ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.7, July 2006, pp.1641−1649
DOI: 10.1360/jos171641

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

基于逻辑“或”约束优化的实时系统设计*

刘军祥[1,2+]，王永吉[1,3]，王 源[1,2]，邢建生[1,2]，曾海涛[1,2]

[1](中国科学院 软件研究所 互联网软件技术实验室,北京 100080)

[2](中国科学院 研究生院,北京 100049)

[3](计算机科学重点实验室(中国科学院 软件研究所),北京 100080)

Real-Time System Design Based on Logic OR Constrained Optimization

LIU Jun-Xiang[1,2+], WANG Yong-Ji[1,3], WANG Yuan[1,2], XING Jian-Sheng[1,2], ZENG Hai-Tao[1,2]

[1](Laboratory for Internet Software Technologies, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

[2](Graduate School, The Chinese Academy of Sciences, Beijing 100049, China)

[3](Laboratory for Computer Science (Institute of Software, The Chinese Academy of Sciences), Beijing 100080, China)

+ Corresponding author: Phn: +86-10-62561197 ext 1003, E-mail: liujunxiang@itechs.iscas.ac.cn

Liu JX, Wang YJ, Wang Y, Xing JS, Zeng HT. Real-Time system design based on logic OR constrained optimization. *Journal of Software*, 2006,17(7):1641−1649. http://www.jos.org. cn/1000-9825/17/1641.htm

Abstract: The logic relationship among the equality and inequality constraints in a standard constrained optimization problem (SCOP) is the logical AND. Various efficient, convergent and robust algorithms have been developed for such a SCOP. However, a more general constrained optimization problem (GCOP) with not only logic AND but also OR relationships exists in many practical applications. In order to solve such a generalized problem, a new mathematical transformations which can transfer a set of inequalities with logic OR into inequalities with logic AND relationships is developed. This transformation provides a necessary and sufficient condition which enables us to formulate real-time system design as a mixed Boolean-integer programming problem. A Branch and Bound Algorithm is applied to find the optimal solution. Experimental results have been presented to show its merits.

Key words: SCOP (standard constrained optimization problem); inequality constraint; branch and bound algorithm; mixed Boolean-integer programming (MBP); RM (rate monotonic)

摘 要: 标准约束优化问题的等式或不等式约束之间是逻辑“与”关系,目前已经有很多高效、收敛的优化算法.但是,在实际应用中有很多更一般的约束优化问题,其等式或不等式约束之间不仅包含逻辑“与”关系,而且还包含逻辑“或”关系,现有的针对标准约束优化问题的各种算法不再适用.给出一种新的数学变换方法,把具有逻辑“或”关系的不等式约束转换为一组具有逻辑“与”关系的不等式,并应用到实时单调速率调度算法的可调度性判

* Supported by the National Natural Science Foundation of China under Grant No.60373053 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2003AA1Z2220 (国家高技术研究发展计划(863)); the Hundred Talents of the Chinese Academy of Sciences (中国科学院“百人计划”); the Chinese Academy of Sciences and the Royal Society of the United Kingdom under Grant Nos.20030389, 20032006 (中国科学院与英国皇家学会联合资助)

Received 2005-09-06; Accepted 2005-11-08

定充要条件中,把实时系统设计表示成混合布尔型整数规划问题,利用经典的分支定界法求解.实验部分指出了各种方法的优缺点.

关键词: 标准约束优化问题;不等式约束;分支定界法;混合布尔型整数规划;单调速率(RM)
中图法分类号: TP391 文献标识码: A

实时系统研究包括多个方面,如形式化验证、资源管理、能耗、QoS 支持等.1973 年,Liu 和 Layland[1]提出单调速率(rate monotonic,简称 RM)调度算法,一个最根本的问题是可调度性判定问题,即给定一组周期性任务集 $\tau=\{\tau_1,\ldots,\tau_i,\ldots,\tau_n\}(i=1,\ldots,n)$,其中每个任务由 $\tau_i=(C_i,T_i)$表示;T_i 是任务 τ_i 的执行周期;C_i 是执行时间,寻找一个高效的方法验证任务集是否可调度.目前已经发表了大量关于 RM 调度算法在理想模型及各种扩展情况下可调度性判定的文章,如最小 CPU 利用率上界[1,2]、多项式判定方法[3]、充分必要调度条件[4]、ISTA[5]等,详细描述可参见文献[6,7].

可调度性判定问题的研究都基于一个共同的假设:各任务参数是固定的.也就是说,在作可调度性判定之前,必须知道任务的参数,如周期、执行时间.从可调度性判定问题可以引出两个新问题:1) 如果给定的任务集 τ 不可调度,那么该如何调整任务参数(C_i,T_i),以使新任务集可调度;2) 能否在一定的区间内调整任务参数从而找到一个可调度的任务集,并且最大化系统的性能指标,如 CPU 利用率.

