文章编号: 0490-6756(2006)03-0544-05

一种基于动态优先级的实时混合任务调度算法

徐文清, 杨红雨
(四川大学计算机学院, 成都 610064)

摘要: 作者对现有多种实时任务调度算法进行研究, 针对复杂实时任务模型, 提出了一种新型的基于动态优先级的混合型实时任务调度算法. 经过模拟测试验证, 算法可以提高硬截止期任务满足截止期的概率, 也可以提高固截止期任务完成数量占总数量的比例.

关键词: 截止期任务; 临界优先级; 服务质量暂时降级; 完成率; 满足率
中图分类号: TP312 文献标识码: A

1 引言

混合型任务集的实时系统中, 按照各种任务对时间的不同要求, 一般分为硬截止期(Hard Deadline)任务、固截止期(Firm Deadline)任务和软截止期(Soft Deadline)任务. 硬截止期任务用于保证系统的正确运行, 而固截止期和软截止期任务用于实现不关键的系统活动. 在该类任务集中, 需要保证硬截止期任务满足截止期(Deadline), 最大化固截止期任务的完成数量, 最小化软截止期任务的平均响应时间[1].

对于实时混合型任务调度, 宜采用基于动态优先级的调度方法, 它主要分为基于服务器的算法和基于空闲时间的算法[1]. 但是由于这两种算法都不适合系统负载较高的情况, 加之目前还缺乏关于它们的真实试验平台, 缺少对这两类算法的综合性能分析与比较; 而且对它们的研究总是基于一些避开实际系统情况的前提和假设, 如要求定期任务的截止期等于周期, 不存在任务的就绪抖动(Release Jitter)与任务同步等[1]. 因而, 如针对实际系统中的调度, 以上两个算法的实现价值不大.

目前已在实验平台试验或在实际系统采用的基于优先级的动态调度算法有很多, 最基本的有: 最短截止期优先 EDF(Earliest Deadline First)[2]、最短处理时间优先 SPTF(Shortest Processing Time First)[2]、最小空闲时间优先 LSF(Least Slack First)和最高价值优先 HVF(Highest Value First)等等. 它们都是只考虑到了任务在某一方面的特征, 如截止期、处理时间、空闲时间、价值等, 而忽略了其它方面的特征, 没有充分考虑任务完成所需要的诸多因素. 因此, 在这些基本调度算法的基础上, 针对具体应用环境的不同, 人们提出了某些更加合适的调度算法, 如 EDV (Earliest Deadline Value)和 VED(Value Earliest Deadline)算法综合了价值和截止期两方面的因素, 设置抢占阈值的方法结合了抢占和非抢占各自的优点[4], 模糊控制反馈调度方法增加考虑了外界不可预测的实时因素等等.

但是, 针对实际中复杂的系统, 以上的算法仍有很多不足: (1)考虑因素有限将不能较好地应变实时系统不可预测的情况; 简单考虑两种因素的算术组合无法真实反映某种因素对任务的特殊重要性; (2)对硬截止期任务不加以特殊考虑无法保证所有的硬截止期任务满足截止期, 甚至阻碍其它固截止期任务和软截止期任务的执行; (3)没有特殊考虑实际系统中执行失败以后再次执行的情况, 失败以后一直按照原来的方法计算优先级, 可能由于执行时机未到, 导致 CPU 时间的浪费.

我们针对影响优先级的诸多因素以及硬截止期任务、固截止期任务和软截止期任务本身的不同特征与它们的执行成败对系统的不同影响程度, 提出了一种新的基于动态优先级的混合型实时调度算法. 经过仿真试验证明, 该算法提高了硬截止期任务满足截止期的概率和固截止期任务完成数量占总数量的比例.

收稿日期: 2005-10-12

2　混合任务动态优先级调度

这种新的基于动态优先级的混合型实时调度算法主要适用于具有以下特征的实时系统中：

(1)所有任务运行在单机系统上.

(2)优先级高的任务可以抢占优先级低的任务.

(3)周期任务都起始于临界时刻.

(4)可能存在某些允许失败以后再次执行的任务.

(5)硬截止期任务可能存在执行时机，且其截止期内必存在执行时机.

