution time (BCET$_k \leq$ EET$_k \leq$ WCET$_k$); AET$_k$ is the actual average execution time. A task's actual execution time is time varying and unknown to the scheduler. $V_k$ is the value contributes to system when task $\tau$ meet its deadline in the service level $k$.

3 BASIC ARCHITECTURE

Supported by Scientific Reserch Fund of Si Chuan Provincial Education Department (project NO.08ZA015) and China West Normal University Startup Foundation for doctor (08B078, 08 A 020)
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 miss ratio in the sampling window and figures out the deviation from the performance reference; the fuzzy-PID controller works out the CPU gain based on the performance deviation. The task service level controller tunes the service level according to the CPU gain \( \Delta CPU \) or accepts new tasks by admission controller according to the CPU gain \( \Delta CPU \) to adaptively tune the system’s workload to maintain the performance near the desirable system performance reference.

4 Fuzzy Scheduling Algorithm

Traditional scheduling algorithms are commonly based on the precise quantity of unique task characteristics. For example, RMS\(^{(1)}\) algorithm (classic static scheduling algorithm) is based on the period of a task, and EDF\(^{(2)}\) algorithm (classic dynamic scheduling algorithm) 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 precise quantity. In our model, the scheduling turn of a task is decided by the fuzzy inference result of the mixed characteristics of the criticality (static task characteristic) and the deadline distance (dynamic task characteristic) of a task. Figure 2 illustrates fuzzy scheduling inference model.

4.1 Fuzzy Inference

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 \( T(I_c) = \{\text{low, normal, high}\} \) and Figure 3 is its membership function. The fuzzy partition sets of the deadline distance \( T(I_d) = \{\text{very short, short, normal, long, very long}\} \), and Figure 4 is its membership function. The fuzzy inference results mean the scheduling priority sub queues and its fuzzy partition sets \( T(O_p) = \{\text{high, normal, low}\} \) and Figure 3 is its membership.

4 Fuzzy Scheduling Algorithm

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 low.

<table>
<thead>
<tr>
<th>( T(I_c) )</th>
<th>( T(I_d) )</th>
<th>( T(O_p) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>very short</td>
<td>short</td>
<td>normal</td>
</tr>
<tr>
<td>very short</td>
<td>normal</td>
<td>normal</td>
</tr>
<tr>
<td>very short</td>
<td>normal</td>
<td>high</td>
</tr>
<tr>
<td>short</td>
<td>very long</td>
<td>low</td>
</tr>
<tr>
<td>normal</td>
<td>very long</td>
<td>normal</td>
</tr>
<tr>
<td>high</td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td>high</td>
<td>normal</td>
<td>high</td>
</tr>
<tr>
<td>high</td>
<td>high</td>
<td>high</td>
</tr>
<tr>
<td>high</td>
<td>high</td>
<td>high</td>
</tr>
</tbody>
</table>

To fuzzify the deadline distance, we transform the deadline distance value \( dd \) to its fuzzy set in according to the following steps:

1) Domain transformation

\[ Rela\_deadline\_dist = dd, / D, \ \text{is the period of task} \tau \]

2) Fuzzification

If \( rela\_deadline\_dist \) is in \( U \) (assuming \( U=\{0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0\} \)), we fuzzify it by singleton. Otherwise, we fuzzify it using linear
proportion method. For example, if \( \text{rela} \_ \text{deadline} \_ \text{dist} \) equals 0.23, its corresponding fuzzy set is:

\[
\{0.0\, 0.0\, 0.7\, 0.3\, 0.0\, 0.0\, 0.0\, 0.0\, 0.0\, 0.0\, 0.0\, 0.0\, 0.0\, 0.9, 1.0\}
\]

To the criticality of a task, we directly produce a task’s criticality fuzzy set with 11 values distributed in [0,1.0] to present its important degree.

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

\[
\text{SND}(A,B)=\frac{1}{2}[A\cdot B+(1-A\cup B)]
\]

In formula (1), “\( \cdot \)” is the operator to compute the inner product of two fuzzy sets A and B. “\( \cup \)” 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 are \( T(i) \) and \( T(id) \), 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 \( T(i) \).

4.2 Fuzzy Scheduling Policy

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

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

2) In the same fuzzy priority queue, we adopt the EDF[2] 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’s running right, namely we adopt the preemptive scheduling policy.

5 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. These pessimistic scheduling algorithms[3] cause the system resources underutilization. In our adaptive performance control, we combine the fuzzy feedback control and PID control to maintain the deadline miss ratio near the desirable system performance reference. Meanwhile, we introduce the fuzzy control decision table instead of fuzzy reference computation to reduce CPU computation overhead.

