ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.3, March 2006, pp.611−619
DOI: 10.1360/jos170611

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种严格按比例派发服务的混合实时调度算法*

龚育昌, 王立刚+, 陈香兰, 齐 骥

(中国科学技术大学 计算机科学与技术系,安徽 合肥 230027)

A Hybrid Real-Time Scheduling Algorithm Based on Rigorously Proportional Dispatching of Serving

GONG Yu-Chang, WANG Li-Gang+, CHEN Xiang-Lan, QI Ji

(Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)

+ Corresponding author: Phn: +86-551-3620501, E-mail: wanglg@mail.ustc.edu.cn, http://www.ustc.edu.cn

Gong YC, Wang LG, Chen XL, Qi J. A hybrid real-time scheduling algorithm based on rigorously proportional dispatching of serving. *Journal of Software*, 2006,17(3):611−619. http://www.jos.org.cn/1000-9825/17/611.htm

Abstract: In hybrid real-time systems, schedulers must guarantee that all of hard real-time jobs are finished by their deadlines and the QoS of soft real-time tasks and non real-time tasks are improved as greatly as possible. This paper presents RPDS (rigorously proportional dispatching server) algorithm, and constructs a hierarchical scheduling framework based on that. RPDS partitions CPU time flow into continuous segments, and in each segment RPDS will forcibly assign one time slice to non-hard real-time tasks. Experimental results show that RPDS can allocate processor time to various application classes reasonably and reduce the deadline miss ratio of real-time tasks effectively.

Key words: hybrid real-time system; scheduling; deadline miss ratio; CPU utilization; proportional dispatching

摘 要: 在混合实时系统中,调度器必须既保证所有硬实时任务严格按照其时间约束在截止期内完成,又要尽可能地提高软实时任务和非实时任务的服务质量.提出了一种严格按比例派发服务器算法(RPDS),并以此为基础构建了一种层次式调度框架.RPDS 将处理器时间流分成连续的小段,并在每一小段中强制为非硬实时任务分配一个时间片.实验结果表明,RPDS 可以合理地为各种类型应用分配处理器时间,并且降低了实时任务的截止期错失率.

关键词: 混合实时系统;调度;截止期错失率;CPU 利用率;按比例派发
中图法分类号: TP316 文献标识码: A

近年来,嵌入式实时系统的系统结构日益复杂.而随着多媒体及网络技术的广泛普及,操作系统中也不断加入了更多的实时任务,并呈现出软硬件相互渗透的趋势.这就给调度系统提出了新的挑战,如何对包括硬实时、软实时、最大努力等多种不同时间要求应用的混合实时系统进行合理、有效的调度已成为人们研究的热点.

* Supported by the National Natural Science Foundation of China under Grant No.60273042 (国家自然科学基金); the Natural Science Foundation of Anhui Province of China under Grant No.03042203 (安徽省自然科学基金)

Received 2005-04-21; Accepted 2005-07-21

针对混合调度问题,研究人员提出了一些解决方法.文献[1,2]分别提出了基于 CUS(constant utilization server),TBS(total bandwidth server)的预留带宽算法,并以此为基础构建了双层调度框架,解决了周期性、偶发性实时任务并存的调度问题,并为任务提供了隔离性.但是该方法需要预先知道精确的实时任务运行参数,并将硬实时、软实时任务同等对待,软实时任务时间参数的变化可能会影响硬实时任务的按时完成.文献[3]提出的基于 CBS(constant bandwidth server)的算法,在不影响硬实时任务执行的前提下可使软实时任务的平均时延达到最小.但它对硬实时的调度时序并未作出调整,从而使得一些本可按时完成的软实时任务错过了截止期.文献[4]提出的 PShED(processor sharing with earliest deadline first)算法为各个任务提供了独立性,使之互不影响,当系统总利用率小于或等于 1 时,只要每个任务都可以在相应利用率的专用处理器上可调度,系统利用该算法也是可调度的.然而,PShED 算法是在任务的级别上提供调度独立性,需要为每个任务维护相应数据结构并进行大量计算,在很大程度上降低了系统的效率.文献[5]对上述各种方法进行了较为深入的探讨.文献[6]提出的 CBS-R(CBS with resources constraints)方法可以使硬实时、软实时任务之间实现资源共享.文献[7]提出的 RBED(rate-based earliest deadline scheduler)策略将传统调度器分为资源分配器和派发器,利用松弛时间管理机制使不同类型应用并存于同一系统中.但在该策略中,只有最大努力任务可以利用实时任务的松弛时间,软实时任务却不能从中获得更多时间,这就极大地限制了其应用范围.

针对已有方法的局限性,本文提出了一种新的基于严格按比例派发服务器的混合实时调度算法——RPDS (rigorously proportional dispatching server).该算法将不同类型实时任务分开,由不同调度器调度,然后由 RPDS 统一派发时间片,在保证硬实时任务不受其他类型任务影响的基础上,使软实时任务的截止期错失率最小化.

