

文章编号:1001-9081(2010)11-2867-03

基于模型分解的多机带时间窗口任务规划算法

张利宁¹, 邱涤珊¹, 李皓平², 黄小军¹

(1. 国防科学技术大学 C4ISR 技术重点实验室, 长沙 410073; 2. 北方电子设备研究所, 北京 100191)

(zhanglining_0917@hotmail.com)

摘要:针对多机带时间窗口任务规划问题,提出了基于模型分解的规划求解算法。通过引入基于逻辑的 Benders 分解方法,将经典 Benders 分解算法应用扩展至带离散时间窗口的混合线性整数规划模型,实现模型分解。采用工艺级商业软件 MOSEK 与 GECODE 分别求解主、子问题,同时给出 Benders 剪枝函数生成方法,以迭代方式收敛解空间获得可行解。实现算法并设计测试案例,实验结果验证了算法的有效性。

关键词:模型分解; Benders 分解; 任务规划; 时间窗口; 组合优化

中图分类号: TP301.6 **文献标志码:** A

Task scheduling algorithm based on model decomposition for multi-machine with time windows

ZHANG Li-ning¹, QIU Di-shan¹, LI Hao-ping², HUANG Xiao-jun¹

(1. Key Laboratory of C4ISR Technology, National University of Defense Technology, Changsha Hunan 410073, China;

2. Northern Electronic Equipment Institution, Beijing 100191, China)

Abstract: The algorithm based on model decomposition was proposed for the problem of multi-machine task scheduling with time windows. The classical Benders decomposition was extended into the field of mixed integer linear programming model by introducing the logic based Benders decomposition. The states of an software MOSEK and GECODE were deployed to solve the master and sub-problems respectively. The method of generating Benders cuts was presented. The solution space was converged to a satisfied feasible solution through running the algorithm iteratively. The algorithm was implemented and tested by testing cases, and its effectiveness was verified.

Key words: model decomposition; Benders decomposition; task scheduling; time window; combinatorial optimization

0 引言

任务规划问题是在工业生产以及供应链领域一种常见问题^[1]。该问题形式多样,任务间常常存在序列关系,高效求解仍较困难。文献[2]中求解多协作车间的计划调度问题,提出了并行协同进化遗传算法;文献[3]针对最小化加权完工时间的等同和非等同并行多机调度一类问题,提出了一种递阶遗传算法;文献[4]根据双向冲压线的实际生产特点,提出了一种基于工序约束并行机的双向冲压线调度模型。

除遗传算法以及其他基于种群的启发式或元启发式算法外,采用数学规划方法求解也是一种途径。在文献[5]中,Hooker 等人将 Benders 分解方法扩展至基于逻辑的一般 Benders 分解方法,并建立相应理论体系。基于逻辑的 Benders 分解方法使得混合整数规划方法与约束规划方法的结合成为可能。因此,对于大规模混合整数规划问题,建立问题的线性混合整数规划模型,采用模型分解方法对问题进行分解,获得主、子问题模型,进一步得到优化边界函数,重新加入主问题模型进行迭代求解,是基于上述思路解决此类问题的一种典型过程。线性混合整数规划模型的经典分解方法包括列生成算法和 Benders 分解算法。

由于混合整数规划模型可以很好地描述多机任务规划问

题,而约束整数规划问题能够高效解决单机规划问题。本文在建立多机带时间窗口任务规划问题的混合整数线性规划模型后,按照基于逻辑的 Benders 分解方法对模型分解。使用工艺级混合整数规划求解软件 MOSEK 以及约束规划软件 GECODE,分别求解主问题(任务指派问题)以及子问题(单机规划问题)。并给出 Benders 剪枝函数的生成方法,通过迭代求解得到满意可行解。

