文章编号:1001 - 9081(2005)08 - 1829 - 04

单机序列相关 Setup 最小化最大拖期算法

罗小川,王成恩
(东北大学 教育部暨辽宁省流程工业综合自动化重点实验室,辽宁 沈阳 110004)
(luoxch@pal.neu.edu.cn)

摘 要:研究了一个具有序列相关 Setup 带交货期的单机调度 NP 问题,优化目标是最小化最大拖期。提出了一个求解该问题的分枝定界枚举算法,其中包括确定问题上界和下界的方法,以及两条优势规则。计算实验证明了本文提出算法的有效性。

关键词:序列相关 Setup;交货期;最大拖期;分枝定界;单机调度
中图分类号: TP316.4 **文献标识码**:A

Algorithm for single machne schedule with sequence dependent setup to minimize maximum tardiness

LUO Xiao-chuan, WANG Cheng-en
(*Key Laboratory of Process Industry Automation, Ministry of Education and Liaoning Province, Shenyang Liaoning* 110004, *China*)

Abstract: The NP-hard problem of scheduling N jobs on a single machine with due dates, sequence-dependent setup was addressed, where the objective was to minimize the maximum tardiness. An algorithm based on branchand-bound permutation schemes was developed including the implementation of lower and upper bounding procedures, and two dominance rules. Computational experiments demonstrate the effectiveness of the algorithm.

Key words: sequence dependent setup; due date; maximum tardiness; branch and bound; single machine schedule

0 引言

随着全球市场竞争的日益激烈,满足交货期成为客户满意的一个重要内容。如果客户合同出现拖期,工厂就要受到拖期的惩罚,因而工厂非常关心能否在交货期完成合同,关于拖期的调度问题是当前的一个热点内容。同时,在许多生产系统中经常遇到序列相关 Setup 调度问题,Setup 既取决于刚刚完成的作业,也和立即要处理的作业有关[1]。工业实践和科学研究都表明有效管理序列相关 Setup 是改进系统性能的关键因素[2]。文献[3]完成的仿真研究结果阐明了序列相关的 Setup 对效率的影响。虽然序列相关 Setup 已成为调度中的常见因素,但序列相关 Setup 的拖期调度问题已被证明是 NP-hard[4],亟需有效的方法来解决工厂应用中的这个难题。

文献[5]提出含有序列相关 Setup 的并行机器的调度问题,并通过启发式过程来实现随机工件到达的最小化最大拖期问题。文献[6]针对序列相关 Setup 单机调度问题,提出了一种基于拉格朗日松弛(Lagrangian relaxation)的算法来最小化所有作业拖期的加权平方和。文献[7]提出一种分枝定界方案用于解决序列相关 Setup 单处理器的最小化最大拖期问题。该算法在求解较大规模问题时比较困难,特别是在交货期变化范围狭小或作业处理时间波动大的情况下。文献[8]研究比较了在求解该问题时分枝定界、遗传搜索[9]和模拟退火法[10]等不同算法的性能。

人们同时提出了一些启发式算法和规则,如 ATCS (Apparent Tardiness Cost with Setups)[11]和扩展 ATCS[12]的规则。局部搜索方法也应用到解决交货期和 Setup 问题。文献[9]提出了一种遗传算法用于解决序列相关 Setup 的最小化拖期惩罚问题。文献[13]采用蚁群优化方法来研究带有序列相关 Setup 的单机调度问题。然而这些方法仅仅提供了近似最优解,不能确保获得最优解。

收稿日期:2005 - 01 - 22;修订日期:2005 - 04 - 06

作者简介:罗小川(1974 -),男,四川西充人,副教授,博士,主要研究方向:生产计划调度、分布式系统设计和优化; 王成恩(1964 -),男,黑龙江鸡东人,教授,博士生导师,主要研究方向:CIMS 建模、设计与实现.

在的问题,同时分析了负项关联规则的产生方式。在此基础上,本文给出了参考度的评价标准,提出了一个基于参考度的关联规则挖掘算法,同时也将其用于挖掘负关联规则。最后,本文对算法进行了实验分析和讨论。

参考文献:

[1] Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases[A]. Proceedings of the ACM SIGMOD Conference on Management of Data[C]. Washington, 1993. 207 - 216.

[2] Srikant R, Agrawal R. Mining quantitative association rules in large relations talbes[A]. Proceedings of the ACM SIGMOD Conferece on Management of Data[C]. Montreal, Canada, 1996. 1 - 12.

