第31卷第11期 宇 航 学 报 Vol. 31 No. 11
2010年11月 Journal of Astronautics November 2010

硬实时系统中自适应反馈软件容错动态调度算法研究

陈源龙，马培军，李 东
（哈尔滨工业大学计算机科学与技术学院，哈尔滨 150001）

摘 要：在飞行控制等硬实时系统中由于任务超时完成将会给系统运行带来灾难性后果，而现有软件容错调度算法在处理机利用率较高时，成功执行主任务所占时间比率下降，针对此提出自适应反馈容错动态调度算法，此算法在经典软件容错调度算法 BCE（Basic CAT EIT）的基础上，引入反馈调度机制，形成 Feedback BCE 调度算法。该算法在运行过程中定期监测处理机利用率，将实际处理机利用率与预期值进行比较，根据比较结果调整对任务集的调度。实验表明，相对于其他同类算法，自适应反馈软件容错调度算法有效降低了浪费的 CPU 时间片数量，提高了成功执行主任务所占时间比率，有效降低了因处理机超载而引起的主任务丢失率。

关键词：硬实时系统；软件容错；自适应反馈调度
中图分类号：TP316 **文献标识码**：A **文章编号**：1000-1328(2010)11-2591-06
DOI：10.3873/j.issn.1000-1328.2010.11.024

Dynamic Scheduling Algorithm with Adaptive Feedback Software Fault-Tolerance in Hard Real-Time Systems

CHEN Yuan-long, MA Pei-jun, LI Dong
(School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China)

Abstract: In hard real-time systems, overtime of completion of a task can induce a disastrous consequence to entire system performances. Moreover, the proportions of completed primary tasks in current software fault-tolerant schedule algorithms decrease with the increase of CPU utilization ratio. To solve these problems, an adaptive feedback software fault-tolerance schedule algorithm based on the BCE (BASIC CAT EIT) method is presented by introducing a feedback schedule mechanism. The algorithm regularly monitors the CPU utilization ratios during processing, and compares them with estimated values. Task schedules are subsequently adjusted according to these comparative results. The experimental results show that the proposed approach increases the proportions of successful primary tasks, while it decreases the wasted CPU time slots and the primary task loss rate caused by CPU utilization overload.

Key words: Hard real-time system; Software fault-tolerance; Adaptive feedback schedule

0 引 言

在航空航天的飞行控制系统等工业领域中，若任务未能在规定时间内产生正确结果将造成灾难性后果，这样的计算机系统称为硬实时系统。

由于计算机软件、硬件在运行过程中都存在可靠性问题，所以对于工作在关键领域的实时系统，除了要满足实时性要求外，还需能够在软件、硬件发生故障时，保证任务能够正确按时完成，这样就要求实时系统具有容错能力。

针对此问题目前已提出多种容错模型，分布式容错模型依赖于多处理器间的任务分配，对任务进行冗余计算[1]。

单机硬实时系统中，容错调度算法的计算模型

收稿日期：2009-11-22； 修回日期：2010-10-23

有以下三种[2]：

（1）硬件容错模型。该模型的思想是通过“硬件冗余”（即重复设置若干个后备处理器）来实现容错。在这种模型中，一个计算任务被复制为基版本和副版本，当主处理器上的基版本运行失败时，运行于后备处理器上的副版本将给出计算结果。

该模型的实时性高，但需增加硬件开销。

（2）非精确计算模型。每个任务分为强制性子任务和选择性子任务，该模型调度算法在保证强制子任务完成的情况下尽可能运行选择子任务，提高运算精度。

（3）软件容错模型。该类模型将任务分为主任务和替代任务。当主任务在运行过程中发生错误时，系统将调度替代任务来完成计算。但由于主任务的运行已占用了一部分时间，所以替代任务必须通过降低计算复杂度的方法，来保证在剩余时间内完成计算，给出满足最低计算精度的结果。

由于软件容错模型不增加硬件，又能满足工业界要求，因此成为目前硬实时系统的主流容错模型。

在有关软件容错模型最近的研究中，Han 等人提出了 BCE（Basic CAT EIT）算法[3]，算法包括 Basic、CAT 及 EIT 三个子算法。此算法在处理机利用率较低的情况下利用 CAT（可利用时间计算算法）算法能够较好地预测出主任务是否可调度，随着处理机利用率的升高，CAT 算法对于主任务是否可调度的预测精度降低，主任务丢失率偏高。