1 多机带时间窗口任务规划问题模型

多机带时间窗口任务规划问题的混合线性整数规划模型可以描述如下,记为“问题1”:

$$\begin{aligned} \min & \sum_{ij} F_{ij} x_{ijt} \\ \text{s. t.} & \sum_{it} x_{ijt} = 1; \quad \text{all } j \end{aligned} \quad (1)$$

$$\sum_j \sum_{t' \in T_{ijt}} c_{ijt'} x_{ijt'} \leq C_i; \quad \forall i, t \quad (2)$$

$$x_{ijt} = 0; \quad \text{for } t \leq r_j \quad \text{or} \quad t > d_j - p_{ij} \quad (3)$$

其中:任务集合为 $j \in [1, n]$, 机器集合为 $i \in [1, m]$ 。任务 j 在机器 i 上的处理时间为 p_{ij} , 资源消耗为 c_{ij} , 机器 i 的资源总量为 C_i 。每一个任务都有一个释放时间 r_j 和加工截止时间 d_j 。 x_{ijt} 为 0-1 变量, 当任务 j 于时刻 t 在机器 i 上开始加工, 则 x_{ijt} 赋值为 1; F_{ij} 表示机器 i 加工 j 的价格系数。 $T_{ijt} = \{t' \mid t - p_{ij} <$

收稿日期:2010-04-10;修回日期:2010-09-15。 基金项目:国家973计划项目(97361361)。

作者简介:张利宁(1981-),男,陕西宝鸡人,博士研究生,主要研究方向:军事运筹学、组合优化算法、人工智能;邱涤珊(1957-),男,湖南南县人,教授,博士生导师,主要研究方向:军事运筹学、航天装备军事应用;李皓平(1982-),女,吉林长春人,工程师,主要研究方向:人工智能、移动通信;黄小军(1981-),男,四川峨眉山人,博士研究生,主要研究方向:军事运筹学、组合优化算法、人工智能。

$t' \leq t\}$ 为机器 i 加工任务 j 的可选离散时间窗口集合。约束(1) 保证每个任务最多被加工 1 次, 约束(2) 为机器资源约束, 约束(3) 是离散时间窗口约束。该模型的优化目标为总的加工价格最小化。

2 基于逻辑的 Benders 分解方法

在介绍基于逻辑的 Benders 分解方法之前, 先给出混合整数线性规划模型引申对偶性的定义。对于一个混合整数线性规划模型, 如式(4) 所示, 记为“问题 2”。

$$\begin{aligned} \min f(x) \\ \text{s. t. } x \in C(x) \\ x \in D_x \end{aligned} \quad (4)$$

其中: $C(x)$ 表示作用在变量 x 上的约束集, D_x 为 x 的定义域。首先引入一个语义描述形式。它是定义在变量定义域上, 对命题推论的一种语义描述。假设有两个命题 P 和 Q , 它们的取值(真-1 或假-0) 是变量 x 的函数, P 在 x 的定义域 D_x 上推出 Q (表示为 $P \xrightarrow{D_x} Q$), 如果对于任意 $x \in D_x$ 命题 Q 为真, 则 P 也为真。那么, “问题 2” 的引申对偶问题可以定义为式(5), 记为“问题 3”。

$$\begin{aligned} \max \beta \\ \text{s. t. } x \in C(x) \xrightarrow{D} f(x) \geq \beta \end{aligned} \quad (5)$$

该对偶问题寻找能够由约束集 $f(x) \geq \beta$ 推导出来的最大 β 值, 即对偶问题是寻找原问题目标函数值的最大下界。当原问题无解或者具有无界解时, 可以将最小化问题的目标函数值看做是 ∞ 以及 $-\infty$, 在最大化问题情况下反之亦然。我们同时假设“问题 1” 具有优化解(可能是 ∞ 或者 $-\infty$)。在这个假设条件下, 根据一般线性规划模型的强对偶性^[1], 问题 2” 总是与其引申对偶问题, 即“问题 3” 具有相同的最优解。

接下来, 对于如式(6) 所示问题, 记为“问题 4”。

$$\begin{aligned} \min f(x, y) \\ \text{s. t. } (x, y) \in C(x, y) \\ x \in D_x, y \in D_y \end{aligned} \quad (6)$$