In our adaptive feedback control architecture model, we choose the deadline miss ratio MissRatio(k) as the controlled variable. 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 is the desired deadline miss ratio MS. The manipulated variable is the total average estimated utilization \( B(k) = \sum U_i[\hat{l}_i(k)] \) of all tasks in ready queue of a soft real-time system, where \( U_i[\hat{l}_i(k)] \) is the average estimated CPU utilization of task i with the service level k in the kth sampling window, i.e.,

\[
U_i[\hat{l}_i(k)] = \frac{\text{EET}_k}{P} \quad (\text{EET}_k \text{ is the the average estimated execution time of task } i \text{, and } P \text{ is its period}).
\]

The rationale for choosing the total average estimated utilization as a manipulated variable is that the 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

Figure 5 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 follows:

\[
e(k) = \text{MissRatio}(k) - \mu_n
\]

\[
ce(k) = \frac{e(k) - e(k-1)}{w}
\]

By a fuzzifier, \( e(k) \) and \( ce(k) \) become \( \hat{e}(k) \) and \( \hat{ce}(k) \), which are used to search in offline fuzzy control decision table to get the output \( \hat{u}(k) \) quickly.

![Fig. 5. the basic architecture of fuzzy controller.](image_url)

5.2 Fuzzy Inference

To use offline fuzzy control decision list to reduce fuzzy computing overhead, \( e(k) \) and \( ce(k) \) adopt singleton fuzzifier after domain transformation. The discrete fuzzy domain set we choose is \( U = \{-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6\} \). Table 1 and table 2 in paper[7] are the domain transformation and fuzzification methods.

5.3 Fuzzy Control Table

Table 9 in paper [6] is the fuzzy inference rules table in which the input fuzzy variables are \( \hat{e} \) and \( \hat{ce} \), and the
output fuzzy variable is \( \dot{u} \). \( T(\dot{e}\,\dot{e}) = \{NB,NM,NS,NZ,PZ,PS,PM,PB\} \) is the fuzzy partition set of fuzzy variable \( \dot{e} \),and table 5 and table 6 in paper [6] are their membership functions. \( T(c\dot{e}) = T(\dot{u}) = \{NB,NM,NS,ZE,PS,PM,PB\} \) are the fuzzy partitions of the fuzzy variable \( c\dot{e} \) and \( \dot{u} \),and table 3 and table 4 in paper [6] are their membership functions. In these fuzzy partition sets,NB,NM,NS,NS,ZE,PS,PM and PB denote the fuzzy sets negative big, negative middle, negative small,negative zero(-0),zero(0),positive small , positive middle and positive big, respectively. According to the membership functions of \( T(\dot{e}) \), \( T(c\dot{e}) \) and \( T(\dot{u}) \) and the fuzzy inference rules (table 9 in paper [6]), we get the final offline fuzzy control decision table (table 2-1 and 2-2) by using Mamdani fuzzy inference principle and weighted average\(^{3}\) to defuzzify.

5.4 CPU UTILIZATION GAIN OF THE FUZZY CONTROLLER

When we get the output by looking up in the fuzzy control decision table by inputs \( \dot{e}(k) \) and \( c\dot{e}(k) \), the output is commonly not directly control the object system. We have to transform it to its object domain. The CPU utilization gain acquired after transforming from the output \( \dot{u}(k) \) by the table 10 in paper[6].

5.5 PID CONTROLLER

In our model (figure 1), we use the discrete digital PID formula (1) as the adaptive PID controller\(^{1}\).

\[
\Delta CPU(k) = K_p e(k) + K_i \sum_{i} e(i) + K_d (e(k) - e(k-1)) W
\]