为改进对主任务是否可调度的预测精度，韩建军等提出 PKSA 和 CUBA 算法[2,4]。这两种算法均类似最早截止期优先[5]（Earlist Deadline First，EDF）算法。PKSA 在 BCE 基础上，通过 K 次试探性检测改进了对主任务可执行性的预测。但由于这种预测只是在某一局部的时间段内进行有限次预测，并且当主任务顺利完成并撤销相应的替代部分时，试探策略并不调整受影响的替代部分执行时间，因此当任务增多时，主任务可执行性的预测精度也随之降低。

刘东等提出了 EDBA 算法[6]，该算法通过构建预测表提高对主任务可否调度的预测精度，此算法以 EDF 为基础，实验结果较其他算法有一定提高。

经研究发现目前单机容错算法思想不少都遵循 EDF 算法思想优先调度周期短[1-5]、截止期早的任务。这类算法当处理机利用率超负荷运行时主任务截止期完成率大幅下降。

本文将反馈调度技术应用于软件容错调度，提出 FC-BCE（Feedback BCE）算法。利用反馈调度技术对任务集的处理机利用率进行监测，可以大幅提高对主任务调度的预测精度，花费较小的代价来达到整个任务集的最优调度。仿真实验结果表明 FC-BCE 算法较目前已有的算法减少了浪费 CPU 时间片的数量，同时增加了被成功调度的主任务数量。

1 FC-BCE 算法计算模型

硬实时系统中包含一个周期性任务集合 $S=\{t_1,t_2,\cdots,t_n\}$，刻画任务的数值为：每个任务 T_i 的周期 PE_i，软件错误概率为 FP_i，任务 T_i 包含主任务 P_i 及替代任务 A_i，P_i 的执行时间为 c_i，A_i 的执行时间为 a_i，且 $a_i < c_i$，任务 T_i 的每个实例 T_{ij} 包含两个部分：主任务 P_{ij} 和替代任务 A_{ij}，T_{ij} 的到达时间 r_{ij} 为 $(j-1)\cdot \mathrm{PE}_i$，截止期 d_{ij} 为 $j\cdot \mathrm{PE}_i$，每个 A_{ij} 有一个通知时间（NT_{ij}）表示 A_{ij} 的开始时间，$\mathrm{ED}_{i,j}$ 为 A_{ij} 的结束时间，在调度过程中若处理器空闲时间充裕则调度主任务，否则调度替代任务。计划周期（Planning Cycle，PC）为所有 N 个任务周期的最小公倍数。在一个计划周期内 T_i 执行的次数为 N_i，处理机利用率 U_p 定义为 $\sum_{i=1}^{N}\frac{c_i}{\mathrm{PE}_i}$。

计算模型中任务具有可抢占性，且仅考虑单机系统，假定所有的替代部分均可调度。本计算模型仅考虑一个计划周期内的任务调度，在确保替代任务能够完成的情况下，尽可能地调度主任务，提高系统计算的精度[4]。为了下面叙述方便，先给出如下两个定义[6]：

定义 1. 成功执行主任务所占时间比率 PSP（Percent for Successful Primaries）：

PSP =（执行成功的主任务所用时间/模拟时间）×100%

此参数表示主任务被调度的情况，值越大说明调度的主任务越多，算法的性能越高。

定义 2. 浪费的 CPU 时间 W：若主任务因执行时间不够而被撤销，此时主任务已经执行的 CPU 时间称为浪费的 CPU 时间。

此参数决定了对主任务可执行性的预测精度，值越大说明对主任务的可执行预测精度越差。

2 BCE 容错调度算法

BCE 算法含基本调度算法（Basic），可用时间检测算法（CAT），以及空闲时间消除算法（EIT）[3]。

其中,基本调度算法依据任务的截止时间划分任务的优先级,截止期最早的拥有最高的优先级。当周期任务的优先级确定后,即周期任务的调度顺序确定,尽可能推迟替代任务的调度。替代任务的执行具有可抢占性,即替代任务的执行时间可能不是连续的。当到达替代任务的通知时间时替代任务具有最高优先级,若此时有主任务在运行则放弃主任务的运行,对替代任务进行调度。若在替代任务的通知时间到达以前,相应的主任务已运算完成,则放弃对替代任务的调度,并释放替代任务的处理器占用时间,调整受影响的其他替代任务的调度,并计算新的通知时间。