其中: $C(x, y)$ 表示包含了变量 x 和 y 的约束集合, D_x 和 D_y 分别为 x 和 y 的定义域。当固定变量 x 为一固定值 $\bar{x} \in D_x$, 则生成如(7) 所示子问题, 记为“问题 5”:

$$\begin{aligned} \min f(\bar{x}, y) \\ \text{s. t. } (\bar{x}, y) \in C(\bar{x}, y) \\ \bar{x} \in D_x \end{aligned} \quad (7)$$

其中 $C(\bar{x}, y)$ 是通过固定变量 x 为值 \bar{x} 得到的约束集。这样, “问题 4” 的引申对偶问题就能通过约束 $C(\bar{x}, y)$ 生成目标函数 $f(\bar{x}, y)$ 的最小下界, 如式(8), 记为“问题 6”。

$$\begin{aligned} \max v \\ \text{s. t. } C(\bar{x}, y) \xrightarrow{P} f(\bar{x}, y) \geq v \\ v \in \mathbf{R}, p \in P \end{aligned} \quad (8)$$

这里 $A \xrightarrow{P} B$ 表示通过证明 P 能够从 A 推导出 B , P 是有效证明集合, 当“问题 4” 是一个标准线性规划问题时, 则“问题 6” 亦是一般线性规划对偶问题。

对偶问题的解可以认为是: 当变量 x 为值 \bar{x} 时, 目标函数值 $f(\bar{x}, y)$ 最小上界 \hat{v} 的证明。在经典 Benders 分解方法中, 也是采用同样的证明模式, 生成一个针对所有 x 值的边界

$B_{\bar{x}}(x)$ 。一般来讲, 边界函数 $B_{\bar{x}}(x)$ 应具备下面两条属性。

1) 对于任意给定 $x \in D_x$, $B_{\bar{x}}(x)$ 是目标函数值 $f(x, y)$ 的合法边界, 即对“问题 4” 中的任意可行解 (x, y) 都有 $f(x, y) \geq B_{\bar{x}}(x)$ 。

2) 特别地, 有 $B_{\bar{x}}(\bar{x}) = \hat{v}$ 。

令 z 是“问题 4” 的目标函数值, 则不等式 $z \geq B_{\bar{x}}(x)$ 称为一个 Benders 剪枝函数。假设在 Benders 算法的第 H 个循环中, 需要求解一个主问题, 如式(9) 所示, 记为“问题 7”, 约束集是已生成的 Benders 剪枝函数集合。

$$\begin{aligned} \max z \\ \text{s. t. } z \geq B_{\bar{x}^h}(x); h = 1, \dots, H-1 \\ z \in \mathbf{R}, x \in D_x \end{aligned} \quad (9)$$

其中: x^1, \dots, x^{H-1} 是之前 $H-1$ 个循环中主问题的解, 而“问题 7” 的解 \bar{x} 则又定义了下一个子问题。

假设 v_1^*, \dots, v_{H-1}^* 表示前 $H-1$ 个循环中子问题的优化解, 算法持续进行直到主问题的优化解 z_H^* 等于 $v^* = \min\{v_1^*, \dots, v_{H-1}^*\}$, 在算法进行过程中的任意时刻, z_H^* 和 v^* 是原问题目标函数值的下界和上界。为了更有助于问题理解, 当“问题 4”、“问题 5” 不具有可行解时, 可以认为这两个问题具有无界解。

定理 1 如果边界函数 $B_{\bar{x}}(x)$ 在 Benders 算法中的每一个循环中满足条件 1) 和 2), 并且变量 y 定义域 D_y 为有限集, 则 Benders 算法经过有限步循环可以收敛至“问题 4” 的优化解(证明参见文献[5])。

3 基于模型分解的任务规划求解过程

对于第 1 章中提出的多机带时间窗任务规划问题模型, 任一面向多机的任务指派方案 \bar{x} 生成如下子问题:

$$\begin{aligned} \min g(\bar{x}, t) \\ \text{s. t. } \text{cumulative}((s_j | \bar{x}_j = i), (p_{ij} | \bar{x}_j = i), \\ (c_{ij} | \bar{x}_j = i), C_i), \text{ all } i \\ r_j \leq t_{ij} \leq d_j - p_{ij}, \text{ all } j \end{aligned} \quad (10)$$

