基于有限优先级的动态调度算法

何福贵[1]，王家礼[2]

(1. 西安电子科技大学机电工程学院 西安 710071; 2. 北京工业大学软件学院 北京 朝阳区 100022)

【摘要】实时任务调度是实时系统中的关键问题，实时动态调度是实时调度的主要方面。当实时调度应用于实际的任务系统时，仅能使用有限的优先级数量。实时调度在理论分析时，都假设系统能够识别任意多的优先级。该文提出了在优先级数量有限的条件下的动态调度算法，给出了一个任务系统动态调度所需的最小优先级的数量的算法，并对算法的复杂性进行了分析。

关 键 词 动态调度; 有限优先级数量; 实时任务调度
中图分类号 TP316 文献标识码 A

Dynamic Scheduling Algorithm Based on Limited Number of Priority

HE Fu-gui[1]，WANG Jia-li[2]

(1. College of Mechanism and Electronics, Xidian University Xi'an 710071;
2. School of Software Engineering, Beijing University of Technology Chaoyang Beijing 100022)

Abstract Real-time tasks scheduling is key problem in real-time system, real-time dynamic scheduling is main aspect of real-time scheduling. When real-time scheduling is used in practical system it may just use limited number of priority level. When real-time scheduling theory carry out theoretic analysis, all of them assume that there are unlimited number of priority level. The paper present the algorithm of dynamic real-time scheduling in the condition of limited priority level, and presents the minimum number of priority level that the tasks system schedules dynamically. Finally, the algorithm complexity is analyzed.

Key words dynamic scheduling; limited number of priority level; real-time tasks scheduling

实时系统是一种计算机系统，指计算的正确性不仅依赖于计算的结果，而且依赖于产生结果的时间。实时调度是实时系统中的核心部分，它分为静态调度和动态调度。静态调度算法主要有单调速率(Rate-Monotonic，RM)算法[1]和截止期单调(Deadline Monotonic，DM)算法[2]；动态调度算法主要有截止期最早最优先(Earliest Deadline First，EDF)算法[1]等。动态调度算法是一种在线(On-Line)调度算法，调度可行性判断在任务的执行过程中完成，需要较少的任务信息。静态调度算法是一种离线(Off-Line)调度算法，在运行中有较少的调度运行开销，但它的CPU利用率较低，且灵活性较差。RM算法是优化的静态调度算法，当系统中任务足够多时，CPU的利用率在0.69以下。动态调度算法的调度开销较大，但灵活，CPU的利用率较高。EDF是优化的动态调度算法，它的CPU的利用率接近1。

文献[3]给出了一种任务集合的划分，它由一些网格组成，每个网格代表一个优先级，网格数就是系统能识别的优先级个数，相邻两个网格的比是一个常数，任务的所有周期分布在这个区域内。当这个比值小于2时，这种划分可以有效地工作。

文献[4]提出了一种系统能识别最少优先级的方法，它利用空槽和调度饱和的概念，能够对文献[3]不能进行分组调度的任务集合进行分组调度，讨论的都是以RM算法为内容的静态调度。本文在文献[4]的基础上，分析在动态调度的情况下，任务系统能识别的最少优先级数量和判定方法。

1 无限优先级的EDF调度算法

考虑一个周期任务系统$S(n)$，它由n个可剥夺的相互独立的实时周期任务组成，可表示为：$\tau_1,\tau_2,\cdots,\tau_n$，$\tau_i=(a_i,C_i,T_i,D_i)$，其中$a_i$表示任务的到达时间；$C_i$表示要求的执行时间；$T_i$表示周期；$D_i$表示相对截止期，则$\tau_i$的绝对截止期$d_i=a_i+D_i$。

收稿日期：2005 – 05 – 12
基金项目：武器装备部电子测试技术重点实验室基金资助项目(51487010503dz104)
作者简介：何福贵(1966 –)，男，博士生，主要从事嵌入实时操作系统方面的研究.

本文假设这些任务以绝对截止期不减的次序排列，为了分析方便，$\forall i, T_i = D_i$。

定理 **1**[1] 任务系统$S(n)$使用EDF算法可调度的充分必要条件是：

$$U = \sum_{i=1}^{n} C_i / T_i \leq 1 \tag{1}$$

式中 U表示CPU的利用率。工作负荷$W(t)$是当n个任务在$[1,t]$的时间内同时释放，所有任务要求的处理时间之和，计算方法如下：

$$W(t) = \sum_{i=1}^{n} \left\lceil \frac{t}{T_i} \right\rceil C_i \tag{2}$$

式中 $\lceil\ \rceil$表示向上取整。设M表示所有任务周期的最小公倍数，当$W(M)<M$时，定义任务系统为不饱和，具有$M-W(M)$个空槽；当$W(M)=M$时，定义任务系统为饱和；当$W(M)>M$时，任务系统不可调度。在文献[5]中正式定义$e_{j(n)}$表示在有n个任务的系统中第j个空槽：

$$e_{j(n)} = \text{最小的} t \mid t = j + W_n(t) \tag{3}$$

定理 **2** 如果任务系统$S(u)$中的u个任务在t=0同时到达，且$D_{\min} \geq \sum_{i=1}^{u} C_i$，则任务系统$S(u)$是可调度的，并可使用任意调度算法。其中，$D_{\min}$表示最小的相对截止期。