上面两个问题可以表述为本文的实时系统优化设计问题:给定周期性任务集 $\tau=\{\tau_1,\ldots,\tau_n\}$,假定任务周期 T_i 已知,任务执行时间 C_i 在区间 $[C_i^{\min},C_i^{\max}]$ 内,在 RM 可调度性约束条件下找到一组 C_i,使得系统某一性能指标最优.实时系统设计包括多个方面的内容,如调度器和控制器设计,本文仅指对任务执行时间的设计.我们假定控制任务的采样周期 T_i 在设计控制器时已得到[8].

这个问题对实时系统的研究具有重要意义.目前,进程 QoS 在多媒体编解码及传输、实时数据库、工程计算等领域被广泛提出[9−11],任务在不同的 QoS 级别需要不同的执行时间.本文工作一方面有益于进程 QoS 调节,另一方面可以指导如何进行过载处理.在 Liu 和 Layland[1]理想任务模型的基础上,一些研究者针对具体应用和需求,提出了相应的扩展模型,如非精确计算[12,13]和 IRIS[14]模型.在这些模型中,任务执行时间被分成两部分:必须执行部分和选择执行部分.前者必须在任务截止期之前完成以保证最低的可接受的质量;而后者在必须执行部分执行完成之后截止期之前执行,并可在任意时刻停止.执行时间越多,计算结果越精确、效用越大.通过调节任务的执行时间,可以有效地进行过载处理,使系统性能平稳下降.文献[12−15]给出了根据一定的最优指标(如最大平均收益、最小错误数)计算最佳任务执行时间的方法;Alvarez 等人[16]提出了一种避免系统过载并且最大化 CPU 资源利用率的计算任务执行时间的近似算法;张尧学等人[17]提出利用反馈机制调整任务执行时间来进行过载处理.但是,这些工作不支持 RM 调度算法下对任意任务集进行求解.

本文工作主要应用于实时控制系统的设计.数字控制系统的设计,既要考虑系统的控制性能,又要考虑系统的可调度性.任务执行时间与整个系统的可调度性密切相关,本文工作对系统设计具有重要的指导意义:一方面可根据任务执行时间设计高效算法.如数字控制系统一旦工况发生变化,计算机需要执行自动寻优程序,寻找合适的控制参数,保持系统的性能处于良好的状态.不同的搜索算法,所需的执行时间不一样,相应地,寻优结果也不一样;另一方面,可以选择合适的硬件,使得硬件资源被充分利用,以节约成本.

通常,有两种方式可以用来描述实时系统优化设计问题:一种是利用 CPU 利用率最小上界进行设计,把实时系统设计转化为一个标准的约束规划问题.例如,对 Liu 和 Layland 提出的最小 CPU 利用率上界,其可调度性约束条件为 $g(C_1,\ldots,C_n)\le n(2^{1/n}-1)$,求解这样一个约束规划问题的优点是算法快速、简单,但许多可行解被充分而非必要的可调度性约束条件排除,不能保证最大的 CPU 利用率;另一种利用 RM 可调度的充分必要条件,RM 算法的充分必要条件的特点是:它由一组限制条件间既具有逻辑“与”关系又具有逻辑“或”关系的不等式组成,此时,实时系统的设计不能再利用标准的约束规划问题求解,而需要寻找高效求解具有逻辑“或”关系的优化算法.

在文献[18]中,Wang 和 Lane 用不等式表示智能机器人路径规划中的障碍物,首次提出了一个更一般的优化问题(generalized constrained optimization problem,简称 GCOP),即约束间的逻辑关系不仅含有“与”,而且含有

“或”关系.他们给出了一种数学变换,把具有逻辑“或”关系的约束条件变换为一个不等式,从而可以利用经典的非线性优化算法求解问题.该文提出的方法值得我们借鉴.

本文从第 2 种方式出发,在我们以前对 RM 可调度性判定研究成果[5]的基础上,解决了如何寻找一个最优的可调度实时系统这一实际问题.具体地,本文将实时系统任务执行时间的优化设计问题表示为一个 GCOP,然后给出一种新的数学方法,将不等式间的逻辑“或”关系消去,把实时系统设计表示成混合布尔型整数规划问题,最后利用分支定界法求解.文献[5]的研究成果被集成到了具体的算法中,以对算法性能进行改进.