1 任务模型

在描述 RPDS 之前,需要说明该算法所处运行环境和各任务的时间特征参数.假设在系统 S 中存在一系列相互独立的任务 T_i,其中 i=1,2,…,n,按照各任务时间约束的不同将其分为 3 类:硬实时任务、软实时任务和最大努力任务,任务集分别表示为 S_H, S_S, S_{BE}.硬实时任务需要系统对其时间约束给予严格的保证,截止期的错失将导致致命错误或引发灾难性后果.相反地,软实时任务错过截止期并不会引发非常严重的后果,从而允许一定比例的截止期错失.最大努力任务,顾名思义,越早完成越好,没有具体时间约束,但也要防止它们发生饿死的情况.

RPDS 参照经典的周期性任务模型[8,9],系统中任务的参数如下:

$J_{i,j}$:组成任务 T_i 的一系列作业,其中 j=1,2,…

$r_{i,j}$:作业 $J_{i,j}$ 的释放时间,即从时刻 $r_{i,j}$ 开始作业 $J_{i,j}$ 就可以开始执行了.

p_i:任务 T_i 的周期,即每过长度为 p_i 的时间任务 T_i 就有一个新的作业被释放.

ϕ_i:任务 T_i 的相位,即任务 T_i 的第 1 个作业的释放时间,$\phi_i = r_{i,1}$.

e_i:任务 T_i 的执行时间.如果 T_i 是硬实时任务,则 e_i 就是作业在最坏情况下无中断的执行时间;如果 T_i 是软实时任务或最大努力任务,则 e_i 即为一个统计值或估计值.

u_i:任务 T_i 的 CPU 利用率,有 $u_i = e_i / p_i$.进一步地,U_H 表示系统 S 中所有硬实时任务利用率之和,U_{Non-H} 表示系统 S 中非硬实时任务即软实时、最大努力等任务利用率之和.

m_i:任务 T_i 允许的截止期错失率,若 T_i 为硬实时任务,$m_i = 0$;若 T_i 为软实时任务,$0 < m_i < 1$;若 T_i 为最大努力任务,$m_i = 1$.从 m_i 的取值可以区分 T_i 属于哪类任务.

在 RPDS 中,硬实时任务的最坏情况执行时间 e_i 是经过事先计算和实验确定的,在整个执行过程中不发生变化;软实时、最大努力任务的执行时间 e_i 则可能会随时间而发生变化.为了简化分析,假设作业的相对截止期都等于该任务的周期,那么作业的绝对截止期就是同一任务中下一个作业的释放时间,实际的实时系统一般都符合这个假设.在本模型中,暂不考虑不可抢占区域对实时调度的影响,每个任务在任何时刻都是可以抢占的.

2 严格按比例派发服务器

注意到硬实时任务只需在截止期之前完成即可,提前完成意义不大,于是系统就可以在适当的时机将硬实

时任务推迟执行适当时间,从而给予其他类型任务更早执行的机会.因此,RPDS 将所有硬实时任务看作一个整体,内部利用 EDF 算法调度,但通过统一的派发服务器将硬实时任务的执行时间分布成较均匀的时间片,而不是连续的大块时间,从而可以把软实时、最大努力任务等非硬实时任务插入到硬实时任务的执行间隙中执行,避免了一些软实时任务错过截止期的情况发生.

2.1 基本概念

定义 **1**. 强制派发轮回(forcibly dispatching round,简称 FDR),系指 RPDS 在固定长度时间段内强制非硬实时任务获得单位时间的处理器时间,其长度定义为

$$Round = 1/(1-U_H) \tag{1}$$

FDR 的时间单位为系统的基本时间片长度.RPDS 派发出的时间序列就是由一个个连续的 FDR 组成的.

在系统 S 中加入 *Idle* 任务,赋予其最低优先级,系统在没有其他任务执行时将执行 *Idle* 任务.如果实时任务的总利用率不大于 1,那么,有如下命题成立:$U_H + U_{Non-H} = 1$.于是,式(1)就可以改写为