证明 在一个周期内，$S(n)$中所有任务的全部执行之和就是$\sum_{i=1}^{n} C_i$，也就是在一个周期内至少需要的执行时间之和，而$D_{\min}$为所有任务中最小的截止期，因此不论采用什么调度方法(如：FIFO：先进先出，RAM：随机执行，RR：轮转调度等)，均可以调度。

2 有限优先级的EDF调度算法

如果系统中优先级的数量有限，且小于任务系统中的任务数量，则必须将任务系统中的任务进行分组。假设系统中能够使用的优先级数量为m，则分组的数量应小于m，且每组至少包含一个任务。在进行静态分析时，任务的数量和任务的参数提前给定，在分析时不变化，相对比较简单。而EDF算法在动态情况下进行判定的过程中，任务的数量和参数在发生变化。在动态调度的情况下，任务的到达情况分为两种，一种是任务系统中的任务在某一时刻同时到达，另一种是任务系统中的任务以先后时间的顺序到达。下面就这两种情况分别进行说明。

2.1 任务在某一时刻同时到达

设在某一时刻t，$S(n)$中n个任务同时到达，在t时刻之前没有任务执行，如果在t时刻之前有任务执行可归结到第二种情况。为分析方便，设t=0，那么这些任务的相对截止期就等于绝对截止期。这些任务可分为m组，表示为$G(S(n))=\{Q_1,Q_2,\cdots,Q_m\}$，$|Q_i|$表示$Q_i$组中任务的数量，则$\sum_{i=1}^{m}|Q_i| = n$，且$\forall i \in S(n), \exists! Q_h \mid \tau_i \in Q_h$，其中，$\exists!$表示存在唯一的一个。设$\tau_i^j, C_i^j, D_i^j$分别表示第$j$组的第$i$个任务、执行要求和相对截止期。$\bar{Q}(j)$表示优先级高于第$j$组任务的集合，明显地，设$C_1^2 = C_{|Q_1|+1}, D_1^2 = D_{|Q_1|+1}, \bar{Q}(j) = \bigcup_{g=1}^{j-1} Q_g$。$C^j$表示属于第$j$组中的所有任务执行一次要求时间之和，则$C^j = \sum_{h=1}^{|Q_j|} C_h^j$。

定理 **3** 给定任务集合$S(n)$和任务组$G(S(n))$，如果下列条件成立：

$$\forall j = 1,2,\cdots,m, D_1^j \geq \min t \mid t = C^j + W_{|\bar{Q}(j)|}(t) = e_{C^j(\bar{Q}(j))} \tag{4}$$

则$G(S(n))$可使用EDF调度，组内的任务可使用任意调度算法调度。

证明 因为$S(n)$的所有任务在t=0时同时到达，D_1^j为第j组最小的截止期，$W_{|\bar{Q}(j)|}(t)$表示优先级高于第j组的所有任务的执行要求，C^j表示第j组中的所有任务的执行要求，即D_1^j大于等于任务系统中的第C^j个空槽，根据定理2，则任务系统是可调度的。

下面的定理说明在有限优先级的情况下，任务系统$S(n)$可调度所需要的最少的优先级数量。

定理 **4** 给定任务系统$S(n)$，使用EDF可调度，如果每一组都是饱和的，那么这个划分$G(S(n))$为最小数量的分组。

证明 设$G(S(n))=\{Q_1,Q_2,\cdots,Q_m\}$，有$m$个分组，每一个分组是饱和的。假设存在一个更小的分组，那么其中一个分组中至少有一个任务要合并到其他的相邻分组中去，因为它的相邻的分组是饱和的，因此合并后是不可调度的，所以m是最小的分组。下面举例说明：

例1 有5个任务$S(5)$，在时刻t=0同时到达，任务的属性如表1所示，$e_{C_i(i-1)}$的计算也在表中。

从表中可以看出，$D_1^1 = D_1 = e_{1(2)}, D_1^2 = D_4 = e_{1(4)}$，所以可获得分组$(\tau_1,\tau_2,\tau_3),(\tau_4,\tau_5)$。因此，调度的最小分组数量为2，即这些任务所需的最小的优先级数量为2。