本文贡献主要有 3 个方面:

- 把实时系统回答“是”与“非”的可调度性判定问题转换为求最优解的最优化模式.一般地,一个优化问题可以看作是其所有实例的集合,即所有 C_i 组合的集合.实时系统判定问题是本文所研究问题的一个特例,它假设 C_i 固定,即 $[C_i^{\min},C_i^{\max}]$,仅仅是所有实例集合中的一个实例.从这个意义上来说,本文的研究更具有一般性;
- 在实时系统理论研究方法的基础上,第一次把优化理论应用到实时系统可调度充要条件上,给出实际的求解方法,并从优化角度对其研究意义进行了具体的解释;
- 给出了一种新的、不同于文献[18]中的数学方法,把逻辑“或”关系转化为逻辑“与”关系,并探讨了各方法的优缺点.

本文第 1 节简要介绍本文工作的研究基础.第 2 节将实时系统任务执行时间优化设计问题表示为一个 GCOP 问题.第 3 节给出一种新的数学方法,将具有逻辑“或”关系的不等式组表示为一组等价的具有逻辑“与”关系的不等式.第 4 节利用数学变换把第 2 节的 GCOP 表示为一个 SCOP.第 5 节给出求解问题的算法描述.第 6 节给出仿真实验,指出各种方法的优缺点.最后给出结论.

1 研究基础

1.1 约束优化理论

标准优化问题(standard constrained optimization problem,简称 SCOP)研究如何找到一个满足等式和不等式约束的可行解的同时使得目标函数值最大或最小,在工程领域有着广泛的应用,如最优控制器设计、曲线拟合、非线性方程求解、机器人路径规划等[18].SCOP 的具体形式可以描述为

$$\begin{aligned} &\min && f(\boldsymbol{x})\\ &\text{s.t.} && g_i(\boldsymbol{x})\le 0,i=1,2,\ldots,m \end{aligned} \tag{1}$$

其中 $\boldsymbol{x}$ 是 n 维变量向量;$f(\boldsymbol{x})$称为目标函数;$g_i(\boldsymbol{x})$称为约束函数.应该注意到,所有约束条件间都是逻辑“与”的关系.也就是说,可行域是由不等式间的“与”关系确定的.

以符号“∨”表示逻辑运算“或”,GCOP 可以描述为

$$\begin{aligned} &\min && f(\boldsymbol{x})\\ &\text{s.t.} && g_{i,1}(\boldsymbol{x})\le 0\vee\ldots\vee g_{i,k}(\boldsymbol{x})\le 0,i=1,2,\ldots,m \end{aligned} \tag{2}$$

可以看出,SCOP 是 GCOP 当 $k_i=1(i=1,2,\ldots,m)$时的一个特例.

1.2 RM可调度性判定条件

这里给出 RM 算法的可调度性判定条件,它是本文工作的基础.

Liu 和 Layland[1]给出了一个 RM 算法的可调度性判定条件:给定任务集 $\tau=\{\tau_1,\ldots,\tau_n\}$,如果这 n 个任务的 CPU 利用率 $U\le n(2^{1/n}-1)$,则该任务集 τ 用 RM 算法是可调度的.$n(2^{1/n}-1)$表示 n 个任务的 CPU 利用率最小上界,记为 LLB.这个最小上界从 $0.83(n\to 2)$单调下降到 $0.693(n\to\infty)$.然而,这个最小上界来自于对最坏情况的估计,带有很大的悲观性.后来,Bini 等人[2]给出了一个双曲线的 CPU 利用率最小上界 HB.一个含有 n 个周期性任务的任务集是 RM 可调度的,如果$(1+U_1)(1+U_2)\ldots(1+U_n)\le 2$.$HB$ 优于 LLB,从而扩大了 RM 可调度性判定的范围.

Lehoczky 等人[4]对固定优先级调度算法的特征进行了更精确的研究,提出 RM 可调度性判定的充要条件.

定理 **1**. 对任务集$\tau=\{\tau_1,\ldots,\tau_n\}$,有

(1) 任务τ_i是可调度的当且仅当

$$L_i=\min_{t\in S_i}\frac{\sum_{j=1}^{i}\left\lceil\frac{t}{T_j}\right\rceil C_j}{t}\le 1 \tag{3}$$

这里,$S_i=\{rT_j|j=1,\ldots,i,r=1,\ldots,[T_i/T_j]\}$.

(2) 整个任务集是可调度的当且仅当

$$\max_{i=1,\ldots,n} L_i\le 1 \tag{4}$$

在式(3)中,各不等式的关系是一种逻辑“或”的关系,而在式(4)中,各不等式的关系是一种逻辑“与”的关系,如果用符号“∨”来表示逻辑“或”操作,用符号“∧”表示逻辑“与”操作,定理 1 可以表示为

$$\max_{i=1,\ldots,n}\min_{t\in S_i}\left(\sum_{j=1}^{i}\left\lceil t/T_j\right\rceil C_j/t\right)\le 1\Leftrightarrow \bigwedge_{i=1,\ldots,n}\min_{t\in S_i}\left(\sum_{j=1}^{i}\left\lceil t/T_j\right\rceil C_j/t\right)\le 1\Leftrightarrow \bigwedge_{i=1,\ldots,n}\bigvee_{t\in S_i}\left(\sum_{j=1}^{i}\left\lceil t/T_j\right\rceil C_j\le t\right) \tag{5}$$

可以看出,RM 算法的充分必要条件的特点是:它由一组限制条件间的逻辑关系为“或”的不等式组成.