$$Round = 1/(1-U_H) = 1/U_{Non-H} \tag{2}$$

2.2 基于**RPDS**的调度框架

基于 RPDS,可以构建出如图 1 所示的层次式树状结构的实时调度框架.在此框架中,RPDS 对硬实时、软实时、最大努力等任务集进行类调度,各任务集就绪队列分别由各自的调度器排序.每个时间片开始时,RPDS 选出下一时间片应由哪类任务执行,再从已由该类任务调度器排好序的就绪队列中取出第 1 个执行.目前,硬实时、软实时任务各自的调度器均为 EDF 算法,以后会加入对速率单调(RM)[8]等算法的支持.硬实时任务之外的任务统称为非硬实时任务,RPDS 在每个 FDR 中优先将处理器派发给硬实时任务,但要保证在每个 FDR 中派发给非硬实时任务单位时间片.软实时、最大努力任务和 *Idle* 任务由非硬实时任务调度器调度.对最大努力任务有两种处理方法:使软实时任务具有相对于最大努力任务绝对的优先权,但在实时任务负载较高的系统中,这会使最大努力任务产生饿死的情况;或者给最大努力任务预留一个最小处理器配额 U_{BE},满足 $U_H + U_S \le 1 - U_{BE}$,将最大努力任务集作为一个普通软实时任务由软实时任务调度器统一调度,以防止此类任务的饿死.具体使用哪种方法视具体需求而定. *Idle* 任务被赋予最低优先级,只有在系统中没有其他任务需要执行时才执行 *Idle* 任务.

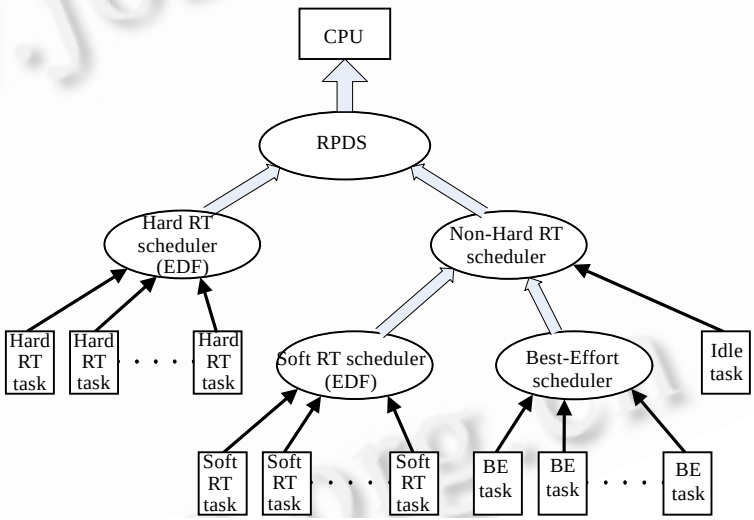

Fig.1 RPDS based scheduling framework

图 1 基于 RPDS 的调度框架

2.3 **RPDS**算法描述

在 RPDS 算法中,通过调整单位时间片的大小,可以很容易地使各种任务的周期、执行时间、相位均为整数.

2.3.1 强制派发轮回 FDR 的长度

由定义 1 可知, FDR 的长度为 $Round = 1/(1-U_H)$,是一个精确的值.而系统中任务的时间参数都是整数,因此我们对它略作调整:从时刻 0 开始依次为第 $1,2,...$ 个 FDR,第 x 个 FDR 的开始时刻为上一个 FDR 的结束时刻,结束时刻为 $\lceil xRound \rceil$.现在,任务的周期、执行时间、相位和 FDR 的开始、结束时刻都变为整数,RPDS 将只在

整数时刻检查各类任务的执行情况,而且单次派发时间片的长度就是单位时间片.

2.3.2 非硬实时任务处理器时间预算 $Budget_{Non-H}$

下面是 $Budget_{Non-H}$ 的消耗和补充规则.

消耗规则:系统中非硬实时任务执行时 $Budget_{Non-H}$ 将消耗对应的值.

补充规则:

(1) 初始状态,$Budget_{Non-H}$=0.

(2) 每个 FDR 的初始时刻,补充 $Budget_{Non-H}$,$Budget_{Non-H}$= $Budget_{Non-H}$+1,即在每个 FDR 内非硬实时任务将获得单位时间片的处理器配额,其余时间分给硬实时任务.

2.3.3 RPDS 算法详述

RPDS 在每个时间片开始时对系统状态进行判断,以确定下一时间片派发给哪类任务.设 t 为一个时间片的开始时刻,那么 RPDS 在 t 时刻执行以下步骤:

步骤 1. 检查各类型任务是否有新作业释放.若有,则将新作业由该类型调度器排序加入该类型就绪队列.每当系统将时间片派发给该类型任务时,只需取队首的作业运行即可.

步骤 2. 如果 $t=\lceil xRound \rceil$,即 t 为一个 FDR 的初始时刻,补充 $Budget_{Non-H}$, $Budget_{Non-H}=Budget_{Non-H}+1$.

步骤 3. 如果 $t\neq\lceil xRound \rceil-1$,即 t 开始的时间片不是一个 FDR 的最后一个时间片:此时,若有硬实时作业就绪,优先将下一时间片派发给硬实时任务;否则,将下一时间片派发给非硬实时任务,并将 $Budget_{Non-H}$ 减 1 .

步骤 4. 如果 $t=\lceil xRound \rceil-1$,即 t 开始的时间片是一个 FDR 的最后一个时间片:此时,若 $Budget_{Non-H}>0$,那么该 FDR 中非硬实时任务还未执行过,将下一时间片强制派发给非硬实时任务,并将 $Budget_{Non-H}$ 减 1 ;否则,将下一时间片分配给硬实时任务.