其中: $\text{cumulative}()$ 是全局约束描述, t_{ij} 表示任务 j 在机器 \bar{x}_j 上的开始执行时间。这样可以将原问题分解为多个单机任务规划问题。通过使用约束规划问题求解工具求解, 生成“问题 7” 中的剪枝函数。根据定理 1, 定义域 D_i 为有限集。这也符合大多数问题背景。通过分析给定变量 x 在赋值为 \bar{x} 时产生边界的原因, 并将此推广到 x 的其他赋值来获取边界函数。

下面通过分析每一个单机规划问题的解(即单机任务规划方案), 来寻找合理的 Benders 剪枝函数。

在“问题 1” 中, 由于目标函数为代价函数, 因此目标函数值的计算仅与主问题解相关。当子问题可行时, 可认为其目标函数值即为 $\sum_j F_{x_{ij}}$, 不可行时, 认为目标函数值为 ∞ 。令集合 $J_{hi} = \{j | x_j^h = i\}$ 表示在第 h 个循环中分派给机器 i 的所有任务集合。如果对于机器 i , 规划问题可能不具有可行解, 在这种情况下, 最简单的一个剪枝函数就是把 J_{hi} 中所有的任务都从机器 i 的任务集中去除。此时, 边界函数具有如下形式:

$$B_{x^h}(x) = \begin{cases} \infty, & \text{当机器 } i \text{ 上的规划问题不可行} \\ \sum_j F_{x_{ij}}, & \text{其他} \end{cases} \quad (11)$$

边界函数 $B_{x^h}(x)$ 满足条件 1), 因为对于任意可行解 (x, s) 都有 $\sum_j F_{x_{ij}}$ 等于 $B_{x^h}(x)$; 而不可行时, $\sum_j F_{x_{ij}}$ 也可认为其

无界。由于 $B_{zh}(x^h)$ 是子问题的优化解,因此满足条件 2)。

然而,这样的边界函数约束性太弱,这是因为并不是集合 J_{hi} 中的所有任务导致了单机规划问题不可行,在大多数情况下是 J_{hi} 的一个子集导致;并且单机规划器无法记录导致不可行的任务,实际应用当中此类信息不可获取,所以这里通过一个贪婪算法来找到 J_{hi} 中导致单机规划问题不可行的任务子集。初始化集合 $\bar{J}_{hi} = J_{hi} = \{j_1, \dots, j_k\}$, 对于 $l = 1, \dots, k$, 做如下操作:尝试在机器 i 上规划任务 $\bar{J}_{hi} \setminus \{j_l\}$, 如果没有可行解,则从集合 \bar{J}_{hi} 中将 j_l 去除。最后用 \bar{J}_{hi} 代替式(11)中的 J_{hi} , 此时条件 1) 和 2) 仍然满足。这样虽然对单机重新规划 k 次,但同时获得了约束性更强的边界函数。

得到边界函数后,重写主问题模型,令集合 I_h 为第 h 个循环中,不具有可行规划方案的机器集合。那么,主问题形式如下:

$$\min \sum_{ij} F_{ij} x_{ij}$$

$$\text{s. t. } \sum_i x_{ij} = 1; \forall j \in J \quad (12)$$

$$\sum_{j \in J_{hi}} (1 - x_{ij}) \geq 1; i \in I_h; h = 1, \dots, H-1 \quad (13)$$

$$\text{relax of sub-problem} \quad (14)$$

其中: $x_{ij} \in \{0, 1\}$, 式(13)即为剪枝约束。

在主问题求解过程中加入子问题约束可以加速求解收敛。这里给出关于任务执行时间 t 的松弛约束生成方法。对任意两个时刻 t_1 和 t_2 , 令 $J(t_1, t_2)$ 表示时间窗口落在 t_1 和 t_2 之间的任务集合, 即 $[r_j, d_j] \subset [t_1, t_2]$, 如果任务 $j \in J(t_1, t_2)$ 被分配至同一机器 i , 如果能够生成可行规划方案, 根据资源有限约束, 则必然有:

$$\frac{1}{C_i} \sum_{j \in J(t_1, t_2)} p_{ij} c_{ij} x_{ij} \leq t_2 - t_1 \quad (15)$$

