硬实时系统在强分区约束下的双层分区调度

李昕颖 顾 健 何 锋 熊华钢

(北京航空航天大学电子信息工程学院 北京 100191)

摘 要 文中研究了硬实时系统在强分区约束下的双层分区的调度问题,合理建立了强分区约束下的双层分区调度模型,给出了最坏情况下的分区任务集可调度的判定条件.同时,在此基础上,提出了与分区利用率匹配的分区设计方法,导出了该方法下的系统可调度利用率的最小上限.仿真实验表明,在严格实时的条件下,文中提出的方法相对于现有方法更具优越性,并提高了分区可调度利用率的最小上限.

关键词 实时系统;强分区约束;分区调度;可调度利用率;分区设计

中图法分类号 TP393 **DOI 号**: 10.3724/SP.J.1016.2010.01032

Two-Level Partition Scheduling in Hard Real Time System Under Strong Partition Constraints

LI Xin-Ying GU Jian HE Feng XIONG Hua-Gang

(*School of Electronic and Information Engineering, Beihang University, Beijing* 100191)

Abstract This paper studies two-level partition scheduling problem of strongly partitioned hard real time systems and proposes a novel two-level partition scheduling model under strong partition constraints. In addition, it presents a schedulability condition under worst case which is based on the notion of partition availability for a known partition, addresses the inverse problem of designing a partition with minimum system level resource requirements to fulfill the application deadline constraints. A new partition design approach is proposed based on partition task set utilization matching. The least upper bounds of system utilization for the scheme have also been derived and formally proved. The experiments results show that under strictly real time condition the matching approach performs significantly better than existing partition design methods and increased least upper bound of partition schedulable utilization.

Keywords real-time systems; strong partition constraints; partition scheduling; schedulable utilization; partition design

1 引 言

在综合化实时系统中,各种应用软件共享计算资源、网络资源,为了防止错误在关键级别不同的应用软件之间传播,通过一定的分组策略(比如根据安全级别)将不同的应用软件封装在单独的分区之中,分区之间相互隔离.这种分组策略逐渐发展成分区管理技术,应用软件按照关键级别分解成不同的构件,使系统的升级和维护更加容易,并且大大加强了

收稿日期:2009-04-09;最终修改稿收到日期:2010-05-11.本课题得到国家自然科学基金(60879024)资助.**李昕颖**,女,1984 年生,博士研究生,主要研究方向为实时系统及综合化航空电子系统. E-mail: xinying_li@yahoo.cn. **顾 健**,男,1976 年生,博士研究生,主要研究方向为航空电子系统架构. **何 锋**,男,1980 年生,博士后,主要研究方向为航空电子网络通信及实时调度. **熊华钢**,男,1961 年生,博士,教授,博士生导师,主要研究领域为嵌入式系统和数字通信网络.

系统的容错能力.综合模块化的航空电子系统软件标准——ARINC653 标准中提出了双层分区的调度策略[1],由此带来两个问题:(1)给定分区参数,如何在严格实时的约束条件下,对分区任务的可调度性进行分析;(2)给定分区任务,分区的设计直接影响系统的实时特性.如何合理地设计分区,使分区中的任务都满足实时性要求,又提供尽可能大的处理器利用率.这是两个重要的理论和工程问题.

商用领域中的多数研究工作侧重于最小化任务的平均响应时间、最小化任务的截止期错过率等平均统计特性方面的研究[2].但是在航空电子等硬实时系统中,强调每个任务必须在截止时限内完成,满足实时性,否则可能带来灾难性后果.硬实时系统与一般的系统相比,具有不同的特性,因此研究方法有较大不同.文献[3-4]讨论了基于服务器思想的分层 EDF 调度方法,但是对于具有大量周期任务的航空电子等硬实时系统来说,是不合适的[5].针对区间层采用 RM 的分层分区调度,主要有两种方法来分析系统的可调度性.一种是采用 CPU 利用率的分析方法,还有一种是利用时间需求函数的分析方法.文献[6-8]都采用了时间需求函数的方法提出任务可调度性条件,但是这些文献中的分层分区调度模型都不满足强分区约束条件,也就是分区隔离且固定周期的约束条件,并且推导得到的可调度判定条件时间复杂度很高,缺少鲁棒性.文献[9]利用了 CPU 利用率的分析方法,给出了区间层采用 RM 调度,分区所有任务可调度利用率的最小上限,但要求分区的周期必须是分区中所有任务周期的最大公约数或最大公约数的约数,约束条件使其实际应用受到很大限制.文献[10]运用以上两种方法对可调度性做了研究,但是得到的可调度利用率比较低.对于分区设计的研究,文献[7,11]提出的方法以时间需求函数推出的可调度性条件为基础,时间复杂度高.文献[12]提出的方法有效地避免了时间需求函数的复杂性,采用均衡型和耗尽型的方法来进行分区设计,但不能完全保证系统的实时性.