由于系统中需要实时更新任务的价值，整个调度系统由优先级计算单元、调度器和采集器组成[3]，如图 1 所示. 其中，采集器负责对影响任务优先级的诸多因素进行实时采集，并根据采集结果实时更新任务的执行价值等参数，使任务优先级的计算和更新与环境的实时变化密切相关；优先级计算单元负责根据采集器采集的结果实时更新所有任务的优先级别，并按优先级顺序排列就绪任务；调度器负责按照一定的全局调度策略，实时调度任务的执行.

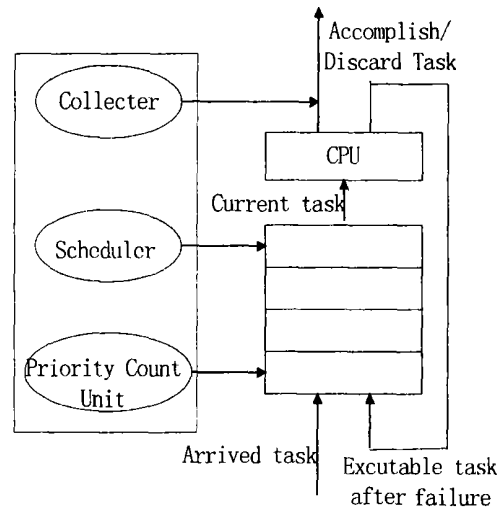

图 1　调度系统结构图

Fig. 1　Architecture of scheduling

2.1　任务模型

定义 $S=\{T_1, T_2, \ldots, T_n\}$ 为任务集合，集合中的任务 T_i 为 SP_i 或 HP_i ($i=1, 2, \cdots, n$)，其中，SP_i 表示软截止期任务或固截止期任务，HP_i 表示硬截止期任务，有：

$$SP_i=(Da_i, Tl_i, Tav_i, Ve_i, Pc_i, P_{\max_i}, P_i, Pv_i, TP_i, Tar_i)$$

其中，Da_i 表示周期任务的绝对截止期限或非周期任务的相对截止期限(Deadline)；Tl_i 表示剩余空闲时间(Slack)；Tav_i 表示平均执行时间；Ve_i 表示执行价值(Value)；它反映任务的执行(包括任务本身执行情况及其执行时的系统环境)对系统整体性能影响程度，由任务的特征决定；Pc_i 表示临界优先级值；$P_{\max_i}$ 表示最大优先级，限定任务被赋予的优先级大小的上限；P_i 表示动态优先级(Priority)，任务运行过程中实时更新的优先级；Pv_i 表示优先级抢占阈值[4]，它设定的大小决定了任务是否允许被抢占及其被抢占的难易程度. 若设定为任务的原始优先级，则属于完全抢占方式；若为无穷大，则属于非抢占方式. 根据需要，它还可以设置成静态或动态类型；TP_i 表示执行周期；Tar_i 表示任务到达时间.

$$HP_i=(Da_i, Tl_i, Tav_i, Ve_i, Pc_i, P_{\max_i}, P_i, Pv_i, Tew_i, \rho_i, Tf_i, Eo_i)$$

其中，Da_i，Tl_i，Tav_i，Ve_i，Pc_i，$P\max_i$，P_i，Pv_i 与 SP_i 中对应参数的含义相同；Tew_i 表示估计最坏执行时间；ρ_i 表示成功率；Tf_i 表示最后一次执行失败时间；Eo_i 表示执行时机，即任务再次执行需要等待的最短时间或者需要满足的最小执行价值.

定义 1　两种不同类型的任务之间的一个优先级门槛称为临界优先级值. 如在固截止期任务和软截止期任务之间设定一个临界优先级，保证固截止期任务的优先级一般比软截止期任务的优先级大，只有当固截止期任务刚刚到达且空闲时间较长而软截止期任务的最小空闲时间接近 0 时，软截止期任务的优先级才可能大于固截止期任务的优先级.

定义 2　若某任务只在某种时刻或环境下执行时成功率较高，而在其它时刻或环境下执行时成功率极低，甚至不可能执行成功，则称该时刻或该环境为任务的执行时机.

此调度算法的性能主要从是否保证硬截止期任务满足截止期,固截止期任务完成数量是否得到最大化,软截止期任务的平均响应时间是否得到最小化这几个角度评估.