步骤 5. 该时间片过后,检查当前作业是否运行完毕,若已完毕,则将其从该类型任务就绪队列中删除.

步骤 6. 循环往复直至系统关闭.

2.3.4 示 例

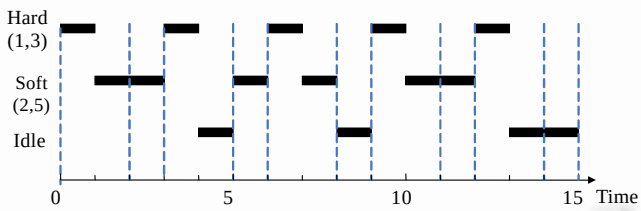

Fig.2 Illustration of RPDS

图 2 RPDS 示例

为简便起见,我们用 (e_i, p_i) 表示任务 T_i 的特征参数,各任务的相位均为 0.假设系统 S 中有一个硬实时任务,特征参数为(1,3);一个软实时任务,特征参数为(2,5).没有最大努力任务,系统空闲时将执行 *Idle* 任务.此系统由 RPDS 派发的时间序列如图 2 所示.其中竖直的虚线表示每个 FDR 的起始时刻.可以看到,每个 FDR 中都有一个时间片是分配给非硬实时任务的.

3 RPDS 的理论分析

下面的两个定理说明了 RPDS 算法的特性.为方便分析,最大努力任务的优先级始终处理为比实时任务要低.

引理 **1**. 每个 FDR 中 RPDS 只会派发给非硬实时任务一个时间片,并且总是把最后一个时间片派发给非硬实时任务.

证明:假设 RPDS 在第 $x+1$ 个 FDR 中第 1 次破坏了引理 1,即在该 FDR 中给非硬实时任务派发了一个以上的时间片,或者只派发了一个时间片但不是该 FDR 中的最后一个.设该 FDR 中第 1 个非硬实时任务时间片是在时刻 t 派发的,那么上述两种情况都满足下面两个命题:

(i) 因为该 FDR 中时刻 $t+1$ 后至少还有一个时间片,不论是派发给硬实时任务还是非硬实时任务,都有 $t+1\leq\lceil (x+1)Round \rceil-1$.

(ii) 时刻 t 开始的时间片并非该 FDR 中的最后一个,那么将该时间片派发给非硬实时任务的原因是 t 时没

有硬实时任务在等待执行.从而有 $t-x\geq\lceil(t+1)/p_k\rceil e_k$,即时刻 t 前释放的硬实时任务均已执行完毕.

进一步有 $t-x\geq\lceil(t+1)/p_k\rceil e_k\geq(t+1)U_H$,

$$1-U_H\geq(x+1)/(t+1)\geq(x+1)/(\lceil(x+1)Round\rceil-1)>(x+1)/((x+1)Round)=1-U_H,$$

即 $1-U_H>1-U_H$,而这是不可能的,引理 1 得证. □

定理 **1**. 如果系统 S 中具有独立的可抢占的硬实时、软实时和最大努力任务,实时任务的周期、执行时间、相位都为整数,并且实时任务的相对截止期都等于各自的周期,那么,系统在单处理器上按照 RPDS 派发策略能够使硬实时任务都不错过截止期,当且仅当硬实时任务的利用率小于或等于 1.

证明:如果硬实时任务的总利用率大于 1,显然系统过载,将不能被可行地调度,条件的必要性得证.条件的充分性通过反证法来证明.

不妨设满足条件的系统 S 从时刻 0 开始执行,在时刻 t 第 1 次有硬实时作业错过截止期,该作业为任务 T_i 中的作业 $J_{i,c}$,且时刻 t 位于第 $x+1$ 个 FDR 中.此时有两种情形,如图 3 所示:

(1) 每个硬实时任务的当前作业都在时刻 $r_{i,c}$ 或它之后开始. $r_{i,c}$ 是作业 $J_{i,c}$ 的释放时间.

(2) 有些硬实时任务的当前作业在时刻 $r_{i,c}$ 之前开始.

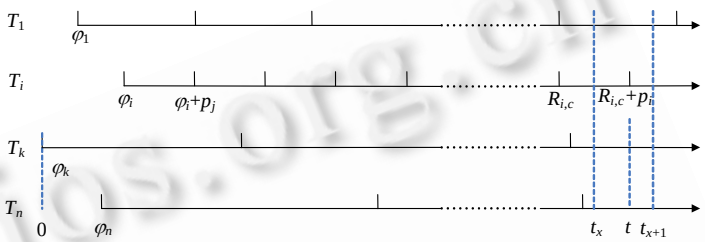

Fig.3 Case (1) of the theorem 1

图 3 定理 1 情形

下面将就上述两种情形进行讨论,这些讨论均基于这样一个命题:

命题 **1**. 时刻 t 之前每个 FDR 中硬实时任务都使用了足额的处理器配额,即每个 FDR 中除了一个时间片派发给非硬实时任务外都派发给了硬实时任务.