本文根据以上不足,建立了满足强分区约束条件的双层分区调度模型.在给定分区参数条件下,推导了最坏情况下的分区任务集严格实时条件,并提出了与分区利用率匹配的分区设计方法,导出了该方法下系统可调度利用率上限.通过仿真进行了比较和验证.

2　基本概念

2.1　分层调度模型

双层分区调度模型如图 1 所示,操作系统层根据 ARINC 653 标准,采用轮转调度的方式激活每一个分区.在区间层,根据分区内定义的调度策略进行任务调度.每一个分区内部的任务只能在当前分区处于激活状态才有可能被执行,从而使得模块中各分区相互独立[13].本文区间层采用 RM(Rate Monotonic)调度算法,它的定时行为比动态优先级算法调度的系统具有更好的预知性,更满足硬实时系统可预测性的要求.

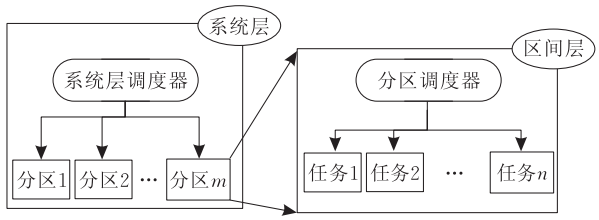

图 1　双层分区调度模型

2.2　任务模型

假设有 n 个实时任务,分别用 $\tau_1,\tau_2,\cdots,\tau_n$ 表示,它们组成一个任务集合 Γ,即 $\Gamma=\{\tau_1,\tau_2,\cdots,\tau_n\}$.任务特征定义如下:

(1)任务最坏情形执行时间 C_i:表示第 i 个任务在最坏情况下无中断执行所需的处理器时间.

(2)任务的周期 T_i:表示第 i 个任务的产生周期.对于非周期任务,则表示任务产生的最小时间间隔.

(3)任务的截止期限等于任务的周期,即 $D_i=T_i$.

(4)任务之间相互独立(不存在因共享资源而导致阻塞).根据以上定义,任务由一个二维数组表述:

$$\tau_i=\{C_i,T_i\} \tag{1}$$

任务 τ_i 的利用率定义为

$$U_i=C_i/T_i \tag{2}$$

任务集 $\tau_1,\tau_2,\cdots,\tau_n$ 的利用率为

$$U=\sum_{1\le i\le n}U_i \tag{3}$$

2.3　分区调度模型

假定分区 P_Ω 由 N_Ω 个任务组成,可表示为 $\Gamma_\Omega=\{\tau_{\Omega_i}(C_i,T_i),\ i=1,2,\cdots,N_\Omega\}$.其中 τ_{Ω_i} 的下标表示

第 Ω 个分区中的第 i 个任务. 分区调度模型如图 2 所示. 在系统层,采用周期性的轮转调度激活分区 P_Ω,我们定义系统的轮转周期时长为 T_{RL}. 在每一个轮转周期内,T_Ω 表示分区 P_Ω 的执行时间,在其余 $T_{RL}-T_\Omega$ 时间段内,分区 P_Ω 处于阻塞状态.

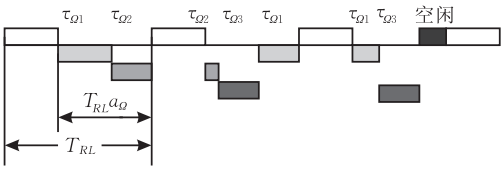

图 2 分区任务调度模型

2.4 相关定义

定义 1. 在一个调度中,若某任务集中的所有任务的实时约束条件都满足,则称在此调度策略下,该任务集是可调度的.

定义 2. 在某调度策略下,只要系统利用率不大于 U^*,就能保证系统中所有任务都是可调度的,U^* 称为可调度利用率的上限. U_{bound} 表示最坏情况下可调度利用率的上限,即为可调度利用率的最小上限.

定义 3. 分区内的任务只能在激活的分区窗口内运行,分区周期性地被激活,当某个处于激活状态的分区中没有任务处于就绪状态,即使其它分区有任务处于就绪状态,也不能跨越分区的边界在当前空闲的分区内执行,在时间和空间上对任务进行这种分区隔离,称为系统的强分区约束条件.

定义 4. α_Ω 表示分区 P_Ω 的执行能力,即每一次轮转周期内分区 P_Ω 的执行时间与轮转周期时长之比,$\alpha_\Omega=T_\Omega/T_{RL}$. 根据 α_Ω 的定义可知

$$0<\alpha_\Omega\leq 1 \tag{4}$$

若系统有 m 个分区,则系统可调度约束条件为

$$\sum_{\Omega=1}^{m}\alpha_\Omega\leq 1 \tag{5}$$

3 分区的可调度利用率分析

3.1 两个分区的简单情况

为了阐述思想,我们首先以两个分区的简单情况为例. 考虑分区 P_1 和 P_2,执行能力分别为 α_1、α_2,分区所包括的硬实时任务分别为 τ_1、τ_2. 假定 C_1、T_1、T_2 都已经确定了,现在考虑 τ_1、τ_2 满足截止时限的要求时,C_2 能取的最大值. 其中分区 P_1 对分区 P_2 的影响,可以看成是由一周期任务 τ_0 对 P_2 的抢占而形成. 任务 τ_0 的周期为 T_{RL},执行时间为 $T_{RL}(1-\alpha_2)$. 设 $R=T_2/T_{RL}\geq 1$,分两种情况讨论:

(1) $kT_{RL}\leq T_2\leq kT_{RL}+T_{RL}(1-\alpha_2)$,此时 C_2 最大为 $k(T_{RL}-T_{RL}(1-\alpha_2))=k\alpha_2T_{RL}$,分区 P_2 的任务利用率 $U_2=C_2/T_2=\dfrac{k(T_{RL}-T_{RL}(1-\alpha_2))}{RT_{RL}}=\dfrac{k\alpha_2}{R}$,$k\leq R\leq k+(1-\alpha_2)$. U_2 是关于 R 的单调递减函数.

(2) $kT_{RL}+T_{RL}(1-\alpha_2)\leq T_2\leq(k+1)T_{RL}$,此时 C_2 最大为 $T_2-(k+1)T_{RL}(1-\alpha_2)$,分区 P_2 的任务利用率 $U_2=C_2/T_2=1-\dfrac{(k+1)T_{RL}(1-\alpha_2)}{RT_{RL}}=1-\dfrac{(k+1)(1-\alpha_2)}{R}$,$k+(1-\alpha_2)\leq R\leq k+1$. U_2 是关于 R 的单调递增函数.

显然,在两种情况的交界点上 U_2 取最小值,即 $R=k+(1-\alpha_2)$ 时,$U_2=\dfrac{k\alpha_2}{k+(1-\alpha_2)}$,对于 $k=1,2,\cdots$,U_2 的最小值为 $\dfrac{\alpha_2}{2-\alpha_2}$.

同理,分区 P_1 的任务利用率 U_1 的最小值为 $\dfrac{\alpha_1}{2-\alpha_1}$.

3.2 两个分区的复杂情况

分区 P_1 和 P_2 分别包括硬实时任务集 $\Gamma_1=\{\tau_1',\tau_2',\cdots,\tau_k'\}$ 和 $\Gamma_2=\{\tau_1,\tau_2,\cdots,\tau_n\}$,执行能力分别为 α_1、α_2,其中分区 P_1 对分区 P_2 的影响,可以看成是由一周期任务 $\tau_0=\{T_{RL}(1-\alpha_2),T_{RL}\}$ 对 P_2 的抢占而形成的.

不失一般性,假定任务按 $T_1<T_2<T_3<\cdots<T_n$ 排列,则 $T_0<T_1<T_2<T_3<\cdots<T_n$,Liu 和 Layland[14] 证明了最难调度的系统发生在所有任务同时开始,并且 C_i,T_i 满足式(6)和式(7):

$$T_0<T_i<2T_0,\ \forall i=1,2,\cdots,n \tag{6}$$

$$C_i=T_{i+1}-T_i,\ 0\leq i\leq n-1,\ C_n=2T_0-T_n \tag{7}$$

此时,分区 P_2 的任务利用率

$$\begin{aligned}U&=\frac{T_2-T_1}{T_1}+\cdots\frac{T_n-T_{n-1}}{T_{n-1}}+\frac{2T_0-T_n}{T_n}\\&=\sum_{i=1}^{n-1}\frac{T_{i+1}-T_i}{T_i}+\frac{2T_0-T_n}{T_n}\end{aligned} \tag{8}$$

定义 $R_i=T_{i+1}/T_i$,式(8)可化为

$$\begin{aligned}U&=\sum_{i=1}^{n-1}R_i+2\frac{T_1}{(1+U_0)T_n}-n\\&=\sum_{i=1}^{n-1}R_i+\frac{2}{(1+U_0)\prod_{i=1}^{n-1}R_i}-n\\&\geq(n-1)\left(\prod_{i=1}^{n-1}R_i\right)^{\frac{1}{n-1}}+\frac{2}{(1+U_0)\prod_{i=1}^{n-1}R_i}-n\end{aligned} \tag{9}$$

当 $R_1=R_2\cdots=R_{n-1}=R$ 时，式(9)等号成立取最小值

$$U=(n-1)R+\frac{2}{(1+U_0)R^{n-1}}-n,$$

$$\frac{\partial U}{\partial R}=(n-1)-(n-1)\frac{2}{(1+U_0)R^n}=0,$$

即 $R=\left(\frac{2}{1+U_0}\right)^{\frac{1}{n}}$ 时，最小值为 $U=n\left(\left(\frac{2}{1+U_0}\right)^{\frac{1}{n}}-1\right)$，$U_0=(1-\alpha_2)$，分区 P_2 的任务总利用率取得最小值，即可调度利用率的最小上界

$$U_{\min}=U_{\text{bound}}=n\left(\left(\frac{2}{2-\alpha_2}\right)^{\frac{1}{n}}-1\right)\qquad(10)$$

同理分区 P_1 的任务可调度利用率的最小上界为 $U_{\text{bound}}=k\left(\left(\frac{2}{2-\alpha_1}\right)^{\frac{1}{k}}-1\right)$.