2 数学模型

2.1 GCOP模型

假设系统设计的目标是使得 CPU 资源利用率最大化,那么系统设计问题可以表示为如下优化问题:

$$\max \quad f=\sum_{i=1}^{n}C_i/T_i \tag{6}$$

$$\text{s.t.} \quad \begin{cases}\forall i, i=1,2,\ldots,n\\ \bigvee\limits_{t\in S_i}\left(\sum\limits_{j=1}^{i}\left\lceil t/T_j\right\rceil C_j-t\right)\le 0\\ C_i^{\min}\le C_i\le C_i^{\max}\end{cases} \tag{7}$$

式(6)是目标函数,式(7)是可调度判定充要条件和变量C_i的取值范围.上述优化问题是一个 GCOP,不能直接利用现有求解 SCOP 的方法求解,必须找到新的数学方法来处理.本文将利用两种等价变化将其转化为 SCOP 并直接求解.

2.2 数学变换

下面先给出文献[18]中的数学变换方法,然后给出一种新的等价变换,以消去逻辑“或”关系.

2.2.1 非线性数学变换

Wang 和 Lane[18]给出并证明了如下定理:

定理 **2**. $h_1\le 0\vee h_2\le 0\vee\ldots\vee h_m\le 0$ 可以用 $\Delta v-\sum_{j=1}^{m}\left(\sqrt{h_j^2}-h_j\right)\le 0$ 求解,其中Δv 是一个很小的正常数,并且$\Delta v\to 0$.

该数学变换的优点是将多个具有逻辑关系“或”的不等式转化为一个非线性不等式,减少了优化算法搜索过程中要考虑的不等式约束判定的个数;缺点是转换前如果 $h_i\le 0(i=1,2,\ldots,m)$是线性约束,经过数学变换后则成为一个非线性约束.一般来说,求解具有非线性约束的优化问题比求解线性规划问题要困难得多,而且不像线性规划有单纯形法这一通用算法,非线性约束优化目前还没有适用于各种问题的一般算法.而且,现有的方法都只能找到局部最优解.

2.2.2 混合布尔型整数数学变换

由 $h_1\le 0\vee h_2\le 0\vee\ldots\vee h_m\le 0$,我们可以假设不等式 $h_i\le 0,i=1,2,\ldots,m$ 之间的关系是互斥的,它不会影响问题的求解.为保证这 m 个约束条件中只有一个起作用,我们引入 m 个 0-1 变量 $y_i(i=1,2,\ldots,m)$和一个充分大的正常数 M.下面的约束条件

$$h_i-y_iM\le 0,i=1,2,\ldots,m \tag{8}$$

$$y_1+y_2+\ldots+y_m=m-1 \tag{9}$$

即符合上面的要求.因为式(9)中 m 个 y_i 中只有一个能取 0 值,设 $y_{i^*}=0$,代入式(8),只有 $i=i^*$的约束条件起作用,其余的式子因 M 是一个充分大的正常数而自然成立.

综合上面的分析,我们可以得到如下定理:

定理 **3**. 设命题为:存在一个可行解$(y_1,y_2,\ldots,y_m)$使得 $h_i-y_iM\le 0(i=1,2,\ldots,m)$, $\sum_{i=1}^{m}y_i=m-1$,其中$\forall i,y_i\in\{0,1\}$.逻辑“或”约束 $h_1\le 0\vee h_2\le 0\vee\ldots\vee h_m\le 0$ 与该命题等价.

该数学变换将多个具有逻辑关系“或”的不等式转化为相互之间是逻辑关系“与”的多个不等式.与非线性数学变换相比,优点是转换后约束条件仍是线性的,取线性目标函数,而解线性规划问题有很多多项式算法,而且能够找到全局最优解;缺点是引入了新的布尔变量,将原有连续的约束函数变为既含有连续变量又含有布尔变量的约束条件.

2.3 SCOP模型

利用上面的数学变换,RM 算法的可调度性充分必要条件可以转化为仅包含逻辑关系“与”的不等式,从而可以利用 SCOP 设计实时系统各任务的执行时间,使得系统的性能指标,如 CPU 利用率最大.