在上述两种情形讨论完毕后,我们将说明命题 1 为什么成立.

情形(1):如图 3 所示,任务时间线上的标记表示作业的释放时间,虚线标出了时刻 $0,t_x,t,t_{x+1}$,其中,t_x 为 $\lceil xRound\rceil$, t_{x+1} 即为 $\lceil(x+1)Round\rceil$. $J_{i,c}$ 在时刻 t 错过了截止期,这说明:截止期在 t 之后的所有当前硬实时作业在 t 之前都未获得过处理器,因为按照 EDF 策略,它们比 $J_{i,c}$ 的优先级要低;而且,完成作业 $J_{i,c}$ 和所有截止期在 t 或 t 之前的硬实时作业所需的总处理器时间超过了总的可用时间 $\lceil xRound\rceil-x+b$,其中 b 是 RPDS 在第 $x+1$ 个 FDR 中 t 之前派发给硬实时任务的处理器时间.那么,有如下不等式成立:

$$\lceil xRound\rceil-x+b<\sum_{T_k\in S_H}\lfloor(t-\phi_k)/p_k\rfloor e_k,$$

不等式右边即截止期在 t 或 t 之前的所有硬实时作业所需总处理器时间,记为 E.由于不等式两边都为整数,有:

$$\lceil xRound\rceil-x+b+1\leq E\leq t\sum_{T_k\in S_H}\frac{e_k}{p_k}=tU_H=\lceil xRound\rceil U_H+fracU_H,$$

$$fracU_H\geq\lceil xRound\rceil(1-U_H)-x+b+1\geq xRound(1-U_H)-x+b+1=b+1,$$

其中 $frac=t-\lceil xRound\rceil$,由引理 1 可知 $frac\leq b+1$,结合上面得到的 $fracU_H\geq b+1$,我们得到 $frac=b+1$,并且等式成立的条件之一为 $U_H=1$.当 $U_H=1$ 时,RPDS 就退化成了普通的 EDF 调度器,根据文献[8]我们知道情形(1)下不会有硬实时任务错过截止期的情况出现.

情形(2):如图 4 所示.用 S_{Sub-H} 表示 S_H 的一个子集,该子集包含所有当前作业在 $r_{i,c}$ 之前释放的硬实时任务.在 $r_{i,c}$ 之前,RPDS 派发给硬实时任务的处理器时间可能分配给了 S_{Sub-H} 中某些任务的当前作业,图中 T_l 就是一个这样的任务.如果没有这样的任务,那么第 2 种情形的证明与第 1 种完全相同,因此仅需考虑存在的情况.

图中阴影部分为 S_{Sub-H} 中任务的当前作业最后一次获得的处理器时间,设 t_{-1} 为其终点,它在第 x_{-1} 个 FDR 中.在 $t_{-1}\sim t$ 时间段中任何截止期在 t 之后的当前作业将不会获得处理器, S_H-S_{Sub-H} 中任务 T_k 在该时间段中的相位记为 ϕ_k'.因为 $J_{i,c}$ 在时刻 t 错过了截止期,所以有:

$$(\lceil xRound\rceil-\lceil x_{-1}Round\rceil)-(x-x_{-1})+b_{-1}+b<\sum_{T_k\in S_H-S_{Sub-H}}\lfloor(t-t_{-1}-\phi_k')/p_k\rfloor e_k,$$

其中 b_{-1} 为第 x_{-1} 个 FDR 中 t_{-1} 后派发给硬实时任务的处理器时间, b 为第 $x+1$ 个 FDR 中 t 前派发给硬实时任务的处理器时间.不等式左边为 $t_{-1}\sim t$ 时间段中硬实时任务实际得到的处理器时间;右边为该时间段中硬实时任务应得的处理器时间,记为 E' .令 $frac_{-1}=\lceil x_{-1}Round\rceil-t_{-1}$, $frac=t-\lceil xRound\rceil$.由不等式两边都为整数,有

$$(\lceil xRound\rceil-\lceil x_{-1}Round\rceil)-(x-x_{-1})+b_{-1}+b+1\le E'\le\sum_{T_k\in S_H-S_{Sub-H}}\frac{t-t_{-1}}{p_k}e_k=(t-t_{-1})U_{H-Sub-H}<(t-t_{-1})U_H$$

$$=(\lceil xRound\rceil-\lceil x_{-1}Round\rceil)U_H+(frac+frac_{-1})U_H,$$

$$(\lceil xRound\rceil-\lceil x_{-1}Round\rceil)(1-U_H)-(x-x_{-1})+b_{-1}+b+1<(frac+frac_{-1})U_H=frac+frac_{-1}-(frac+frac_{-1})(1-U_H),$$