[3] Brin S, Motwani R, Silverstein C. Beyond market baskets: Generalizing association rules to correlations[A]. Proceedings of the ACM SIGMOD Conference on Management of Data[C]. Tucson, USA, 1997. 265 - 276.

[4] Brins S, Motwani R, Ullman JD, *et al*. Dynamic itemset counting and implication rules for market basket data[A]. Proceedings of the ACM SIGMOD Conf on Management of Data[C]. Tucson, USA, 1997. 255 - 264.

[5] Agarwal CC, Yu PS. Online generation of association rules[A]. Proceedings of the Int'l Conference on Data Engineering[C]. Orlando, Florida, USA, 1998. 402 - 411.

[6] 周皓峰,朱扬勇,施伯乐.一个基于兴趣度的关联规则裁决算法[J].计算机研究与发展,2002,39(4):450 - 457.

本文提出一种基于分枝定界思想的算法来求解具有序列相关 Setup 的最小化最大拖期单机调度问题(简称 SDSMT),并通过随机产生的数据验证了这一算法的有效性。

1 问题描述

1.1 标记定义

1)作业索引和集合

N:作业处理总数;J:所有作业的集合,$J = \{1,2,3,\cdots,N\}$;$J_0 = J \cup 0$:处理作业的扩展集合,包括一个虚拟作业,用"0"表示;i,j,k,l:处理作业索引号,$i,j,k,l \in J_0$;S:以虚拟作业0开始的序列,为问题的一个解;K:以虚拟作业0开始的部分序列,相当于搜索树的前k级节点。

2)输入数据

p_i,d_i:作业的处理时间和交货期;C_{ij}:完成作业i后处理作业j的费用;S_{ij}:在调度序列S中作业i之后立即安排作业j所需要的*Setup*费用。

3)计算输出参数

C_i:作业i的完成时间;T_i:作业i的拖期值;$T_{\max}$:所有拖期T_i的最大值,$T_i \in J_0$。

作业i(没有括号)指作业i本身,而作业$[i]$指在作业调度序列中的第i位置的作业索引。

1.2 假设

(1)全部作业在准备好;(2)处理时间$p_i, i \in J$,在进行处理之前已知;(3)作业无优先级别,不可分割作业;(4)机器不能同时处理两个或多个作业;(5)在$t = 0$机器处理虚拟作业0,且$p_0 = 0, d_0 = 0, C_0 = 0, s_{0i}, i \in J$,取决于从处理作业0进入处理作业$i$的费用,作业0和作业$[0]$都对应于虚拟作业0;(6)$s_{ij} + s_{jk} \geq s_{ik}$。

1.3 问题的数学描述

假设有N项作业索引号从1到N,每项作业$j \in J$有处理时间p_j、交货期d_j和*Setup* s_{ij}。令$S = \{[0],[1],[2],\cdots,[N]\}$为一个所有作业的调度序列,序列$S$第$i$位置作业的拖期是:

$$T_{[i]} = \max\{C_{[i]} - d_{[i]}, 0\} \tag{1}$$

这里$C_{[i]}$是作业$[i]$的完成时间:

$$C_{[i]} = \begin{cases} 0 & i = 0 \\ \sum_{k=1}^{i} (s_{[k-1][k]} + p_{[k]}) & i \neq 0 \end{cases}$$

希望找到一个序列能最小化作业序列的最大拖期,序列S的最大拖期是:

$$T_{\max} = \max_{i \in [0,1,2,\cdots,N]} T_{[i]} \tag{2}$$

2 分枝定界算法

为了找出最优序列S^*得到最大拖期的最小值,提出了分枝定界隐式枚举法来求解SDSMT问题。通常分枝定界算法在解最小化问题时有四个步骤:分解、定界、分枝和探测。

2.1 分解

搜索树第k级的一个节点对应于部分序列$K = \{[0],[1],[2],\cdots,[k]\}$,共含有$k+1$个作业。通过将剩下的未调度作业分别放在第$k+2$位置,可以构造出新的部分序列,从而实现此节点的分解。

2.2 定界

当部分序列K处理完成,可以由:(1)由序列K中作业产生的实际拖期;(2)由未调度的$N-k$个作业的拖期,获得当前子问题最大拖期的下界。我们提出了用分配问题算法计算下界的方法。在定出下界的基础上,采用改进 Karp-Steel 补偿启发式方法(modified Karp-Steel patching heuristic)构造一个可行解作为上界。