根据非线性数学变换,不难得到如下非线性规划模型来求解任务的执行时间:

$$\max \quad f=\zeta\sum_{i=1}^{n}C_i/T_i \tag{10}$$

$$\text{s.t.}\quad \begin{cases}\forall i,i=1,2,\ldots,n\\ \Delta v-\sum_{j=1}^{k_i}\left(\sqrt{\left(\sum_{m=1}^{i}\lceil S_{ij}/T_m\rceil C_m-S_{ij}\right)^2}-\left(\sum_{m=1}^{i}\lceil S_{ij}/T_m\rceil C_m-S_{ij}\right)\right)\le 0\\ C_i^{\min}\le C_i\le C_i^{\max}\end{cases} \tag{11}$$

其中ζ是一个放大因子,通常取$\zeta=10^4$;Δv 是一个无限小的正数;k_i 表示集合 S_i 中元素的个数;S_{ij} 表示集合 S_i 中第 j 个元素.式(11)是变换了形式后的可调度性约束条件以及任务 τ_i 执行时间的有效范围.

相应地,根据定理 3,可以得到混合布尔型整数规划模型来求解任务的执行时间.

$$\max \quad f=\sum_{i=1}^{n}C_i/T_i \tag{12}$$

$$\text{s.t.}\quad \begin{cases}\forall i,i=1,2,\ldots,n\\ \sum_{j=1}^{i}\lceil S_{ik}/T_j\rceil C_j-S_{ik}-y_{ik}M\le 0,k=1,2,\ldots,k_i\\ C_i^{\min}\le C_i\le C_i^{\max}\\ \sum_{j=1}^{m_i}y_{ij}=m_i-1\\ \forall j=1,\ldots,m_i,y_{ij}=0\text{或}1\end{cases} \tag{13}$$

其中 y_{ij} 是新引入的布尔变量;k_i 表示集合 S_i 中元素的个数;S_{ik} 表示集合 S_i 中第 k 个元素;其他符号同上.

可以看出 GCOP(式(6))利用两个等价的数学变换,分别转换成为 SCOP(式(10)和式(12)),对问题(式(10)),可以直接利用现有的求解非线性规划经典的优化方法,如序列二次规划,来解决实时系统任务执行时间设计的问题,其设计结果还依赖于参数ξ,Δv 的取值.对混合布尔型整数规划问题(式(12)),可以利用分支定界法求最优解.

3 算法描述及复杂度分析

分支定界法一般仅检查可行的布尔整数组合的一部分就能得出最优解,是求解混合布尔型整数规划的常用方法,其基本思想是:设 A 是最大化的混合布尔型整数规划问题(式(12)),与它对应的线性规划问题为 B.从解问题 B 开始,若其最优解不符合 A 的整数条件,那么,B 的最优目标函数值必是 A 的最优目标函数 f^*的上界,记为 uf;而 A 的任意可行解的目标函数值将是 f^*的一个下界 lf.分支定界法将 B 的可行域分成子区域(称为分支)的方法,逐步减小 uf 和增大 lf,最终得到 f^*.

为减少计算量,分支定界法的一个重要步骤就是比较与剪枝,若分支的最优目标函数值小于 lf,则剪掉这一枝.此外,考虑实际实时系统可调度性判定约束条件的特点,可以得出更进一步的结论.

文献[5]给出了两个对剪枝极其有用的定理.

定理 **4**. 如果条件 $\sum_{j=1}^{i}\lceil t/T_j\rceil C_j \le t$ 满足,使得任务 τ_i 可调度,并且 $t\le T_j(j<i)$,那么任务 τ_j 必定可以调度.

定理 **5**. 给定任务集 $\tau=\{\tau_1,\ldots,\tau_n\}$,如果 $T_n\le 2T_1$,并且 τ_n 是 RM 可调度的,那么任务集 τ 是可调度的.

证明请参见文献[5].

根据不符合整数条件的变量 y_{ij} 进行分支(每个分支都有一个 y_{ij} 及与之相应的 S_{ij} 关联),有如下 3 个结论:

推论 **1**. 由 *LLB* 有:问题 *A* 目标函数值的一个下界是 $lf=n(2^{1/n}-1)$,凡是分支的最优目标函数值小于 $n(2^{1/n}-1)$,则剪掉该分支.

推论 **2**. 如果对问题 *B* 的后继规划问题 B_i 有 $S_{ij}\le 2T_1(i=1,2,\ldots,n)$,那么 B_i 的所有分支都可以被剪掉.

推论 **3**. 如果子问题 B_i 的一个分支 B_{i-1} 有 $S_{(i-1)j}\le T_i$,那么该分支是多余的,可以被剪掉.

我们用例 1 来说明算法过程.

例 1:假设任务集属性如表 1 所示,T_i 是与 y_{ij},S_{ij} 相对应的任务 τ_i 的周期.