$$(\lceil xRound\rceil+frac-(\lceil x_{-1}Round\rceil-frac_{-1}))(1-U_H)+b_{-1}+b+1<frac+frac_{-1},$$

$$(t-t_{-1})/Round-(x-x_{-1})+b_{-1}+b+1<frac+frac_{-1}\qquad(3)$$

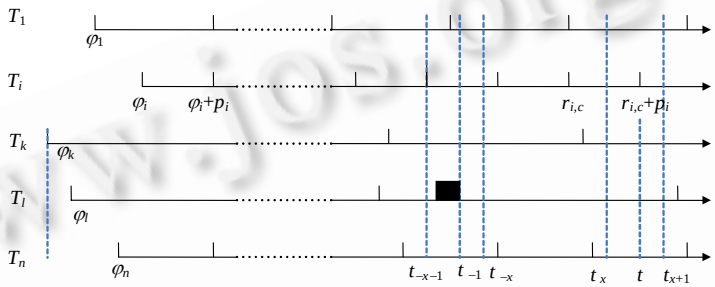

Fig.4 Case (2) of the theorem 1

图 4 定理 1 情形(2)

根据 $(t-t_{-1})/Round-(x-x_{-1})$ 符号的不同,对式(3)分两种情况讨论:

(i) 当 $(t-t_{-1})/Round-(x-x_{-1})\ge 0$ 时, $b_{-1}+b+1<frac+frac_{-1}$.因为其中各项均为整数,由其取值范围可知 $frac=b+1$, $frac_{-1}=b_{-1}+1$,进而由引理 1 有 $t=\lceil(x+1)Round\rceil$, $t_{-1}=\lceil x_{-1}Round\rceil-b_{-1}-1$.上述 4 个等式代入式(3),有

$$(\lceil(x+1)Round\rceil-(\lceil x_{-1}Round\rceil-b_{-1}-1))/Round-(x-x_{-1})+b_{-1}+b+1<b_{-1}+b+2,$$

$$(\lceil(x+1)Round\rceil-(\lceil x_{-1}Round\rceil-1))/Round-(x+1-x_{-1})+b_{-1}<0,$$

而 $(\lceil(x+1)Round\rceil-(\lceil x_{-1}Round\rceil-1))/Round-(x+1-x_{-1})>((x+1)Round-x_{-1}Round)/Round-(x+1-x_{-1})=0$,由此我们得到 $b_{-1}<0$,但这是不可能的.

(ii) $(t-t_{-1})/Round-(x-x_{-1})<0$ 只可能出现在 $t_{-1}=\lceil x_{-1}Round\rceil$ 与 $t=\lceil xRound\rceil$ 同时成立的情况下,否则根据 t_{-1} , t 的取值范围可知:

$$(t-t_{-1})/Round-(x-x_{-1})\ge(\lceil xRound\rceil-(\lceil x_{-1}Round\rceil-1))/Round-(x-x_{-1})>(x-x_{-1})-(x-x_{-1})=0.$$

由 t_{-1} , t 的值可知 $frac_{-1}=frac=b_{-1}=b=0$,代入不等式(3),有

$$(\lceil xRound\rceil-(\lceil x_{-1}Round\rceil-1))/Round-(x-x_{-1})+U_H<0.$$

可知 $U_H<0$,这也是不可能的.

(i)和(ii)都得到了不可能的结果,所以在情形(2)中硬实时任务也是不会错过截止期的.

现在来看命题 1,用反证法证明.如果存在硬实时任务未使用完足额处理器配额的 FDR ,不妨设 t 前最后一个满足命题的 FDR 是第 x_{-2} 个 FDR .那么 $\lceil x_{-2}Round\rceil$ 前释放的硬实时任务均已执行结束.有下式成立:

$$(\lceil xRound\rceil-\lceil x_{-2}Round\rceil)-(x-x_{-2})+b<\sum_{T_k\in S_H}\lfloor(t-\lceil x_{-2}Round\rceil-\phi'')/p_k\rfloor e_k.$$

类似于情形(1)的推导,很容易得出 $frac=b+1$ 且 $U_H=1$,从而得出同样的矛盾.

至此,定理 1 中条件的充分性证明完毕,定理 1 得证. □

定理 **2**. 如果系统 S 中具有独立的可抢占的硬实时、软实时和最大努力任务,实时任务的周期、执行时间、相位都为整数,并且实时任务的相对截止期都等于各自的周期,那么,系统在单处理器上按照 RPDS 派发策略能够使软实时任务都不错过截止期,当且仅当硬实时、软实时任务的总利用率小于或等于 1.

类似于定理 1 的证明方法,很容易证明定理 2.