2.2.1 SDSMT 的下界

通过松弛子路径连通约束,可以用分配问题(Assignment Problem,AP)求解方法来获 SDSMT 下界。

定理1 AP问题是SDSMT的下界问题。

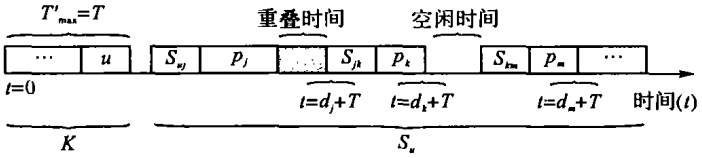


图1 序列中的重叠时间与空闲时间示例

证明:如图1所示,设U_k为未调度作业集合,$N_u = |U_k|$,T是部分调度序列K的最大拖期,作业u为K的最后一个作业。序列S_u是U_k子问题的可行解。$T[j]([j] \in U_k)$是序列S_u第j位置作业的拖期时间,原始问题的目标函数可改写为:

$$\min Z = \max(T, \max_{[j] \in U_k}(T_{[j]})) \tag{3}$$

从式(3)可知,最小值Z^*一定为$Z^* \geq T$。若$T_{[j]} \leq T, Z^*$值为T。若想保持$T_{[j]} \leq T$,集合U_k形成的作业序列中,前后两个作业之间会出现重叠时间或者空闲时间。这样,对每一作业$[j](j \in U_k)$有$T_{[j]} = T$,则有$Z^* = \max(T,T) = T$。

因为最小化Z就是最小化$\max_{[j] \in U_k}(T_{[j]})$,也就是$Z$取决于$\max_{[j] \in U_k}(T_{[j]})$的最小值。考虑$T_{[j]} = T$时的重叠时间和空闲时间,设作业$i$为作业$j$直接前驱时的重叠时间为$OI_{ij}(i,j \in U_k)$,则$OI_{ij} \leq 0$时为空闲时间。任一对作业$i$和作业$j$之间的重叠时间为:

$$OI_{ij} = T + d_i - (T + d_j - s_{ij} - p_j) = d_i - d_j + s_{ij} + p_j \tag{4}$$

序列中第j位置作业的实际拖期为:

$$T_{[j]} = T + \max(\sum_{i=1}^{j} OI_{[i-1][i]}, 0) \tag{5}$$

则式(3)的第二项为:

$$\max_{[j] \in U_k}(T_{[j]}) = T + \max_{[j] \in U_k}(\max(\sum_{i=1}^{j} OI_{[i-1][i]}, 0)) \tag{6}$$

因只需要考虑$T[j]([j] \in U_k)$的最大值,所以应考虑式(6)第二项。令:$D = \min_{i,j \in U_k} OI_{ij}$,定义$OI_{ij}' = OI_{ij} - D, (i,j \in U_k)$,则对于所有的$OI_{ij}' \geq 0 (i,j \in U_k)$,显然,序列中最后一项作业具有$OI_{ij}'$费用的最大增量,最大增量为:

$$\max_{[j] \in U_k}(\max(\sum_{i=1}^{j} OI_{[i-1][i]}, 0)) = \max(\sum_{i=1}^{N_u} OI_{[i-1][i]}' - N_u \times D, 0)$$

最后有:

$$\max_{[j] \in U_k}(T_{[j]}) = T + \max_{[j] \in U_k}(\max(\sum_{i=1}^{j} OI_{[i-1],[i]}, 0)) = T + \max(\sum_{i=1}^{N_u} OI_{[i-1][j]}' - N_u \times D, 0) \tag{7}$$

从式(7)可知,最小化$\max_{[j] \in U_k}(T_{[j]})$只需最小化第二项中的$\sum_{i=1}^{N_u} OI_{[i-1][j]}'$。我们知道,最小化$\sum_{i=1}^{N_u} OI_{[i-1][j]}'$是一个ATSP(非对称旅行售货员问题),并且AP(分配问题)是ATSP的较

好下界。令 ΔT^{AP} 为 ATSP 问题 $\sum_{i=1}^{N_u} OI_{[i-1][i]}'$ 的 AP 下界,则可得 SDSMT 的下界为:

$$Z = \max(T, \max_{j \in U_k}(T_{[j]})) = \max(T, T + \max(\sum_{i=1}^{N_u} OI_{[i-1][i]}' - N_u \times D, 0)) \geqslant T + \max(\Delta T^{AP} - N_u \times D, 0) \quad (8)$$