记为不等式 $R^i(t_1, t_2)$, 将集合 $\{r_1, \dots, r_n\}$ 中的值不同元素按照增序排列为 $\bar{r}_1, \dots, \bar{r}_{n_r}$, 同理可以得到 $\bar{d}_1, \dots, \bar{d}_{n_d}$, 这样对于机器 i 可以得到如下不等式集合:

$$R^i(\bar{r}_j, \bar{d}_k) \quad (16)$$

其中: $j = 1, \dots, n_r; k = 1, \dots, n_d$ 。这些约束即为式(14)中的子问题约束(relax of sub-problem)。这些约束需要转化为变量 x_{ij} 的线性松弛。约束松弛集合中, 有可能存在冗余约束, 因此需要将其从约束集中删除。可以证明对于时间窗口 $[t_1, t_2]$ 和 $[u_1, u_2]$, 如果 $[t_1, t_2] \subset [u_1, u_2]$, 则有 $T^i[t_1, t_2] > T^i[u_1, u_2]$, 其中:

$$T^i[t_1, t_2] = \frac{1}{C_i} \sum_{j \in J(t_1, t_2)} p_{ij} c_{ij} - t_2 + t_1 \quad (17)$$

此时约束 $R^i(u_1, u_2)$ 为冗余约束, 可从约束集中删除。查找并删除冗余约束算法的过程的伪代码如下所示:

Let $R_i = \emptyset$

For $j = 1, \dots, p$:

Set $k' = 0$

For $k = 1, \dots, q$

If $r_k \geq r_j$ and $T^i[\bar{r}_j, \bar{d}_k] > T^i[\bar{r}_j, \bar{d}_{k'}]$

Remove from R_i all $R^i(\bar{r}_j, \bar{d}_k)$ for

which $T^i[\bar{r}_j, \bar{d}_k] > T^i[\bar{r}_j, \bar{d}_{k'}]$

Add $R^i(\bar{r}_j, \bar{d}_k)$ to R_i , and $k' = k$

该算法的时间复杂度最高为 $O(n^3)$ 。在实际算法应用中, 可以假设所有的任务释放时间均为 $r_0 = \min\{r_j\}$, 这样对于每

个机器, 约束(13)包含:

$$R^i(r_0, \bar{d}_k); k = 1, \dots, n_d \quad (18)$$

此时, 可以采用如下所示的算法进行冗余约束消除, 此算法的时间复杂度简化为 $O(n)$ 。

Let $R_i = \emptyset, j = 0$

For $k = 1, \dots, p_d$:

If $T^i[r_0, \bar{d}_k] > T^i[r_0, \bar{d}_j]$

Add $R^i(r_0, \bar{d}_k)$ to R_i , and $j = k$

在给出总的求解过程后, 下面介绍求解主问题与子问题采用的求解工具。

由于主问题模型属于标准混合整数线性规划模型, 这里采用求解软件 MOSEK5.0^[6] 完成求解。子问题, 即一系列单机规划问题采用开源约束规划工具 GECODE^[7] 求解。

4 方法验证与结果分析

4.1 测试问题构造

测试问题采用随机方式构造。对每一个机器, 令其资源约束为 $C_i = 10$, 对任务 j , 在机器 i 上的资源消耗率 c_{ij} 在区间 $[1, 10]$ 内随机产生。机器数量为 m , 任务数量为 n , 任务 j 在机器 i 上的加工时间 p_{ij} 在区间 $[i, 10i]$ 内随机生成。每一个任务的释放时间 r_j 置为 0, 即同时释放。由于 m 个机器的平均最长加工时间为 $5(m+1)$, 因此所有任务的总加工时间接近 $5n(m+1)$ 。每个机器的加工截止时间为 $\alpha L, L = 5n(m+1)/m$, 通过调节 α 值来生成不同的测试问题。价值系数 F_{ij} 在区间 $[2(m-i+1), 20(m-i+1)]$ 内随机生成。