Table 1 Task set characteristics

表 **1** 任务集属性

Task	T_i	C_i	S_i
τ_1	100	[20, 60]	100
τ_2	150	[20, 75]	100,150
τ_3	210	[30, 100]	100,150,200,210
τ_4	400	[30, 150]	100,150,200,210,300,400

分支定界法描述如下:

① 解问题 *B*,得到 uf=1.1,同时由结论 1,lf=0.693;

② 分支,从 $i=n(n=4)$开始,根据不符合布尔整数条件的变量 y_{ij} 构造约束条件 y_{ij}=0,求后继规划问题;

③ 定界,求出每一个分支的结果,与其他问题的解进行比较,找出最优目标函数值最大者作为新的上界 uf,从已符合整数条件的各分支中,找出目标函数值最大者作为新的下界 lf;

④ 比较与剪枝,在图 1 中,子规划问题 *B*1,*B*2 和 *B*3 因为 $S_{4i}\le 2T_1(i=1,2,3)$,由推论 2,其所有分支被剪掉;子规划问题 *B*4,因为 f=0.7893<lf=0.8512,所有的分支被剪掉;解子问题 *B*5 时得到 uf=0.9179,解子问题 *B*52 时得到 lf=0.9179,$lf=uf$,得到该分支的最优解,停止继续分支;

⑤ 分支结果大于 lf 且有不符合整数条件的变量 y_{ij},如解子问题 *B*6 时得到 uf=0.9762,回到②.

利用上面分支定界法求混合布尔型整数规划问题(式(12))时可以发现,问题(式(12))一共有 $k_nk_{n-1}\ldots k_2$ 个可行的布尔整数组合数,设 $\lceil T_n/T_1\rceil=m$, 最坏情况下, $k_2=\lceil T_2/T_1\rceil=m, k_3=\lceil T_3/T_1\rceil+\lceil T_3/T_2\rceil=2m,\ldots,k_n=(n-1)m$, 不同的组合方案有 $(n-1)!m^{n-1}$ 种.在最坏情况下,算法计算时间随任务集规模呈指数增长.但是,实际算法要比上面分析的结果高效许多.从图 1 可以看出,求解只需要检查 9 个可行的布尔整数组合,而不是检查 6×4×2=48 个.

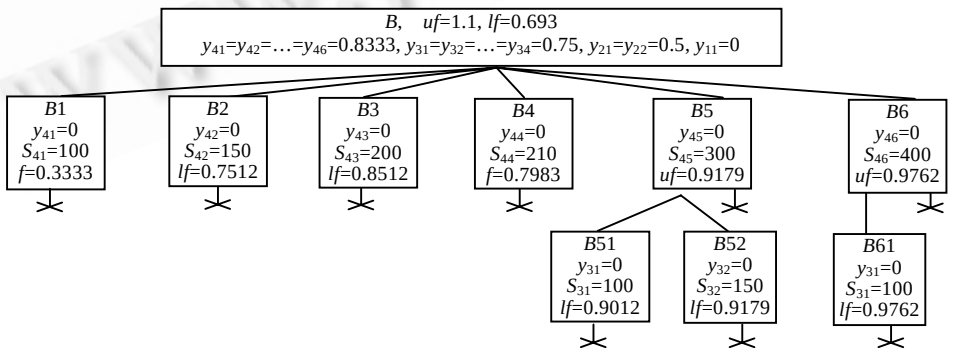

Fig.1 Solution space for real-time system design problem of Ex.1

图 1 例 1 实时系统设计问题的解空间

4 实验结果

实验利用 Matlab 优化工具包比较利用本文的两种数学模型与利用最小 CPU 利用率上界得到的设计结果.用 $U=\sum_{i=1}^{n}C_i/T_i$ 表示设计所得到任务集的性能指标,利用数学模型(式(10))求解问题称为非线性规划(nonlinear programming,简称 NLP)方法,利用数学模型(12)求解问题称为混合布尔型整数规划(mixed Boolean-integer programming,简称 MBP)方法.

4.1 实验1

针对例 1,比较 NLP,MBP 与利用最小 CPU 利用率上界 *LLB* 和 *HB* 进行设计时的性能参数.设计结果见表 2.从表 2 可以看出,NLP 和 MBP 方法可以找到性能参数指标要比利用最小 CPU 利用率上界所得到的指标优越很多的可调度的实时系统设计方案.本文的两种数学模型找到了两个不同的解,但目标函数值一致;而利用 *HB* 得到的 CPU 利用率要明显好于 *LLB*,间接说明了最小 CPU 利用率上界 *HB* 要比 *LLB* 乐观.