3.3　一般情况

定理 1.　处理器系统有 m 个分区，每个分区 P_i 上执行 n_i 个任务，分区执行能力为 α_i，分区的任务总利用率 U_i 满足

$$U_i\leq U_{\text{bound}}=n_i((2/(2-\alpha_i))^{\frac{1}{n_i}}-1),$$

则分区可调度.

证明.　考虑某一个分区 P_i，其它 $(m-1)$ 个分区对本分区的阻塞，都可以看作是一周期任务 $\tau_0=\{T_{RL}(1-\alpha_i),T_{RL}\}$ 对分区 P_i 中任务的抢占形成的. 因此根据式(10)，直接得出定理 1 的结论.　证毕.

4　分区设计

4.1　分区设计方法

对于系统中运行的特定任务集，如果所采用的分区设计方法既能保证各个分区任务的可调度性，又能满足系统可调度约束条件，那么在该分区设计方法下，该任务集可实时执行，即系统是可调度的. 如果分区设计得太大，则分区占用系统资源太多，造成系统资源一定程度的浪费. 如果分区设计得太小，则不能保证系统的可调度性. 因此为每个分区分配合理的分区执行能力是分区设计的关键.

由定理 1 可知，只要 $U_i\leq n_i\left(\left(\frac{2}{2-\alpha_i}\right)^{\frac{1}{n_i}}-1\right)$，则分区 P_i 一定可调度. 在满足分区任务可调度的约束条件下，为了使系统的可调度利用率尽可能大，应取分区执行能力的最小值. 从匹配的角度令上式相等，可以得到

$$\alpha_i=2-2(U_i/n_i+1)^{-n_i}\qquad(11)$$

上式为本文提出的匹配型分区设计方法. 当给定分区任务时，按此方法进行分区设计，为每个分区分配合理的执行能力，则能保证各分区任务的可调度性.

对于有 m 个分区，每个分区 P_i 上执行 n_i 个任务的系统，分区设计方法的复杂度最多为 $O(mn)$，其中 $n=\max(n_i,i=1,2,\cdots,m)$. 复杂度开销是系统完全可以接受的.

4.2　系统可调度利用率分析

定理 2.　处理器系统有 m 个分区，每个分区 P_i 上执行的硬实时任务集 $\Gamma_i=\{\tau_1,\tau_2,\cdots,\tau_{n_i}\}$，如果每个分区的执行能力按照式(11)分配，只要系统的总利用率满足

$$U\leq m\left(\prod_{i=1}^{m}n_i\right)^{\frac{1}{m}}\left(\frac{2m}{2m-1}\right)^{\frac{1}{n_{\max}}}-\sum_{i=1}^{m}n_i,$$

则系统可调度.

证明.

(1) 分区的执行能力按照式(11)分配，根据定理 1 可知，各个分区可调度.

(2) $\sum_{i=1}^{m}\alpha_i=2m-2\sum_{i=1}^{m}(U_i/n_i+1)^{-n_i}$，由于 $(U_i/n_i+1)^{-n_i}$ 为 n_i 的单调递减函数，所以可得

$$\begin{aligned}\sum_{i=1}^{m}\alpha_i&\leq 2m-2\sum_{i=1}^{m}\left(\frac{U_i}{n_i}+1\right)^{-n_{\max}}\\&\leq 2m-2m\prod_{i=1}^{m}\left(\frac{U_i+n_i}{n_i}\right)^{-\frac{n_{\max}}{m}}\\&\leq 2m-2m\left(\prod_{i=1}^{m}n_i\right)^{\frac{n_{\max}}{m}}\left(\frac{\sum_{i=1}^{m}U_i+n_i}{m}\right)^{-n_{\max}}\\&=2m-2m\left(\prod_{i=1}^{m}n_i\right)^{\frac{n_{\max}}{m}}\left(\frac{U+\sum_{i=1}^{m}n_i}{m}\right)^{-n_{\max}}\\&\leq 2m-2m\left(\prod_{i=1}^{m}n_i\right)^{\frac{n_{\max}}{m}}\cdot\\&\quad\left(\left(\prod_{i=1}^{m}n_i\right)^{\frac{1}{m}}\left(\frac{2m}{2m-1}\right)^{\frac{1}{n_{\max}}}\right)^{-n_{\max}}=1,\end{aligned}$$

即满足系统可调度性约束条件.　证毕.

推论 1.　处理器有 m 个分区，每个分区 P_i 上执行的硬实时任务集 $\Gamma_i=\{\tau_1,\tau_2,\cdots,\tau_{n_i}\}$，每个分区的执行能力按照式(11)分配，$n_1=n_2=\cdots=n_m=n$，则系统的可调度利用率最小上界为

$$U_{\text{bound}}=mn\left(\frac{2m}{2m-1}\right)^{\frac{1}{n}}-mn.$$

证明.