表**1**　任务分组计算

i	C_i	D_i	$e_{C_i(i-1)}$
1	1	3	1
2	1	5	2
3	1	7	3
4	1	9	5
5	1	9	9

2.2　任务系统中的任务以先后时间的顺序到达

任务系统$S(n)$中任务在某一时刻t到达了一个新任务，在t时刻之前有k个任务到达，这k个任务是可调度的。但是在t时刻活动的任务有j个，$j \le k$。活动的任务定义为：τ_i是活动的，那么在t时刻τ_i的剩余执行时间大于0且$t \le d_i$。在时刻t到达了一个新的任务，这些任务是EDF可调度的。

在t 时刻活动的任务$S(j)$为$(\tau_i,\tau_{i+1},\cdots,\tau_{i+j})$，它们的绝对截止期为$(d_i,d_{i+1},\cdots,d_{i+j})$，其中，$d_i = a_i + D_i, a_i$表示任务$\tau_i$在$t$时刻之前最近一次的到达时间。在$t$时刻这些任务的剩余计算时间为$(r_i,r_{i+1},\cdots,r_{i+j})$，设在$t$时刻到达的任务为$\tau_s$，则$d_s = t+D_s$。

定理 **5**　如果新到达的任务τ_s，对于$S(j)$有$d_{i+l-1} \le d_s \le d_{i+l}, 1 \le l \le j$，这些任务由EDF判定可调度，且$C_s + r_{i+l} \le d_s$，则$\tau_s$的优先级等于$\tau_{i+l}$的优先级(即$\tau_s$和$\tau_{i+l}$可分为一组)，且任务系统可调度。

证明　因为新到达的任务与原来的任务使用EDF是可调度的，可以假设成具有执行时间r_{i+l}任务τ_{i+l}与τ_s同时到达，根据定理2，任务系统可调度。

对于$S(j)$，即t时刻活动的任务$(\tau_i,\tau_{i+1},\cdots,\tau_{i+j})$，与新到达的任务$\tau_s$，以$t$为所有这些任务的到达时间；以$S(j)$的剩余执行时间$(r_i,r_{i+1},\cdots,r_{i+j})$作为执行时间以及$\tau_s$的执行时间$C_s$；以$(d_i - t, d_{i+1} - t,\cdots, d_{i+j} - t)$作为$j$个任务的相对截止期，以及$\tau_s$的相对截止期$D_s$；以这些属性构成$j$+1个新任务且它们以绝对截止期不减的次序排列，构成一个新的任务集合$S'(j+1)$。

定理 **6**　对于任务集合$S'(j+1)$，可以应用定理4进行最小数量分组的划分。

证明　由定理2和定理4可得出。

推论 1　在任务系统执行过程中的任意时刻，都可进行优先级最小分组的判定。

证明　由定理6直接可得。

3　算法的复杂性分析

对于任务系统$S(n)$，在每一个时间间隔$[0,D_i]$，此算法共计算D_i次，设$D_{\max}$为最大的相对截止期，则最多计算$D_{\max}$次。因为系统同时活动的任务最多为n个，所以算法的复杂度为$O(nD_{\max})$。这个算法实际上是计算任务系统的空槽，根据式(3)，计算是从$W_n(t)+1$开始的，前面$W_n(t)$个时间单位不计算，所以实际的计算量远小于$nD_{\max}$。减少计算量的另一种方法是从第n个任务计算开始，按任务排列的倒序计算，按照表1，计算只有两次，第五个任务和第三个任务就可以进行判断了。按照排列组合的理论，把n个任务中分成m组的组合有$[(n-1)!]/[(m-1)!\times(n-m)!]$，算法复杂度明显减少了。

4　结　论

对于实时周期独立的可剥夺的任务集合，它们执行在一个单处理器上，如果运行在实际的系统中，一般都有优先级数量的限制。现有的文献描述的是在静态调度使用RM算法的情况下，任务分组和任务最小分组的方法。本文提出在使用EDF动态调度的情况下，任务分组和任务最小分组的方法。当然在动态的情况下增加了系统的开销，如需要保存任务的剩余执行时间。但是动态调度比静态调度的CPU利用率高，任务的数量可变化，它的适应性更强。

参　考　文　献

[1] LIU C L, LAYLAND J M. Scheduling algorithms for multiprogramming in a hard real-time environment[J]. Journal of the ACM, 1973, 20(1): 46-61.

[2] LEUNG J, WHITEHEAD J. On the complexity of fixed-priority scheduling of periodic, real-time tasks[J]. Performance Evaluation, 1982, 2: 237-250.

[3] LEHOCZKY J P, SHA L. Performance of real-time bus scheduling algorithms[J]. ACM Performance Evaluation Review, 1986, 14(1): 44-53.

[4] OROZCO J, CAYSSIAL R, SANTOS J, et al. On the minimum number of priority levels required for the rate monotonic scheduling of real-time systems[C]//Proc 10th Euromicro Workshop on Real-Time Systems. Berlin: Wip Session, 1988: 19-21.

[5] SANTOS J. Priority and protocols in real-time LANs[J]. Computer Communications, 1991, 144(9): 507- 514.

编　辑　漆　蓉