2.2 调度流程

通过对以往调度算法研究,结果表明,使用单纯的 EDV 或 HVF 调度方式下,由于对所有任务(包括硬截止期、固截止期和软截止期任务)采用相同的调度策略而错过一些硬截止期任务的调度时机,致使某些硬截止期任务的截止期得不到满足;而 QoS(Quality of Service)降级策略则由于任务失败后其优先级一直降级,致使任务在刚刚失败以后成功率极低的情况下过多执行,而失败一定时间后成功率较高的情况下又因优先级降至过低而得不到充分执行,因此不仅错过了一些硬截止期任务的调度时机,还阻碍了固截止期任务和软截止期任务的执行.

本算法鉴于硬截止期、固截止期和软截止期任务本身的不同特征以及它们的执行成败对系统的不同影响程度,对它们分别采用不同的优先级计算方法.

虽然基于动态优先级的算法相对较好,但是在本任务模型中,如果仅仅以优先级的大小选取任务作为当前任务,效果将不太理想:硬截止期任务的执行时机未到时,会由于优先级最高而浪费 CPU;系统异常繁忙时为保证硬截止期任务的执行,可能会延后其它任务的执行,如果将延后的任务都堆积在队列中,则后续任务都将延后,系统处理将从整体上慢下来.基于这种情况,在实时计算更新任务优先级以确定最大优先级任务之后,本算法还采取了一定的全局调度策略,对硬截止期、固截止期和软截止期任务分别采用不同的全局调度方法.整个调度流程如图 2 所示.

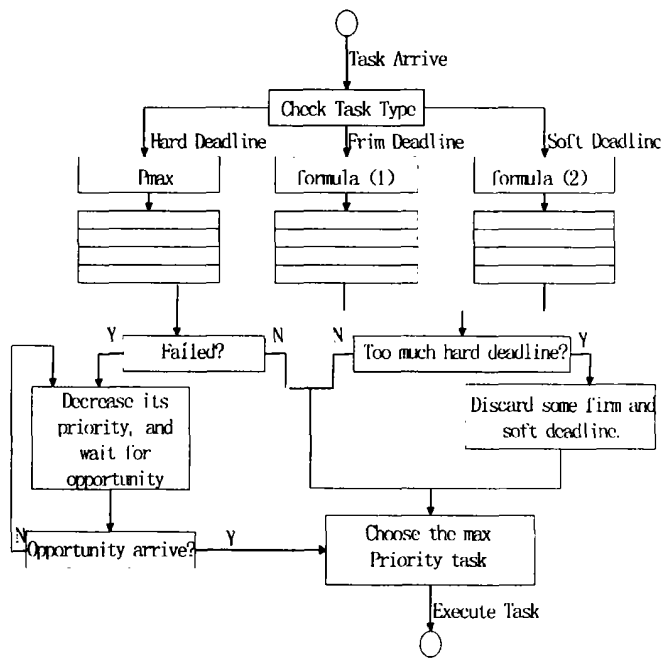

图 2 调度流程图

Fig. 2 Scheduling workflow

2.3 优先级算法

初始化时,对每种类型的任务分别设定最大优先级和临界优先级值.一般,同种任务的最大优先级不低于临界优先级值;硬截止期任务的最大优先级高于所有任务的最大优先级,确保硬截止期任务满足截止期;而固截止期任务的最大优先级高于软截止期任务的最大优先级,保证固截止期任务优于软截止期任务完成;硬截止期任务的临界优先级值约等于软截止期任务的临界优先级值,保证硬截止期任务在执行时机未到时不占用 CPU;固截止期任务的临界优先级值略低于软截止期任务的最大优先级值,保证软截止期任务的响应时间得到最小化.

对于软截止期和固截止期任务,基本上都借鉴第 1 节中提到的调度思想.但是,出于对硬截止期任务的特殊考虑,对所有的任务都限定了最大优先级值 $P\max_i$ 和临界优先级值 Pc_i,因此,需要在其它调度思想的基础上稍加修改,以保证动态优先级的大小界于最大优先级值 $P\max_i$ 和临界优先级值 Pc_i 之间,满足文中提出的任务模型.