(1) $n_1=n_2=\cdots=n_m=n$ 时，按定理 2 可知，

$U\leq mn\left(\frac{2m}{2m-1}\right)^{\frac{1}{n}}-mn$,则系统可调度.

(2)下面证明对于任意给定实数 ξ,$0<\xi<1$,至少存在一个任务集的利用率 $U\leq mn\left(\frac{2m}{2m-1}\right)^{\frac{1}{n}}-mn+\xi$,使得 $\sum_{i=1}^{m}\alpha_i>1$,即不满足系统可调度性约束条件.

对于给定的 m 和 n,假设 $\beta=\left(\frac{2m}{2m-1}\right)$,构造一个任务集,具有以下参数:

$$U_1=U_2=\cdots=U_m=n\beta^{\frac{1}{n}}-n+\xi/m,$$

$$U=\sum_{i=1}^{m}U_i\leq mn\beta^{\frac{1}{n}}-mn+\xi,$$

$$\alpha_1=\alpha_2=\cdots=\alpha_m=2-2\left(\frac{n\beta^{\frac{1}{n}}-n+\xi/m}{n}+1\right)^{-n}$$

$$=2-2\left(\beta^{\frac{1}{n}}+\frac{\xi}{mn}\right)^{-n},$$

则 $\sum_{i=1}^{m}\alpha_i=2m-2m\left(\beta^{\frac{1}{n}}+\frac{\xi}{mn}\right)^{-n}$,由于 $\beta^{\frac{1}{n}}+\frac{\xi}{mn}>1$,所以 $\left(\beta^{\frac{1}{n}}+\frac{\xi}{mn}\right)^{-n}$ 单调递减,因此

$$\left(\beta^{\frac{1}{n}}+\frac{\xi}{mn}\right)^{-n}<(\beta^{\frac{1}{n}})^{-n}=1/\beta,$$

则 $\sum_{i=1}^{m}\alpha_i>2m-2m\frac{1}{\beta}=1$. 证毕.

由定理 2 可知,各个分区的任务数 n_i 越大,系统可调度利用率上界越小.

推论 2. 处理器有 m 个分区,每个分区 P_i 上执行的硬实时任务集 $\Gamma_i=\{\tau_1,\tau_2,\cdots,\tau_{n_i}\}$,每个分区的执行能力按照式(11)分配,$n_i$ 都取较大值,则系统的可调度利用率最小上界为

$$U_{\text{bound}}=m\ln\left(\frac{2m}{2m-1}\right).$$

证明.

(1)分区的执行能力按照式(11)分配,根据定理 1 可知,各个分区可调度.

(2)$\alpha_i=2-2(U_i/n_i+1)^{-n_i}$,当 n_i 取较大值时,α_i 接近于 $2-2e^{-U_i}$,因此

$$\sum_{i=1}^{m}\alpha_i=2m-2\sum_{i=1}^{m}e^{-U_i}\leq 2m-2m\left(\prod_{i=1}^{m}e^{-U_i}\right)^{\frac{1}{m}}$$

$$=2m-2m\left(e^{-\sum_{i=1}^{m}U_i}\right)^{\frac{1}{m}}=2m-2m(e^{-U})^{\frac{1}{m}}$$

$$\leq 2m-2m\frac{2m-1}{2m}=1,$$

即满足系统可调度性约束条件.

(3)下面证明对于任意给定实数 ξ,$0<\xi<1$,至少存在一个任务集的利用率 $U\leq m\ln\left(\frac{2m}{2m-1}\right)+\xi$,使得 $\sum_{i=1}^{m}\alpha_i>1$,即不满足系统可调度性约束条件.

对于给定的 m 和 n_i,构造一个任务集,具有以下参数:

$$U_1=U_2=\cdots=U_m=\ln\left(\frac{2m}{2m-1}\right)+\frac{\xi}{m},$$

$$U=\sum_{i=1}^{m}U_i\leq m\ln\left(\frac{2m}{2m-1}\right)+\xi,$$

$$\alpha_i=2-2\left(\frac{\ln\left(\frac{2m}{2m-1}\right)+\xi}{n_i}+1\right)^{-n_i}.$$

当 n_i 取较大值时,$\alpha_i=2-2e^{-\ln\left(\frac{2m}{2m-1}\right)-\xi}$,

$$\sum_{i=1}^{m}\alpha_i=2m-(2m-1)e^{-\xi}>2m-(2m-1)=1.$$

证毕.

由推论 2 可知,当 m 的值越大,U_{bound} 越小. $m\to\infty$ 时,$\min(U_{\text{bound}})\approx 1/2$,即最坏情况下,系统可调度利用率不小于 50%.