设当前时间为 t,准备执行的主任务为 P_{ij},t 到 NT_{ij} 之间分配给替代部分的时间间隔时间总和为 $I=\sum_{m}\sum_{n}\sum_{p}(ED_p(m,n)-ST_p(m,n))$,$t\leqslant ST_p(m,n)\leqslant ED_p(m,n)\leqslant NT_{ij}$,则 P_{ij} 可用时间 $AT_{ij}=NT_{ij}-t-I$。其中 m 为任务编号,j 为实例编号,p 为第 p 个时间间隔,$ST_p(m,n)$ 为 $A_{m,n}$ 的开始执行时间。

若某时刻运行的是主任务,则可用时间检测算法检测从该时刻至该实例的通知时间之间的可用时间是否足以完成主任务剩余执行,若足够则继续执行,若不够则放弃。目前已知的大多算法都是在此算法的基础上进行改进,改进的方向主要是提高主任务按时完成的预测精度。

以上分析可以看出,软件容错动态调度算法大多先按照 RM 反向调度算法事先分配替代部分的时间间隔,然后采用以替代部分通知时间做为截止期的 EDF 正向调度算法调度主任务执行。

3　FC-BCE 算法

3.1　现有算法存在的问题

目前容错调度算法通过对主任务的可调度性进行预测,随着预测精度的提高,能够降低主任务丢失率。但是通过研究分析发现造成任务丢失的主要原因在于任务集处理机利用率过高时,若还继续调度主任务,那么必会出现任务丢失,尤其是处理机利用率大于 1 时,任务集的任务丢失率急剧升高。这是因为,当任务集处理机利用率大于 1 时,说明若任务集要全部调度完成,所需的时间将大于 PC,即当任务集的处理机利用率大于 1 时,在 PC 内不可能将任务集内的全部任务调度完成。这也是造成目前以 EDF 算法为基础提出的软件容错调度算法当处理机利用率大于 1 时任务丢失率偏高的根本原因。

若在调度过程中,引入反馈调度机制,保证处理机利用率维持在 1,那么就可以降低任务丢失率,提高主任务的成功执行率,基于这样一种考虑,利用反馈控制器的反馈结果对将要被调度的任务集进行调整,从而将任务集的处理机利用率维持在 1 附近,以达到降低任务丢失率,减少浪费 CPU 时间片数量的目的。

3.2　FC-BCE 算法原理

FC-BCE 算法的基本思想是:定时对处理机利用率进行检测,若发现处理机利用率高于预估值,计算出需要改变的任务执行时间量,通过放弃一部分主任务,直接调度替代任务来达到改变任务集占用处理机时间的目的,当任务集处理机利用率回到预估值时,对任务集按照 BCE 算法进行调度。

反馈控制调度算法可以保证任务集处理机利用率在 1 附近变化,通过本文下面提出的定理 1 可知只要保证任务集处理机利用率在 1 以内就可以保证任务集是可调度的,同时达到提高成功执行主任务所占时间比率的目的。

与任务调度算法证明过程[8]不同,以下证明过程中 $p_1,\cdots,p_n$ 代表任务被调度时所运行的时间,而这个时间受处理机利用率的影响是动态变化的。任务的调度过程中,由于时限限制,有的任务在某一时间内主任务被调度,其他时间内替代任务被调度,因此在整个调度过程中任务每个周期内平均调度时间为 p_i。α,β 分别为受替代任务、主任务被调度次数影响的权值。

定理 1.　在软件容错调度算法中,硬实时周期性任务集在[0,PC]内调度,周期性任务集可调度的充要条件为 $\sum_{i=1}^{n}\frac{p_i}{PE_i}<1$ ($p_i=\alpha a_i+\beta c_i$, $0\leqslant\alpha\leqslant 1$,$0\leqslant\beta\leqslant 1$)。