Table 2 Optimal task execution times and maximum utilization bound ($\zeta=10^4$, $\Delta v=10^{-6}$)

表 2 最优任务执行时间及最大 CPU 利用率($\zeta=10^4$,$\Delta v=10^{-6}$)

Task	Tasks execution times C_i			
	NLP	MBP	*LLB*	*HB*
τ_1	56.793 8	50.000 0	31.859 6	43.638 9
τ_2	20.000 0	20.000 0	27.906 4	20.000 0
τ_3	30.000 0	30.000 0	35.647 4	30.000 0
τ_4	52.824 3	80.000 0	32.964 9	30.000 0
U	0.976 2	0.976 2	0.756 8	0.787 6

4.2 实验2

实验比较本文两种不同数学模型求解得到的系统性能指标 U 及其时间开销,其中,时间开销以算法运算的内部循环次数进行评价.仿真条件如下:(1) 在[50,10^3]之间均匀、随机地选择任务周期 T_i;(2) 各任务的执行时间 C_i 在区间 $\lceil c,\lambda T_i\rceil$ 内,c 是一个正常数,λ 是一个调节系数以控制任务执行时间的上限,实验选取 $c=\lceil 0.1\times T_1\rceil,\lambda=0.5$;(3) 对具有相同任务数的任务集随机生成 10 个样本;(4) 算法精度为 0.0001(见表 3).

Table 3 Comparative study result between the NLP and MBP method

表 3 NLP 和 MBP 方法性能对比结果

Number of tasks	NLP model		MBP model	
	U	Iterations	U	Iterations
2	0.960 6	16	0.960 6	7
3	0.962 7	29	0.967 8	22
4	0.941 5	37	0.958 5	36
5	0.941 1	50	0.957 3	82
6	0.944 7	60	0.968 5	220
7	0.943 2	78	0.948 2	396
8	0.937 2	92	0.946 3	291
9	0.908 7	89	0.922 9	665
10	0.947 3	122	0.955 5	860
11	0.953 4	149	0.957 1	1 636
12	0.931 4	150	0.935 9	1 089
13	0.945 8	157	0.948 5	1 734
14	0.917 6	152	0.921 0	1 898
15	0.935 6	174	0.937 5	2 463
16	0.940 3	218	0.944 3	3 901
17	0.941 6	207	0.946 6	4 058
18	0.908 7	236	0.910 4	5 527
19	0.935 8	232	0.936 4	11 639
20	0.956 3	300	0.957 3	18 829

从表 3 可以看出:在任务集规模较小时,两种方法都能快速地找到可调度的任务集,并且随着规模的增大,两种方法求解问题的时间逐渐增加;在任务集规模较大时,NLP 方法仍然能很快地找到解,MBP 方法能找到最优解,但随着任务集规模的增大,计算工作量显著增加;MBP 方法所得到的目标函数值(CPU 利用率)要好于 NLP 方法所得到的值.

这些结果与算法特性是紧密相关的.NLP 方法利用非线性优化方法求解,充分利用函数的解析特性,求解速度快,其不足是有时只能找到局部最优解;MBP 方法实质上是一种全局搜索方法,当任务集规模很大时,最坏情况下要搜索的组合数相当大,导致寻找最优解速度较慢.设 $card(S_i)$表示集合 S_i 中元素的个数,MBP 方法最坏情况下要搜索的组合数等于 $card(S_1)\cdot card(S_2)\cdot\ldots\cdot card(S_n)$.由定理 1 可知,$card(S_i)$与 T_i/T_1 密切相关:T_i/T_1 越小,$card(S_i)$越少,并且 $card(S_1)<card(S_2)<\ldots<card(S_n)$.所以在任务集规模较大时,如果 T_n/T_1 比较小,则采用 MBP 方法;否则,采用 NLP 方法.这里给出两种方法的比较,希望发现各方法的优缺点,可根据具体情况进行取舍.

5 结 论

本文基于 RM 算法可调度判定充要条件的特征,结合我们以前对 RM 可调度性判定的研究成果,解决了实时系统针对一般任务集的任务执行时间设计问题.首先将找到 CPU 利用率最大的实时系统问题描述为一个 GCOP,然后利用一种新的等价数学变换,将其表示为混合布尔型整数规划问题,利用分支定界法求最优解.与其他方法相比,本文的方法能够确保找到性能指标最优的可调度实时系统.

如何进一步改进算法,如快速找到一个较高的 lf,减小计算工作量,是我们正在进行研究的.事实上,如果设定一个期望的目标函数值,当搜索到不小于该值的任务集时就停止,可以大幅度提高 MBP 算法的求解速度.同时,对某些特殊任务集,本文的方法是多项式计算时间的.限于篇幅,这里不予讨论.本文的优化方法不仅适用于实时系统领域,也适用于其他领域,如机器人路径规划等,有望在更多的工程或其他领域得到应用.

References:

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

[2] Bini E, Buttazzo GC, Buttazzo G. Rate monotonic analysis: The hyperbolic bound. IEEE Trans. on Computers, 2003,52(7): 933−942.