5 计算机仿真实验

5.1 实时性比较

通过 C++建模,对处理器双层分区调度策略下的多个任务集进行了仿真实验,实验平台采用 VC++6.0.将匹配型的分区设计方法与耗尽型和均衡型两种分区设计方法下的调度情况进行了实验比较.耗尽型分区设计思想是期望分区每次都能占用全部处理器资源执行全部任务;均衡型分区设计思想是将处理器资源平均分配给每个分区[12].实验中,将任务延迟时间率作为衡量系统强实时性的指标,任务延迟时间率定义为任务的实际执行延迟时间与任务的截止期限的比率.最大延迟时间率、最小延迟时间率和平均延迟时间率分别表示任务执行的最大延迟时间、最小延迟时间和平均延迟时间与任务截止期限的比率.实验结果如图 3~图 5 所示,在仿真环境中,系统由两个分区组成,分区中包括周期性的实时任务,并且任务的截止时限与周期相等,表 1 给出了实验中的一组任务集,任务的执行时间和周期参数设置如表 1 所示.表 2 给出了实验的对比结果.

表 1 分区任务参数(C,T)

分区 1	分区 2
$A(1,28)$	$D(2,14)$
$B(3,43)$	$E(3,15)$
$C(5,45)$	$F(2,26)$

表 2　采用不同方法分区设计的仿真结果比较

比较参数	仿真时间	执行任务总数	处理器利用率/%	最大延迟时间率/%	平均延迟时间率/%	最小延迟时间率/%	超过截止时限任务数
耗尽型	3000	774	63.766	188.462	38.660	3.5714	12
均衡型	3000	775	63.800	103.846	29.788	3.5714	4
匹配型	3000	775	63.800	76.923	27.469	3.5714	0

从表 2 的结果可以看出，对于同一任务集，在相同的仿真时间内，如图 3～5 所示的 3 种分区设计方法的最大延迟时间率差别较大. 耗尽型和均衡型的分区设计均使得任务的实际执行延迟时间有可能超过截止时限（最大延迟时间率分别为 188.642%、103.846%，超过截止时限的周期任务数分别为 12 和 4），而匹配型的分区设计实验中，没有任务的实际执行延迟时间超过截止时限（最大延迟时间率 76.923%，超过截止时限的周期任务数为 0）. 仿真实验结果表明基于匹配型的分区设计可以更好地满足硬实时系统双层分区调度的实时性要求.

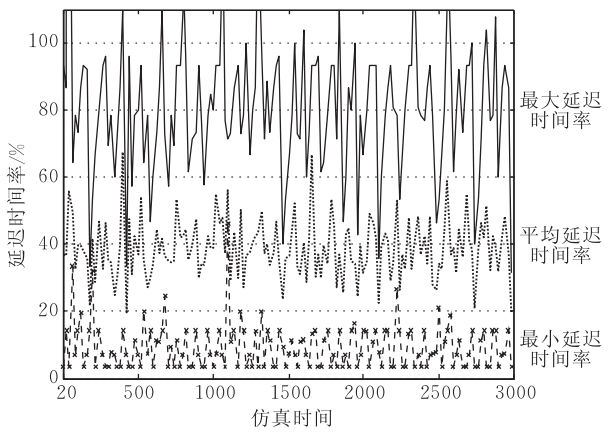

图 3　耗尽型分区设计情况下延迟时间率曲线

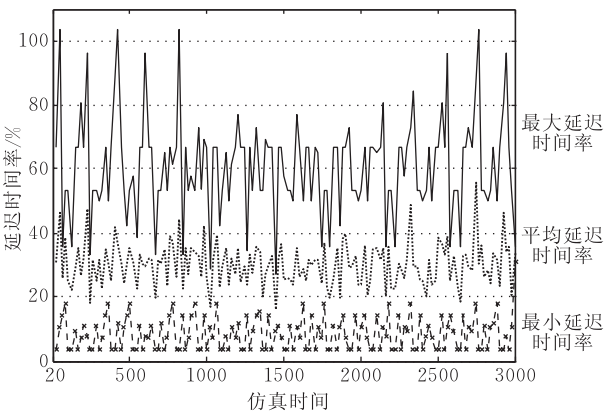

图 4　均衡型分区设计情况下延迟时间率曲线

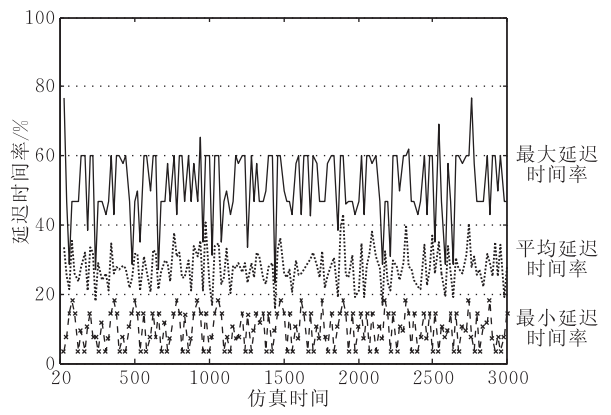

图 5　匹配型分区设计情况下延迟时间率曲线

5.2　分区可调度利用率比较

文献[10]给出了双层调度下分区可调度利用率的最小上限. 参照本文定理 1，可以得到分区执行能力下相应的分区可调度利用率最小上限 U_{bound}，其结果与文献[10]的比较如表 3 所示.