<table>
<thead>
<tr>
<th>( \dot{u} )</th>
<th>( \dot{e} )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>-6</td>
<td>-6.53</td>
<td>-5.23</td>
<td>-5.23</td>
<td>-5.23</td>
<td>-5.23</td>
<td>-5.23</td>
<td>-5.23</td>
<td>-5.23</td>
</tr>
<tr>
<td>-5</td>
<td>-5.00</td>
<td>-4.95</td>
<td>-4.95</td>
<td>-4.95</td>
<td>-4.95</td>
<td>-4.95</td>
<td>-4.95</td>
<td>-4.95</td>
</tr>
<tr>
<td>-4</td>
<td>-4.69</td>
<td>-4.52</td>
<td>-4.52</td>
<td>-4.52</td>
<td>-4.52</td>
<td>-4.52</td>
<td>-4.52</td>
<td>-4.52</td>
</tr>
<tr>
<td>-2</td>
<td>-4.00</td>
<td>-4.00</td>
<td>-3.78</td>
<td>-3.78</td>
<td>-3.47</td>
<td>-3.47</td>
<td>-3.47</td>
<td>-3.47</td>
</tr>
<tr>
<td>-1</td>
<td>-3.59</td>
<td>-3.56</td>
<td>-2.93</td>
<td>-2.93</td>
<td>-2.60</td>
<td>-2.60</td>
<td>-2.60</td>
<td>-2.60</td>
</tr>
<tr>
<td>0</td>
<td>-2.92</td>
<td>-2.92</td>
<td>-2.33</td>
<td>-2.33</td>
<td>-1.91</td>
<td>-1.91</td>
<td>-1.91</td>
<td>-1.91</td>
</tr>
<tr>
<td>1</td>
<td>-1.81</td>
<td>-1.79</td>
<td>-0.57</td>
<td>-0.57</td>
<td>-0.31</td>
<td>-0.31</td>
<td>-0.31</td>
<td>-0.31</td>
</tr>
<tr>
<td>2</td>
<td>1.00</td>
<td>1.00</td>
<td>0.25</td>
<td>0.25</td>
<td>0.94</td>
<td>0.94</td>
<td>0.94</td>
<td>0.94</td>
</tr>
<tr>
<td>3</td>
<td>-0.58</td>
<td>-0.64</td>
<td>0.69</td>
<td>0.69</td>
<td>1.42</td>
<td>1.42</td>
<td>1.42</td>
<td>1.42</td>
</tr>
<tr>
<td>4</td>
<td>0.23</td>
<td>0.24</td>
<td>1.12</td>
<td>1.12</td>
<td>1.79</td>
<td>1.79</td>
<td>1.79</td>
<td>1.79</td>
</tr>
<tr>
<td>5</td>
<td>0.0</td>
<td>0.0</td>
<td>1.29</td>
<td>1.29</td>
<td>2.00</td>
<td>2.00</td>
<td>2.00</td>
<td>2.00</td>
</tr>
<tr>
<td>6</td>
<td>0.0</td>
<td>0.0</td>
<td>1.29</td>
<td>1.29</td>
<td>2.00</td>
<td>2.00</td>
<td>2.00</td>
<td>2.00</td>
</tr>
</tbody>
</table>

In the formula, \( K_p \) is proportion coefficient; \( K_i \) is the integral time coefficient; \( K_d \) is the differential time coefficient. \( W \) is 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 state using PID feedback control. On the contrary, when the workload changes little, it easily cause 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 the system performance.

When the deviation of the current performance value is greater than the set threshold value, fuzzy feedback is chosen, or PID feedback do 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_k; c\dot{e}(k) = (e(k) - e(k-1))/W

if \((abs(e(k))) < M_{threshold}\) then PID control function, \( abs(e(k)) \) is the absolute value of \( e(k) \)

\( \Delta CPU(k) = K_p e(k) + K_i \sum_{i} e(i) + K_d c\dot{e}(k); \)

\} else if fuzzy control \}

Get the \( e(k) \) and \( c\dot{e}(k) \) by fuzzier;

Get the output value \( \dot{u}(k) \) by looking up in the fuzzy decision control table using the inputs \( e(k) \) and \( c\dot{e}(k) \);

Get the \( \Delta CPU(k) \) by domain transformer using the input \( \dot{u}(k) \);

Call the 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 are mainly referred to paper [1][2][6]; the \(M_s\) and \(M_{threshold}\) are set to 0.15 and 0.05, respectively.

5.7 Service Level Controller and the 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 \(\Delta CPU(k)\) in the sampling instant \(KW\) by the fuzzy-PID feedback controller, the next step is to tune the requested utilization by service level controller and the admission controller. If \(\Delta CPU(k) > 0\), 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 \(\Delta CPU < 0\). The pseudo code function SLC in Paper [1] is the Service Level and admission controller.

6 EXPERIMENT AND CONCLUSION

6.1 Workload Model

To compare the result of the PID adaptive control with fuzzy scheduling model and our fuzzy-PID adaptive control with fuzzy scheduling model, our simulation test environment refers to that of paper [1][2][6]. 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 100th, 200th, and 300th sampling instantnts, \(etf\) changes to 1.6, 1.3 and 0.9, respectively.

6.2 EXPERIMENT RESULT AND CONCLUSION

| TABLE 3 THE DEADLINE MISS RATIO COMPARISON OF FUZZY SCHEDULER AND EDF SCHEDULER |
|---------------------|----|----|----|
|                     | high | normal | low |
| Fuzzy scheduler     | 0.0016 | 0.0041 | 0.376 |
| EDF scheduler       | 0.097 | 0.106  | 0.103 |

Figure 6 and Table 3 is our test experiment result. In figure 6, the red curve denotes the deadline miss ratio variation of our adaptive fuzzy-PID performance control and the blue curve denotes that of the adaptive PID performance control. In the initial state (\(etf = 0.5\)) the actual workload is smaller than the estimation in the 100th sampling instantnt, the actual workload is varyied greater than the estimation abruptly (\(etf\) changes from 0.5 to 1.6); then in the 200th and 300th sampling instatnts, the workload varies abruptly due to \(etf\) change. The simulation test experiment result of figure 6 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 3 shows that the more important a task is, the less deadline miss ratio a task has in the fuzzy scheduler; but in the EDF scheduler, the deadline miss ratios of all three kind tasks are very near. Of course, the fuzzy control need CPU overhead. The use of the fuzzy control decision table in our model can greatly reduce the time overhead.

REFERENCES