证.　必要性证明:

设在调度期[0,PC]内所有任务在调度完成后所需的计算时间可以通过下式计算:

$$p_1\cdot(PE_2\cdots PE_n)+p_2\cdot(PE_1\cdots PE_n)+\cdots+p_n\cdot(PE_1\cdot PE_2\cdots PE_{n-1})$$

$$p_i=\alpha a_i+\beta c_i,\ 0\leqslant\alpha\leqslant 1,0\leqslant\beta\leqslant 1$$

如果所有任务所需的计算时间超过可以得到的处理机时间之和:

$$p_1\cdot(PE_2\cdots PE_n)+p_2\cdot(PE_1\cdot PE_3\cdots PE_n)+\cdots+p_n\cdot(PE_1\cdot PE_2\cdots PE_{n-1})>$$

$$PE_1 \cdot PE_2 \cdots PE_n$$

即 $\sum_{i=1}^{n} \frac{p_i}{PE_i} > 1$。

充分性证明：

当 $\sum_{i=1}^{n} \frac{p_i}{PE_i} < 1$ 时，假设任务集是不可调度的，设在时刻 π 任务集发生溢出，则当时间 t，在0到 π 时处理器没有空闲时间。设任务 $a_1, a_2, \cdots, a_m$ 完成时间在 π 以前。$b_1, b_2, \cdots$ 的完成时间在 π 以后。分两种情况进行考虑：

情形1. 任务 $b_1, b_2, \cdots$ 在 π 之前没有被调度，在0到 π 之间任务 $a_1, a_2, \cdots, a_m$ 所需要的计算时间之和为 $\lfloor \frac{\pi}{T_1} \rfloor p_1 + \lfloor \frac{\pi}{T_2} \rfloor p_2 + \cdots + \lfloor \frac{\pi}{T_m} \rfloor p_m$，因为在0到 π 之间处理器没有空余时间，所以

$$\lfloor \frac{\pi}{T_1} \rfloor p_1 + \lfloor \frac{\pi}{T_2} \rfloor p_2 + \cdots + \lfloor \frac{\pi}{T_m} \rfloor p_m > \pi$$

同样，$x \geqslant \lfloor x \rfloor$

$$\lfloor \frac{\pi}{T_1} \rfloor p_1 + \lfloor \frac{\pi}{T_2} \rfloor p_2 + \cdots + \lfloor \frac{\pi}{T_m} \rfloor p_m > \pi$$

则 $\frac{p_1}{T_1} + \frac{p_2}{T_2} + \cdots + \frac{p_m}{T_m} > 1$

与题设矛盾。

情形2. 任务 $b_1, b_2, \cdots$ 在 π 之前有部分被调度，同时溢出发生在时刻 π，则必然存在 π'，当 $\pi' < t < \pi$，任务 $b_1, b_2, \cdots$ 没有被调度的，也就是说在这段时间内，被调度的任务的截止期都小于 π。当然截止期小于 π 的任务也会在 π' 之前执行完成，因此在 $\pi' < t < \pi$ 内，任务的计算时间小于或等于

$$\lfloor \frac{\pi - \pi'}{T_1} \rfloor p_1 + \lfloor \frac{\pi - \pi'}{T_2} \rfloor p_2 + \cdots + \lfloor \frac{\pi - \pi'}{T_m} \rfloor p_m$$

由于在时刻 π 发生溢出，所以

$$\lfloor \frac{\pi - \pi'}{T_1} \rfloor p_1 + \lfloor \frac{\pi - \pi'}{T_2} \rfloor p_2 + \cdots + \lfloor \frac{\pi - \pi'}{T_m} \rfloor P_m > \pi - \pi'$$

即 $\frac{p_1}{T_1} + \frac{p_2}{T_2} + \cdots + \frac{p_m}{T_m} > 1$

因此 $\sum_{i=1}^{n} \frac{p_i}{PE_i} < 1$ 时任务集是可调度的。

定理1说明在采用主任务，替代任务的软件容错调度算法中，若想提高任务集的可调度性，降低截止期错失率，可以通过改变任务集占用处理机运行时间，使整个被调度的任务集处理机利用率维持在1左右。在调度过程开始前主动放弃一些主任务的调度，直接调度替代任务，来改变任务集占用处理机时间，使处理机利用率维持在一个合理的范围内，从而降低任务截止期错失率，达到优化调度的目的。