实验进行 4 组, 分别记为 T_1, T_2, T_3, T_4 , 对应参数设置如表 1 所示。

表 1 实验参数设置

实验名称	m	n	α
T_1	50	5	0.50
T_2	100	10	0.60
T_3	200	15	0.75
T_4	300	20	1.00

4.2 外部求解插件接口与测试环境

与外部求解插件之间交互采用文件输入输出方式。与混合整数规划求解软件 MOSEK 的接口, 以及约束规划求解软件 GECODE 接口均通过标准 OPL 文件。算法程序在 VC.net 环境下开发, 测试环境为 2.60 GHz 双核处理器, 2 GB RAM。

4.3 实验结果与分析

从两个角度对实验结果进行分析, 首先是运行结果, 评价指标包括总价值消耗, 各机器占用率; 其次是运行效率, 评价指标包括运行时间、内存占用率以及 Benders 剪枝约束数量 (Benders cuts)。

运行结果指标值如表 2 所示。

表 2 运行结果指标值

实验名称	总资源消耗	平均机器占用率/%
T_1	56.4	45.00
T_2	107.2	57.00
T_3	169.3	81.00
T_4	271.8	97.50

运行效率指标值如表 3 所示。

(下转第 2909 页)

若信号 j 亦能观测,考查的故障诊断系统则变成确定性问题。而通过 MSBN 法,诊断结果为器件 $go4$ 后验故障概率 = 1.00, 而其他器件后验故障概率均小于先验故障概率,可以确定出仅有 $go4$ 故障。这说明了 MSBN 诊断方法的正确性。

若仅有信号 j 是不可观测的,而其他信号均已观测获得,经过 MSBN 法推理,则仍有 $go4$ 和 $go8$ 2 个门电路出现故障概率最大(后验故障概率分别为 0.502 512 700 和 0.502 512 450),但信度与 4.3 节用部分观测方法所得相差不大,说明了基于图模型多智能体故障诊断方法具有较强的鲁棒性。

表 1 第一轮检测后故障概率推理结果

子系统	器件	局部推理	全局推理
C_0	$go8$	0.006 672 613	0.009 808 213
	$gn10$	0.010 000 001	0.010 000 000
	$ga9$	0.010 000 000	0.010 000 000
C_1	$go1$	0.005 025 125	0.005 025 124
	$gn2$	0.010 000 000	0.010 000 002
	$ga3$	0.009 999 999	0.014 687 307
	$go4$	0.010 000 003	0.014 663 284
C_2	$ga5$	0.007 512 562	0.014 687 311
	$go6$	0.007 512 562	0.005 025 125
	$gn7$	0.010 000 002	0.014 711 103

4.5 实验分析

由于复杂、不确定系统中某些门电路出现故障,可导致观测信号间出现矛盾,要使系统恢复正常工作状态,一种解决办法是替换所有的门电路,这无疑是一种成本昂贵、不甚可行的做法。考查 C_1 子系统可知,由于设备 $go4$ 的输出始终不可观测,采用传统故障诊断方法无法判断 $go4$ 的状态是否正常。

而基于图模型多智能体的系统故障诊断方法,利用智能体间少量的通信和协同推理,可将故障范围大大缩小。如图 3 中,尽管某些信号始终为不可观测的(信号 j),但经过分布式各 Agent 自主的推理及智能体间紧凑、有效的通信,可有效弥补局部传感器观测能力的不足,协同推理结果将出现故障范围的 10 个门电路有效缩减至 2 个,大大提高了复杂、不确定系统的故障诊断效率,并且实验验证了该诊断方法的正确