由表 3 分析可知，随着分区处理能力 α 的增加，分区最大可调度利用率的最小上限也相应增加. 对于相同的分区处理能力，同一分区中子任务数目越多，分区的最大可调度利用率最小上限越小. 相比文献[10]，本文给出的计算方法有效地提高了可调度利用率的最小上限.

表 3　分区可调度利用率最小上限

α	U_{bound}					
	$n=2$		$n=10$		$n=\infty$	
	本文	文献[10]	本文	文献[10]	本文	文献[10]
0.1	0.052	0.035	0.051	0.035	0.051	0.035
0.2	0.108	0.075	0.106	0.074	0.105	0.074
0.3	0.169	0.121	0.164	0.118	0.163	0.118
0.4	0.236	0.174	0.226	0.168	0.223	0.167
0.5	0.309	0.236	0.292	0.226	0.288	0.223
0.6	0.391	0.309	0.363	0.292	0.357	0.288
0.7	0.481	0.398	0.440	0.370	0.431	0.363
0.8	0.582	0.507	0.524	0.462	0.511	0.452
0.9	0.697	0.646	0.616	0.576	0.598	0.560
1.0	0.828	0.828	0.718	0.718	0.693	0.693

注意到 $\alpha=1$ 的情况，分区可调度利用率最小上限取最大值，此时分区占用全部系统资源，即相当于单层的 RM 调度，两种方法得到的结果都与 Liu 和 Layland[14] 所推导的 RM 可调度利用率上限相一致，α 取其它值时，本文的方法相比文献[10]，在其基础上至少提高了 6.7%.

5.3　分区设计成功率

仿真实验中，使用分区设计成功率（successful

design percentage)作为衡量分区设计算法好坏的一个性能指标.成功分区设计是指能找到满足系统可调度约束条件的分区参数,使得系统所有分区内的任务可成功调度,找不到这样的分区参数,则说分区设计失败.分区设计成功率越高,说明分区设计算法越好.

系统分别由 2 个分区、3 个分区、5 个分区组成,任务的参数根据下面的方法产生:

(1)每个分区中任务的个数在 2～10 个之间随机选择,服从均匀分布;

(2)任务最坏情况下的执行时间 C_i 在 2 个时间单位到 30 个时间单位之间随机选择,服从均匀分布;

(3)每个任务实例 τ_i 的松弛时间 l_i 服从 100～400 之间的均匀分布;

(4)任务周期 $T_i=C_i+l_i$.

实验结果是采用 200 次独立实验结果的平均值,为了比较算法在不同的系统负载情况下的性能,实验中系统负载率的取值范围为 0.4～0.8.

图 6 给出了分区数目分别为 2、3、5 时,不同的系统负载下分区设计成功率的变化趋势.从图中我们不难发现,只要系统负载率不大于 50%,则调度成功率为 100%,这与推论 2 的结论相一致.当系统负载率大于 50%之后,随着系统负载的增加,SDP 呈下降趋势.在相同的负载情况下,随着分区数目的增加,SDP 呈下降趋势.这是由于分区数越多,任务越多,系统更复杂,因此保证系统实时性的分区设计成功率越低.

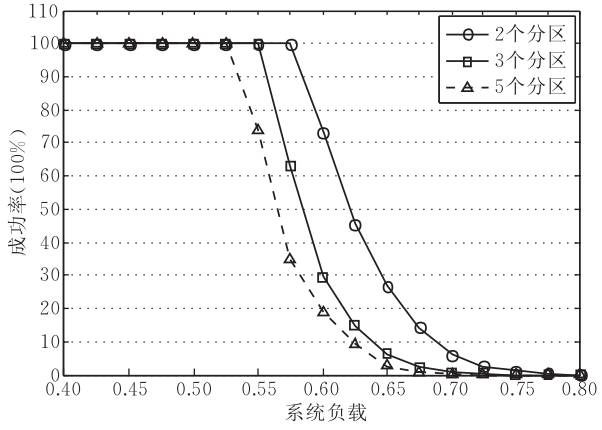

图 6 分区设计成功率曲线

6 结 论

一般的可调度性研究大多集中于最大响应时间分析的方法,它的分析结果对任务的周期和执行时间的值敏感,缺少鲁棒性.在工程设计的初始阶段,任务集合也许不能完备,可调度利用率对于指导我们在设计时灵活选择任务具有重要意义.

本文根据 ARINC 653 标准,建立了可行的分区调度模型,通过可调度利用率,对系统的实时性能进行了理论分析.本文给出了分区可调度利用率的最小上限.提出了与分区利用率匹配的分区设计方法,证明了在最坏情况下,系统可调度的利用率不小于 50%,并进行了仿真验证.为硬实时系统双层分区调度设计提供了理论分析基础.