3.3 FC-BCE 算法实现

图1给出了反馈软件容错动态调度算法的体系框架。该框架由基本软件容错动态调度器(BCE Scheduler)和反馈控制组成。反馈控制由监视器和准入控制器构成。

根据定理1只要任务集处理机利用率小于等于1，则任务集就是可调度的，并且任务集的截止期错失率也会维持在一个较低的水平，更多的主任务将会得到成功的调度。

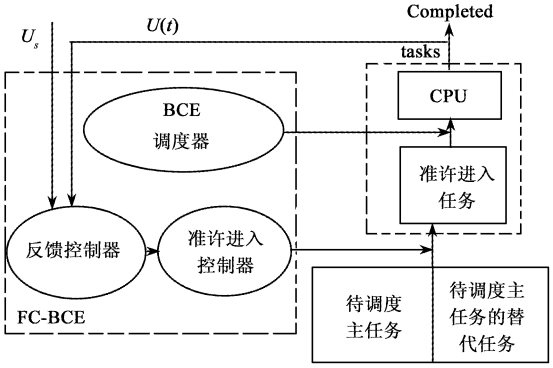

图1 反馈软件容错动态调度算法体系框架

Fig. 1 Architecture of algorithms with adaptive feedback software fault-tolerance

将BCE算法作为底层调度器，根据各实时任务截止期限对任务进行调度，为上层调度提供反馈信息。在调度间隔时调用反馈控制器，以期望的处理机利用率为输入，以需要改变调度任务占用时间量为输出。如果系统中实际处理机利用率高于期望的处理机利用率，根据需要改变进入就绪队列的任务，此时直接调用某些任务的替代任务，而不对主任务进行调度或试探调度。具体用哪个任务的替代任务代替主任务，则选择主任务与替代任务时间差最大的任务。

4 FC-BCE 算法反馈控制器

4.1 PID 控制器

反馈控制器是本算法的核心部分，它将处理机利用率限制在一定的范围内，从根本上有效地降低

了实时任务截至期错失率。

基本的反馈控制器是 PID 控制器。PID 控制器周期性监测受控变量 $U(t)$,并通过以下公式计算出任务占用处理机时间的改变量。PID 控制器公式如下[8]:

$$\Delta T = C_P \mathrm{Error}(t) + C_I \int \mathrm{Error}(t)\,\mathrm{d}t + C_D \frac{\mathrm{dError}(t)}{\mathrm{d}t} \tag{1}$$

PID 控制器的主要参数:

受控变量 Controlled Variable:实际处理机利用率 $U(t)$;

系统期望值 Set Point:系统期望的处理机利用率 U_s;

误差 Error:受控变量与期望值之间的差值 $e(t) = U_s - U(t)$;

控制变量 Manipulated Variable:由 PID 控制器算出的估计值,用以间接调整受控变量;

C_p, C_I, C_D: PID 控制器的计算参数。

间隔一段时间通过公式 1,利用期望值与受控变量之间的差计算出需改变的任务集时间量。处理机处理任务时,具有时变非线性等特点,受随机干扰影响,常规 PID 控制器的参数不易在线自调整,使其应用受到限制,因此采用单神经元 PSD 控制器。

4.2　单神经元 PSD 控制器

单神经元 PSD 控制器输出可表示为[8]:

$$\Delta T = K(w_1'(k)x_1(k) + w_2'(k)x_2(k) + w_3'(k)x_3(k)) \tag{2}$$

其中 $x_i(k)(i = 1,2,3)$ 为神经元输入信号,它们分别为

$$\begin{cases} x_1(k) = e(k) \\ x_2(k) = \Delta e(k) \\ x_3(k) = (\Delta e(k))^2 \end{cases} \tag{3}$$

$$\begin{cases} e(k) = U_s - U(k) \\ \Delta e(k) = e(k) - e(k-1) \\ (\Delta e(k))^2 = e(k) - 2e(k-1) + e(k+2) \end{cases} \tag{4}$$