[3] Kuo TW, Chang LP, Liu YH, Lin KJ. Efficient online schedulability tests for real-time systems. IEEE Trans. on Software Engineering, 2003,29(8):734−751.

[4] Lehoczky JP, Sha L, Ding Y. The rate monotonic scheduling algorithm: Exact characterization and average case behavior. In: Proc. of the Real-Time Systems Symp. Santa Monica: IEEE Computer Society Press, 1989. 166−171. http://www-static.cc.gatech.edu/classes/AY2001/cs6235_spring/papers/

[5] Liu JX, Wang YJ, Cartmell M. An improved Rate Monotonic schedulability test algorithm. Journal of Software, 2005,16(1):89−100 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/89.htm

[6] Wang YJ, Chen QP. On the schedulability test of rate monotonic (RM) and its extendible algorithms. Journal of Software, 2004,15(6):799−814 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/799.htm

[7] Krishna CM, Shin KG. Real-Time Systems. Beijing: Tsinghua University Press, McGraw-Hill, 2001.

[8] Seto D, Lehoczky J, Sha L, Shin K. On task schedulability in real-time control systems. In: Proc. of the IEEE Real-Time Systems Symp. Washington: IEEE Computer Society Press, 1996. 13−21. http://www.cs.umd.edu/users/rich/courses/cmsc818G-s98/papers/

[9] Lu CY, Stankovic JA, Tao G, Son SH. Design and evaluation of a feedback control EDF scheduling algorithm. In: Proc. of the 20th IEEE Real-Time Systems Symp. Phoenix: IEEE Computer Society Press, 1999. 56−67. http://www.cs.virginia.edu/papers/

[10] Lu CY, Stankovic JA, Tao G, Son SH. Feedback control real-time scheduling: Framework, modeling and algorithms. Real-Time Systems Journal, Special Issue on Control-Theoretical Approaches to Real-Time Computing, 2002,23(1/2):85−126.

[11] Melissa AR, Evgenia S. Adaptive CPU scheduling policies for mixed multimedia and best-effort workloads. In: Proc. of the 7th IEEE Int'l Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems. IEEE Computer Society Press, 1999. 252−261. http://www.cs.wm.edu/~esmirni/docs/

[12] Chung JY, Liu JWS, Lin KJ. Scheduling periodic jobs that allow imprecise results. IEEE Trans. on Computers, 1990,19(9): 1156−1173.

[13] Liu JWS, Lin KJ, Shih WK, Yu ACS, Chung C, Yao J, Zhao W. Algorithms for scheduling imprecise computations. IEEE Computer, 1991,24(5):58−68.

[14] Dey JK, Kurose J, Towsley D. Online scheduling policies for a class of IRIS (increasing reward with increasing service) real-time tasks. IEEE Trans. on Computers, 1996,45(7):802−813.

[15] Aydin H, Melhem R, Mosse D, Mejia-Alvarez P. Optimal reward-based scheduling for periodic real-time tasks. IEEE Trans. on Computers, 2001,50(2):111−130.

[16] Alvarez PM, Melhem R, Mosse D, Aydin H. An incremental server for scheduling overloaded real-time systems. IEEE Trans. on Computers, 2003,52(10):1347−1361.

[17] Zhang YX, Fang CH, Wang Y. A feedback-driven online scheduler for processes with imprecise computing. Journal of Software, 2004,15(4):616−623 (in English with Chinese absrtact). http://www.jos.org.cn/1000-9825/15/616.htm

[18] Wang YJ, Lane DM. Solving a generalized constrained optimization problem with both logic AND and OR relationships by a mathematical transformation and its application to robot path planning. IEEE Trans. on Systems, Man and Cybernetics, Part C: Application and Reviews, 2000,30(4):525−536.

附中文参考文献:

[5] 刘军祥,王永吉,Cartmell M.一种改进的 RM 可调度性判定算法.软件学报,2005,16(1):89−100. http://www.jos.org.cn/1000-9825/16/89.htm

[6] 王永吉,陈秋萍.实时单调速率及其扩展算法的可调度性判定.软件学报,2004,15(6):799−814. http://www.jos.org.cn/1000-9825/16/799.htm

[17] 张尧学,方存好,王勇.非精确计算中基于反馈的 CPU 在线调度算法.软件学报,2004,15(4):616−623. http://www.jos.org.cn/1000-9825/15/616.htm

刘军祥(1975－),男,湖北潜江人,博士生,主要研究领域为实时系统,优化理论.

邢建生(1972－),男,博士生,主要研究领域为实时系统.

王永吉(1962－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为实时系统,网络优化,智能软件工程,优化理论,机器人,控制理论.

曾海涛(1979－),男,博士生,主要研究领域为安全实时系统.

王源(1981－),女,硕士生,主要研究领域为实时系统.