参 考 文 献

[1] ARINC 653-1-2003. Avionics application software standard interface. ARINC Specification 653, 2003

[2] Zou Yong, Huai Xiao-Yong, Li Ming-Shu. The adaptive scheduling approaches to open real-time systems. Chinese Journal of Computers, 2004, 27(1): 58-65(in Chinese)
(邹勇,淮晓永,李明树. 开放式实时系统中的自适应调度方法. 计算机学报, 2004, 27(1): 58-65)

[3] Lorente J L, Palencia J C. An EDF hierarchical scheduling model for bandwidth servers//Proceedings of the 12th Embedded and Real-Time Computing Systems and Applications. Sydney, Australia, 2006: 261-266

[4] Zhang Feng-Xiang, Burns Alan. Analysis of hierarchical EDF pre-emptive scheduling//Proceedings of the 28th IEEE International Real-Time Systems Symposium. Tucson, Arizona, 2007: 423-434

[5] Easwaran Arvind, Lee Insup, Sokolsky Oleg, Vestal Steve. A compositional scheduling framework for digital avionics systems//Proceedings of the IEEE Real-Time Computing Systems and Applications. Beijing, China, 2009: 371-380

[6] Lipari G, Bini E. Resource partitioning among real time applications//Proceedings of the 15th Euromicro Conference on Real-Time Systems. Porto, Portugal, 2003: 151-158

[7] Almeida L, Pedreiras P. Scheduling within temporal partitions: Response-time analysis and server design//Proceedings of the 4th ACM International Conference on Embedded Software. Pisa, Italy, 2004: 95-103

[8] Davis R I, Burns A. Hierarchical fixed priority preemptive scheduling//Proceedings of the 26th IEEE Real-Time Systems Symposium. Miami, USA, 2005: 389-398

[9] Kuo T W, Li C H. A fixed priority driven open environment for real-time applications//Proceedings of the IEEE Real-Time Systems Symposium. Phoenix, USA, 1999: 256-267

[10] Saewong S, Rajkumar R, Lehoczky J, Klein M. Analysis of hierarchical fixed-priority scheduling//Proceedings of the Euromicro Conference on Real-Time Systems. Vienna, Australia, 2002: 173-181

[11] Davis R I, Burns A. An investigation into server parameter selection for hierarchical fixed priority pre-emptive systems//Proceedings of the Real-Time and Network Systems. Rennes, France, 2008: 19-28

[12] Liu J W S. Real-Time Systems. New Jersey Upper Saddle River: Prentice Hall, 2000

[13] Li Xin-Ying, Xiong Hua-Gang. Modeling and simulation of integrated modular avionics//Proceedings of the 28th Digital Avionics Systems Conference. Orlando, USA, 2009: 7. B. 3. 1-7. B. 3. 8

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

Li Xin-Ying, born in 1984, Ph. D. candidate. Her current research interests include real-time systems, integrated modular avionics systems.

GU Jian, born in 1976, Ph. D. candidate. His current research interests focus on avionics systems architecture.

HE Feng, born in 1980, Ph. D.. His main research interests focus on avionics network communication and real time scheduling.

XIONG Hua-Gang, born in 1961, Ph. D., professor. His main research interests include embedded systems and digital communication network.

Background

The work is supported by the National Natural Science Foundation of China under the grant No. 60879024.

In an integrated real time system, applications of diverse levels of temporal and mission criticality are supposed to share the same computing resources while maintaining their own functional and temporal behaviors. To protect applications from potential interference, the system must provide strong partitioning schemes which include spatial and temporal separation. There is currently considerable interest in hierarchical and partition scheduling as a way of providing isolation between applications in integrated systems.

A good example of an integrated real-time system is the Integrated Modular Avionics, which is being embraced by the aerospace industry these days. ARINC 653 standards which introduced a hierarchical and partition scheduling in avionics, is developed and adopted by Airlines Electronic Engineering Committee (AEEC), but the problem of providing real-time capability in two-level partition scheduling under strong partition constraints is not solved satisfactorily and remains to be answered.

The authors have made researches in the fields of avionics systems real-time and fault-tolerant scheduling. According to standard, built two-level partition scheduling model, the problem to run periodic tasks on time with minimum amounts of resources and good utilization of every resource is studied. A partition schedulability condition based on worst case has been derived. A new partition design approach is proposed based on processor utilization matching. The necessary and sufficient conditions of guaranteeing task deadlines under the new partition design approach have been derived. The least upper bounds to system utilization for the scheme have also been derived and formally proved. Based on this new method, successful partition design percentage is analyzed.

This paper contributes on the problem of the two-level partition scheduling problem of strongly partitioned hard real time systems. The simulation results show that the proposed design method has significantly better performance compared with the existing methods. In short, the above mentioned works lay a theoretical foundation for engineering applications of hard real time two-level partition scheduling under strong partition constraints.