定理 1 说明,在 RPDS 策略下,硬实时任务的可调度性只与硬实时任务的负载率相关,RPDS 为不同实时应用提供了严格的隔离.定理 2 说明,RPDS 在保证硬实时时间约束基础上,降低了软实时任务的截止期错失率.

4 仿真实验及结果分析

4.1 仿真实验方案

我们分别在系统中各任务时间参数无变化及软实时任务时间参数随机变化两种情况下,将 RPDS 与另外两种方法做了仿真实验比较.第 1 种方法是我们设计的分离的 EDF 方法,记为 SEDF,它将硬实时、软实时任务分开调度,均采用 EDF 策略,且硬实时任务相对于软实时任务具有绝对的优先权.第 2 种方法是 CUS 策略[3].这两种方法分别代表了将硬实时任务与软实时任务简单分开和等同对待两种方向.

为简便起见,仅用 (e_i, p_i) 表示实时任务的特征参数,各任务的相位 ϕ_i 均为 0,这对实验结果不会产生影响.系统中存在 6 个实时任务,分别为硬实时或软实时任务.用 *Idle* 任务代表最大努力任务,系统中没有实时任务可运行时执行 *Idle* 任务.选取 15 以内的随机自然数作为任务的 e_i , p_i 来模拟任务时间参数无变化的系统,条件是 $e_i < p_i$,并且系统总利用率不超过 1.在此基础上,随机抽取系统中的一个软实时任务,将其 e_i 随机增加一个整数值但增量不超过其 p_i 与原 e_i 之差,依次来模拟软实时任务时间参数变化的混合实时系统.

4.2 实验结果比较与分析

截止期错失率(deadline miss ratio,简称 DMR)[10]是指系统中未在截止期前完成的实时作业数与实时作业总数之比,是一个常用的衡量调度策略的性能指标.截止期错失率越低,调度成功率越高.

图 5 显示了实时任务时间参数无变化情况下 3 种方法截止期错失率的比较,横轴为实时任务利用率.由于 RPDS,CUS 策略调度的系统没有错过截止期的情况,DMR 总为 0,所以图中只显示出了 SEDF 的 DMR 曲线.可以看出,在 SEDF 策略下硬实时任务的 DMR 也总为 0,但随着系统利用率的增加,软实时任务的 DMR 呈明显上升的趋势.

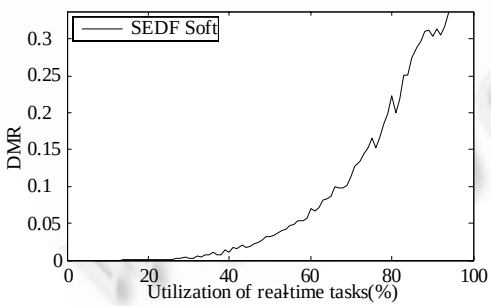

Fig.5 Comparison of DMR under static load

图 5 静态负载下 DMR 比较

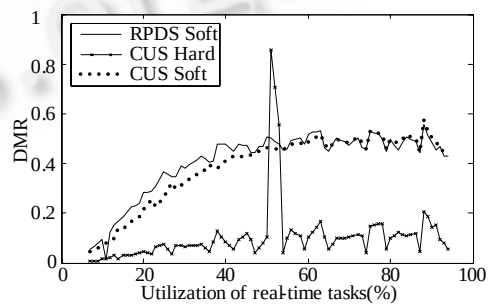

Fig.6 Comparison of DMR under dynamic load

图 6 动态负载下 DMR 比较

图 6 显示了软实时任务时间参数随机变化情况下 RPDS 与 CUS 截止期错失率的比较.可以看出,软实时任务的 DMR 在 RPDS 和 CUS 调度下相差不大,但在 RPDS 下有少许波动.在硬实时任务的 DMR 比较中,RPDS 表现出了更大的优势,没有错过截止期的情况,DMR 总为 0,图中没有显示出来;而 CUS 策略下,硬实时任务的 DMR 通常在 10%左右.

由于 RPDS 会在特定时机强制未完成的硬实时任务切换到非硬实时任务,所以由 RPDS 调度的系统任务切

换次数可能比由 SEDF,CUS 调度的切换次数要多.对此我们用切换率(switch rate),即单位时间内任务切换次数作为标准,进行了实验比较,结果如图 7 所示.图 7 表明,RPDS 确实比另外两种策略产生的任务切换次数要多,但最多并未超过它们的 1.5 倍.并且由于任务切换时间在整个系统运行中所占时间比例很小,所以 RPDS 相对于其他两种策略的任务切换率增量不会对系统正常运行产生影响.

仿真实验表明,RPDS 克服了以往一些方法的局限性,在系统开销增量较小的情况下,为不同实时应用提供了有效的隔离,并降低了实时任务的截止期错失率.