$w_i(k)(i = 1,2,3)$ 为对应于 $x_i(k)$ 的加权系数。

其中 $w_i'(k) = w_i(k) / \sum_{i=1}^{3} | w_i(k) |$

$$\begin{cases} w_1(k+1) = w_1(k) + d_I z(k) u(k)[e(k) + \Delta e(k)] \\ w_2(k+2) = w_2(k) + d_p z(k) u(k)[e(k) + \Delta e(k)] \\ w_3(k+3) = w_3(k) + d_D z(k) u(k)[e(k) + \Delta e(k)] \end{cases} \tag{5}$$

$z(k) = e(k)$;

$d_I = 0.4$;

$d_p = 0.35$;

$d_D = 0.4$;

$w(1) = C_P = 0.05$;

$w(2) = C_I = 0.05$;

$w(3) = C_D = 0.05$;

$K = 0.5$。

5　模拟测试及结果分析

5.1　评价标准

本文将在不同 CPU 利用率的情况下对 BCE 和 FC-BCE 算法性能进行观察比较。通过调整主任务执行时间相对于任务周期所占的比例来调整 CPU 利用率。处理机利用率 U_p 在 0.8 ~ 1.6。任务集选择(15,p_1,2),(30, p_2,4),(50, p_3,11),(120, p_4, 15)。其中任务集用三元组(PE_i, $p_{i,j}$, $A_{i,j}$) 表示运行时间取 3000 个时间单位。实验中改变主任务占用时间可以改变任务集的处理机利用率。

本文采用成功执行主任务所占时间比率和浪费 CPU 时间片数量来评价算法的性能。

5.2　不同 U_p 下 FC-BCE 与 BCE 的比较

设软件出错概率为 0.1,算法的执行结果如图 2、图 3 所示。由图中可看出当处理机利用率小于 1 时,BCE 算法和 FC-BCE 算法的各项指标一致,当处理机利用率大于 1 时 BCE 算法的主任务完成率开始降低,降幅超过 FC-BCE 算法。这是由于当处理机利用率小于 1 时,反馈算法不运行。此时两个算法的调度结果一致。随着处理机利用率的增加,反馈调度算法在监测到处理机利用率大于 1 时,调度算法计算出需要改变的任务占用处理机时间量,对任务集中主任务部分与替代任务部分运行时间差值最大的主任务主动放弃执行,直接调度替代任务。当处理机利用率降到 1 时,开始正常调度。

由图 2,3 中可进一步看出当处理机利用率超过 1 时,BCE 算法性能降低很快,而 FC - BCE 算法性能虽也有一定降低,但过程比较平缓,稳定性更好。

经过分析发现 BCE 算法只对某一局部时间内可能参与运行的任务子集进行预测,周期越长的任务预测精度越低。而 FC-BCE 算法依据对处理机利用率的定期监测结果对整个任务集进行容错调度,有选择地主动放弃一部分主任务的执行,达到降低