对于软截止期任务,采用 LSF(最小空闲时间优先)算法,各个任务的动态优先级 P_i 随着空闲时间 Tl_i 的减少而增加.由于对优先级大小范围的限定,我们按照式(1)对 P_i 进行动态更新.

$$P_i=(P\max_i-Pc_i)/(Da_i-Tav_i)\ *(Da_i-Tav_i-Tl_i)+Pc_i \tag{1}$$

式(1)表示在正常周期情况下,任务周期开始时,优先级被设置为临界优先级 Pc_i,随着 Tl_i 的减少 P_i 线性

增加，直到 Tl_i减小到 0 时，P_i增大到最大优先级$P\max_i$.

对于固截止期任务，将执行价值 Ve_i和空闲时间 Tl_i两种因素结合，任务的动态优先级 P_i由执行价值 Ve_i和空闲时间 Tl_i两种因素共同决定.由于对优先级大小范围的限定，我们按照式(2)对 P_i进行动态更新.

$$P_i = a(Da_i - Tav_i - Tl_i) * \alpha + bVe_i(1-\alpha) \quad (2)$$

式(2)中，a 和 b 的取值保证 P_i在 Tl_i最大时优先级为临界优先级 Pc_i，Tl_i为 0 时，优先级为最大优先级 $P\max_i$；且 a 和 b 的取值保证两个因素所得到的优先级值的大小相当. α 的大小决定执行价值 Ve_i和空闲时间 Tl_i两种因素在优先级计算中所占比重.

对于硬截止期任务，将其优先级设定为最大优先级.

所有任务一旦运行，其优先级便被设定为其优先级抢占阈值 Pv_i，限定任务被抢占的几率，以减少系统内上下文切换的开销.

2.4　全局调度策略

由于软截止期任务执行成败对系统性能没有明显影响；固截止期任务执行失败对系统性能没有明显损害.因此，当系统异常繁忙且硬截止期任务积累较多时，从全局上对它们采用多分辨率模型方法[5]，适当放弃几个不相邻的软截止期或固截止期任务，在不影响系统的全局性能的情况下保证硬截止期任务执行.

对于硬截止期任务，文中借鉴服务质量降级策略的思想，提出了一种服务质量暂时降级策略.

由于我们提出的这种任务模型中，硬截止期任务执行的成败可能受到执行时机 Eo_i的影响，而其截止期内一般都存在至少一个执行时机，在该执行时机可以保证该硬截止期任务的执行.因此，需要在其截止期内，满足执行时机 Eo_i条件下，及时抢占 CPU，保证其在满足执行时机 Eo_i的时间内执行.

在这种模型下，使硬截止期任务的优先级一直保持最大而一直抢占 CPU 固然是保证其满足截止期的最妥当的方法.然而，在该任务模型下，也需要最大化固截止期任务的完成数量，最小化软截止期任务的平均响应时间.因而需要在执行时机未到时，为其它任务让出 CPU 时间.在本文提出的优先级暂时降级策略中，当硬截止期任务的执行时机未到时，将其优先级暂时降低至临界优先级值 Pc_i，同时，不断检查执行时机是否到达.一旦执行时机到达，则立即将其优先级升高到 $P\max_i$，如果当前执行的任务不是硬截止期任务，则该任务将以最高优先级的身份抢占当前任务而投入运行.

由于获取了合适的执行时机，硬截止期任务的执行基本能够保证其满足截止期.而且，由于在执行时机未到时没有盲目占用 CPU 时间，因此，也为其它固截止期任务和软截止期任务让出了更多的 CPU 时间，保证了它们的执行时间.仿真实验证明，这种方法确实可以从整体上能够保证硬截止期任务满足截止期，最大化固截止期任务的完成数量，最小化软截止期任务的平均响应时间.

3　实验与性能分析

为了评估本算法的性能，分别在不同的负载下对本算法以及 HVF、EDV、QoS 降级策略一共四种调度算法分别进行了试验.这里的负载是指按照任务截止期的限制，在单位时间内所需要执行的任务的总时间.如，在 10s 内系统需要执行的所有任务一共需要花费的 CPU 时间为 15s，则认为该系统负载为 1.5.即系统负载大于 1 时，系统的执行成功率必然小于 1.试验中设定了硬截止期、固截止期任务各一个，软截止期任务两个，其中，每个硬截止期任务的执行时间约为 1ms，每个固截止期任务的执行时间约为 2ms，每个软截止期任务的执行时间分别为约 5ms 和约 1.9s.对于每种调度算法，在负载为 0.5，0.8，1.0，1.6，2.0，2.6，3.0，3.5，4.0 的情况下执行该混合任务集.经过多次试验，采集了每种调度算法在每种负载下分别运行 10min 后硬截止期，固截止期，以及软周期任务的执行情况.