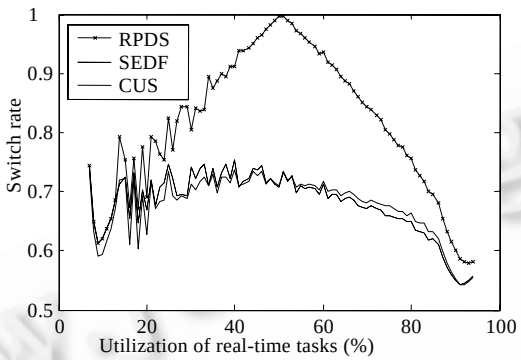

Fig.7 Comparison of task swith rates

图 7 任务切换率比较

5 结束语

本文提出的RPDS算法解决了开放混合实时系统中协同调度多类型时间约束应用的问题.以往的研究或者将各类实时任务简单地分开,赋予硬实时任务绝对的优先权,导致软实时任务的截止期错失率产生了不应有的增加;或者将各类实时任务不加区别地调度,会引发软实时任务影响硬实时任务执行的情况.RPDS 克服了已有方法的不足,根据系统中硬实时任务利用率动态地将处理器时间流分成连续小段,通过强制性地将非硬实时任务插入到每个小段中,达到了将处理器时间按利用率分配给各类任务的目的.仿真实验表明,RPDS 既避免了硬实时任务截止期的错过,又最大程度地降低了软实时任务的截止期错失率.

References:

[1] Deng Z, Liu JWS. Scheduling real-time applications in an open environment. In: Proc. of the 18th IEEE Real-Time Systems Symp. San Francisco: IEEE Computer Society Press, 1997. 308−319.

[2] Spuri M, Buttazzo G. Scheduling aperiodic tasks in dynamic priority systems. Real-Time Systems Journal, 1996,10(2):179−210.

[3] Abeni L, Buttazzo G. Integrating multimedia applications in hard real-time systems. In: Proc. of the 19th IEEE Real-Time Systems Symp. Madrid: IEEE Computer Society Press, 1998. 4−13.

[4] Lipari G, Carpenter J, Baruah S. A framework for achieving inter-application isolation in multiprogrammed, hard real-time environments. In: Proc. of the 21st IEEE Real-Time Systems Symp. Orlando: IEEE Computer Society Press, 2000. 217−226.

[5] Zou Y, Li MS, Wang Q. Analysis for scheduling theory and approach of open real-time system. Journal of Software, 2003,14(1): 83−90 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/83.htm

[6] Caccamo M, Sha L. Aperiodic servers with resource constraints. In: Proc. of the 22nd IEEE Real-Time Systems Symp. London: IEEE Computer Society Press, 2001. 161−170.

[7] Brandt SA, Banachowski S, Lin C, Bisson T. Dynamic integrated scheduling of hard real-time, soft real-time and non-real-time processes. In: Proc. of the 24th IEEE Real-Time Systems Symp. Cancun: IEEE Computer Society Press, 2003. 396−407.

[8] Liu CL, Layland JW. Scheduling algorithms for multiprogramming in a hard real-time environment. Journal of the ACM, 1973, 20(1):46−61.

[9] Liu JWS. Real-Time Systems. Upper Saddle River: Pearson Education, 2000.

[10] Liu D, Hu XS, Lemmon MD, Ling Q. Firm real-rime system scheduling based on a novel QoS constraint. In: Proc. of the 24th IEEE Real-Time Systems Symp. Cancun: IEEE Computer Society Press, 2003. 386−395.

附中文参考文献:

[5] 邹勇,李明树,王青.开放式实时系统的调度理论与方法分析.软件学报,2003,14(1):83−90. http://www.jos.org.cn/1000-9825/14/83.htm

龚育昌(1943－),女,上海人,教授,博士生导师,主要研究领域为操作系统,数据库,算法,超媒体.

陈香兰(1977－),女,博士生,主要研究领域为分布式操作系统.

王立刚(1979－),男,博士生,主要研究领域为操作系统,实时系统.

齐骥(1978－),男,博士生,主要研究领域为可重构系统调度,操作系统.

oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo

第 23 届中国数据库学术会议(NDBC 2006)

征 文 通 知

2006 年 11 月 10-13 日，广州

主办单位：中国计算机学会数据库专业委员会（www.ccf-dbs.org.cn）

承办单位：中山大学（www.zsu.edu.cn）

协办单位：暨南大学，华南理工大学，广东工业大学，华南师范大学，广东计算机学会

会议网址：http://ndbc2006.zsu.edu.cn

重要日期：论文提交截止时间：2006 年 5 月 10 日

论文录用通知时间：2006 年 7 月中旬

排版稿件截止时间：2006 年 7 月下旬

有关会议信息可以访问网站 http://ndbc2006.zsu.edu.cn,也可以与会务组联系 E-mail:ndbc2006@zsu.edu.cn

电话：020-84112137, 020-84110087

传真：020-84112290

通讯地址：中山大学信息学院计算机科学系 NDBC2006 会务组（广州 510275）