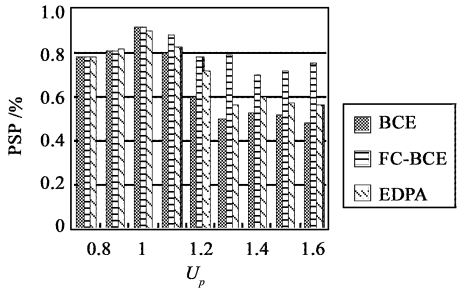

图2 不同 U_p 下算法的 PSP 比较

Fig. 2 Comparison of PSP under different U_p

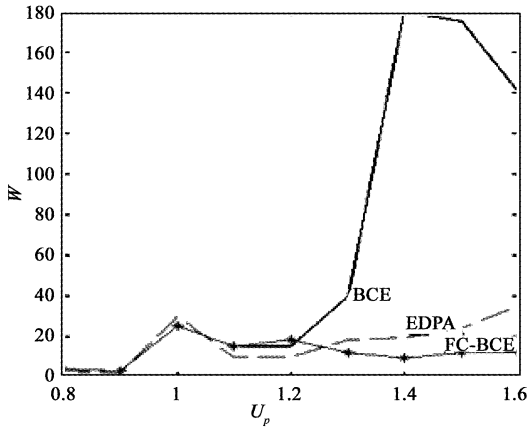

图3 不同 U_p 下算法 W 比较

Fig. 3 Comparison of W under different U_p

处理机利用率的目的,虽然也会有一部分主任务未被调度,但是从实验结果可以看出 FC-BCE 算法调度了更多的主任务,同时浪费 CPU 时间片基本为0,说明 FC-BCE 算法具有更好的调度性能,更高的预测精度。

5.3 FC-BCE 算法与其他软件容错算法的比较

本文还模拟了文献[6]的 EDPA 算法从图2,图3中可看出处理机利用率超过1.2时,EDPA 算法较 BCE 算法主任务完成率提高3% ~8%,而 FC-BCE 算法较 BCE 主任务完成率提高10% ~13%,同时处理机浪费时间片数量大幅降低。和 PKSA、CUBA 算法相比,FC-BCE 对 BCE 算法性能提升得更高。

6 结 论

本文针对硬实时系统中的软件容错模型,在 BCE 算法的基础上引入自适应反馈调度技术,提出 FC-BCE 算法。较之其它软件容错调度算法,FC-BCE 算法从一个宏观的角度对任务集的可调度性进行评估,而不是局限在某一时间段内对任务是否可调度进行预估。实验表明,依据这一思路导出的 FC-BCE 算法具有更好的调度性能,更高的预测精度。

参 考 文 献

[1] Anne B, Mourad H, Yves R. Contention awareness and fault-tolerant scheduling for precedence constrained tasks in heterogeneous systems[J]. Parallel Computing, 2009, 35: 83 - 108.

[2] 李庆华,韩建军. 硬实时系统中基于软件容错的动态调度算法[J]. 软件学报, 2005, 16(1): 101 - 107. [Li Qing-hua, Han Jian-jun. Dynamic scheduling algorithms with software fault-tolerance in hard real-time systems[J]. Journal of Software, 2005, 16(1): 101 - 107.]

[3] Ching C H, Kang G S, Jian W. A fault-tolerant scheduling algorithm for real time periodic tasks with possible software faults[J]. IEEE Trans. on Computers, 2003, 52(3): 362 - 372.

[4] 韩建军,李庆华. 基于软件容错的动态实时调度算法[J]. 计算机研究与发展, 2005, 42(2): 315 - 321. [Han Jian-jun, Li Qing-hua. A dynamic real-time scheduling algorithm with software fault-tolerance[J]. Journal of Computer Research and Development, 2005, 42(2): 315 - 321.]

[5] Liu C L, James W L. Scheduling algorithms for multiprogramming in a hard real-time environment[J]. Journal of the Association for Computer Machinery, 1973, 20(1): 46 - 61.

[6] 刘东,张春元,李瑞. 软件容错模型中的容错实时调度算法[J]. 计算机研究与发展, 2007, 44(9): 1495 - 1500. [Liu Dong, Zhang Chun-yuan, Li Rui. Fault-tolerant real-time scheduling algorithm in software fault-tolerant module[J]. Journal of Computer Research and Development, 2007, 44(9): 1495 - 1500.]

[7] Chen Y L, John A. Design and evaluation of a feedback control earliest deadline first scheduling[C]. The 20th IEEE Real-Time system Symposium, Phonenix, USA, December 1 - 3, 1999.

[8] 柴华. 一种反馈控制机制在 EDF 算法上的应用[D]. 太原:太原理工大学, 2008. [Cai Hua. A feedback control scheme applied to earliest deadline first scheduling algorithm[D]. Taiyuan: Tai Yuan University of Technology, 2008.]

作者简介:陈源龙(1981 -),男,哈尔滨工业大学空间计算技术研究中心博士研究生,主要从事分布式实时仿真系统实时性与容错技术方向研究。

通信地址:哈尔滨市南岗区西大直街92号哈尔滨工业大学319信箱(150001)

电话:18686856429

E-mail:cyuanlong@126.com

(编辑:余 未)