试验结果表明，在负载增大的情况下，我们所提出的调度算法的硬截止期任务满足截止期的概率以及固截止期任务的完成数量占总数量的比率都比其它 3 种调度算法高；4 种调度算法的软周期任务平均响

应时间相当.试验结果曲线图如下,其中,图 3 表示硬截止期任务满足截止期的概率,图 4 表示固截止期任务的完成数量占总数量的比率.

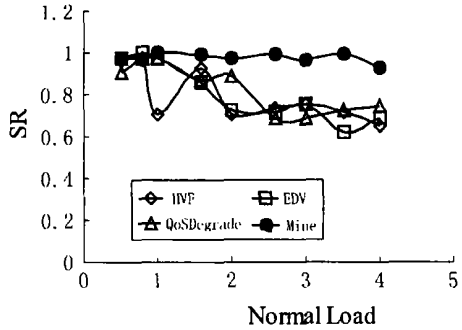

图 3 硬截止期任务满足截止期概率图
Fig. 3 SR of hard deadline tasks

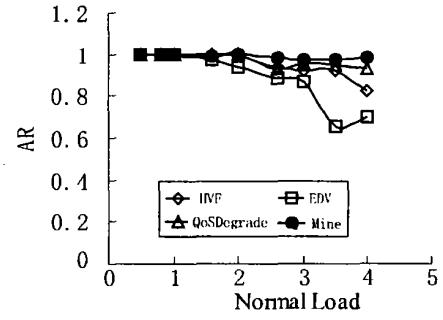

图 4 固截止期任务的完成率图
Fig. 4 AR of firm deadline tasks

4 结束语

我们将多种调度算法结合起来,针对不同类型的任务进行区分考虑,并提出了一种特别针对硬截止期任务的调度算法.该调度算法能够处理单机系统中具有硬截止期、固截止期和软截止期任务的混合型实时任务集;算法特别考虑了具有执行时机特征的硬截止期任务,避免了 CPU 时间不必要的浪费;对于没有执行时机的硬截止期任务,只需认为执行时机一直满足,本算法同样适用.

在基于第 2 节中所述特征的实时环境的仿真试验平台试验中该算法表现出良好的性能,在保证硬截止期任务满足截止期,最大化固截止期任务的完成数量,最小化软截止期任务的平均响应时间这一目标上存在优势,具有一定的应用前景.

参考文献:

[1] 王强, 王宏安, 金宏, 等. 实时系统中的非定期任务调度算法综述[J]. 计算机研究与发展, 2004, 41(3): 385.
[2] 刘怀, 胡继峰. 实时系统的多任务调度[J]. 计算机工程, 2002, 28(3): 43.
[3] Lu C, Stankovic J A, Tao G, et al. Feedback Control Real-Time Scheduling: Framework, Modeling, and Algorithms[J]. Real-time Systems Journal, 2002, 23(1/2): 85.
[4] 金宏, 王强, 王宏安, 等. 基于动态抢占阈值的实时调度[J]. 计算机研究与发展, 2004, 41(3): 393.
[5] 金士尧, 宾雪莲, 杨玉海. 基于多分辨率模型的实时调度方法[J]. 计算机工程与科学, 2004, 26(7): 1.

A Dynamic-Priority-Based Real-Time Scheduling Algorithm for Mixed Task Set

XU Wen-qing, YANG Hong-yu
(College of Computer Science, Sichuan University, Chengdu 610064, China)

Abstract: Aim at complicated real-time model, the paper proposes a new Dynamic-Priority-Based real-time scheduling algorithm for mixed task set. The experimental results show that, the algorithm can improve the probability of satisfaction of deadline for hard deadline tasks; and it also improves the proportion of accomplish of firm deadline.

Key words: deadline task; critical priority; temporarily degrade of service of quality; (AR)accomplish rate; (SR)satisfaction rate