性和鲁棒性。

5 结语

为解决复杂、不确定系统的故障诊断问题,本文提出了基于 MSBN 架构下多智能体协同推理的故障诊断方法。理论分析和仿真结果表明了利用图模型多智能体协同故障诊断方法的有效性和正确性,为解决复杂、不确定性系统的故障诊断提供了一种具有广泛应用前景的新手段。

参考文献:

- [1] 朱大奇. 电子设备故障诊断原理与实践[M]. 北京: 电子工业出版社, 2002.
- [2] 杨昌昊, 胡小建, 竺长安. 从故障树到故障贝叶斯网映射的故障诊断方法[J]. 仪器仪表学报, 2009, 30(7): 1481–1486.
- [3] RUSSELL S, NORVIG P. Artificial intelligence: A modern approach [M]. 2nd ed. New Jersey: Prentice Hall, 2002.
- [4] CORREA M, BIELZA C, PAMIES-TEIXEIRA J. Comparison of Bayesian networks and artificial neural networks for quality detection in a machining process [J]. Expert Systems with Applications: An International Journal, 2009, 36(3): 7270–7279.
- [5] JENSEN F V. Bayesian networks and decision graphs [M]. Berlin: Springer-Verlag, 2001.
- [6] ZHANG Y, MANISTERSKI E, KRAUS S, et al. Computing the fault tolerance of multi-Agent deployment [J]. Artificial Intelligence, 2009, 173(3/4): 437–465.
- [7] XIANG Y. Probabilistic reasoning in multi-Agent systems: a graphical models approach [M]. Cambridge: Cambridge University Press, 2002.
- [8] XIANG Y, JENSEN F V, CHEN X. Inference in multiply sectioned Bayesian networks: methods and performance comparison [J]. IEEE Transactions on Systems, Man, and Cybernetics, 2006, 36(3): 546–558.
- [9] XIANG Y, LESSER V. On the role of multiply sectioned Bayesian networks to cooperative multi-Agent systems [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part A, 2003, 33(4): 489–501.

(上接第 2869 页)

表 3 运行效率指标值

实验名称	运行时间/s	内存占用率/%	Benders 剪枝约束数量
T_1	56	31	127
T_2	112	39	384
T_3	379	46	562
T_4	547	69	745

5 结语

本文在给出多机带时间窗口任务规划问题的混合线性整数规划模型基础上,通过引入引申对偶问题以及基于逻辑的 Benders 分解方法,给出了基于模型分解的多机带时间窗口任务规划求解算法。算法核心包括模型分解过程与 Benders 剪枝函数的生成过程。

下一步研究工作将集中在任务之间带有执行序列关系约束的多机任务规划问题,在此研究基础上对已有算法进一步扩展。

参考文献:

- [1] 运筹学教材编写组. 运筹学[M]. 北京: 清华大学出版社, 2005: 35–41.
- [2] 于晓义, 孙树栋. 基于并行协同进化遗传算法的多协作车间计划调度[J]. 计算机集成制造系统, 2008, 14(5): 991–1000.
- [3] 周辉仁, 郑丕谔, 王海龙. 基于递阶遗传算法的最小加权完工时间并行机调度[J]. 系统仿真学报, 2008, 20(13): 3510–3514.
- [4] 李峥峰, 喻道远, 杨曙年. 基于工序约束并行机模型的冲压线调度[J]. 计算机集成制造系统, 2009, 15(12): 2432–2438.
- [5] HOOKER J N, OTTOSSON G. Logic-based Benders decomposition [J]. Mathematical Programming, 2003, 96(1): 33–60.
- [6] The MOSEK C API manual [EB/OL]. [2009–12–12]. http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/capi/index.html.
- [7] SCHULTE C, TACK G, LAGERKVIST M Z. Modeling with Gecode [EB/OL]. [2009–10–12]. <http://cs.swan.ac.uk/~csoliver/ok-sat-library/OKplatform/ExternalSources/sources/CSP/Gecode/modeling.pdf>.