2.2.2　SDSMT 的上界

好的上界可以改进分枝定界算法的性能,采用由文献[14]提出的改进 Karp-Steel 补偿启发式算法(MKSP)实现上界过程。这一算法区别于原有 KSP 算法的是,不是联结两条最短子路径,而是遍历所有的子路径对,最后采用最好的一个连接方案。和KSP算法一样,改进KSP算法有相同的最坏情况时间复杂度 $O(n^3)$。

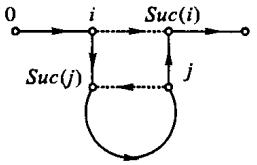


图 2　(i,j) 补偿操作

首先去掉含有作业 0 的连接弧得到基本哈密尔顿(Hamiltonian)路径,然后切断其他的子路径,将其加入全面得到的基本路径中。如图 2,作业 i 和 j 分别位于两个将要进行连接的两个子路径中,其重叠时间的补偿费用为:

$$\Delta(i,j) = OI_{i,Suc(i)} + OI_{j,Suc(j)} - OI_{i,Suc(j)} - OI_{j,Suc(i)} \quad (9)$$

式中 $Suc(i)$ 是子路径中作业 i 的后继作业。

上界过程中将测试所有的连接作业对以找到最好的 $\Delta(i,j)$。在生成搜索树节点时,使用上界过程可能得到一个更好的可行解。

2.3　分枝

分枝过程要指明分枝树中哪一个活动节点要在下一步分解。我们在算法中采用 Best-First 规则,首先分解具有最小下界的活动节点。在每一个节点都将计算相应子问题的下界和上界,并利用优势规则(见第 3 节)去除不需要进一步探测的节点。

2.4　探测

一个节点会在两种情况下被进一步的搜索探测:1)当该节点已经分解并已产生所有的后继子节点时;2)如果该节点的下界大于等于当前已经产生的最佳费用,它将被剔除出活动节点集,这种情况可能发生在节点第一次产生并且下界已算出,或者以后产生新的上界时。

3　SDSMT 的优势规则

优势规则对算法的性能具有重要的作用。通过改变作业序列 S 中作业的位置,可以得到一个新的作业序列 S',比较两个序列的费用可以确定哪一个序列占优势。

$J(K)$:部分序列 K 中的作业集合;$U(K)$:未调度作业集合;$C(K)$:K 中最后一项作业的完成时间;$K|u$:在给定部分序列 K 之后新增加一个作业 u 得到一个新的部分序列;$\sum(K|u)$:含有 $K|u$,和由 $J - J(K|u)$ 中的作业组成的部分调度方案形成的完整调度序列。

通过改变序列 S 中一些作业的位置来构造一个新的序列 S',$[i]$ 仍然指序列 S 中第 i 位置的索引。序列 S 和 S' 中作业的拖期分别为 $T_{[i]}$ 和 $T_{[i]}'(i \in J)$。T_i 和 T_i' 指序列 S 和 S' 的第 i 部分$(i = 0,1,2)$ d 的最大拖期。

定理 2　若 $\exists i,[i] \in J(K)$,$ds = s_{[i-1][i]} + s_{[i][i+1]} + s_{[k-1][k]} - s_{[i-1][k]} - s_{[k][i+1]} - s_{[k-1][i]}$,$s_{[k]u} + ds \geqslant s_{[i]u}$,$s_{[i-1][i]} + s_{[i][i+1]} + p_{[i]} \geqslant s_{[i-1][k]} + s_{[k][i+1]} + p_{[k]}$,$T_{[k]}' \leqslant T_{max}(K|u)$,并且 $T_{[i]} \leqslant T_{max}(K|u)$,那么 $\sum(K|u)$ 被占优。

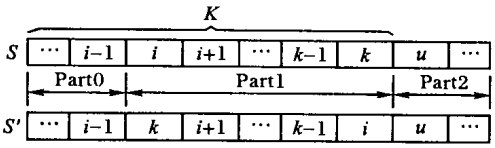


图 3　交换作业$[i]$和作业$[k]$的位置

证明:考虑调度序列 $S = \sum(K|u)$,交换作业$[i]$和$[k]$的位置构造调度序列 S'(图 3),序列 S' 可分为三部分:

在 Part 0:$T_0 = T_0'$,$C_0 = C_0'$。

在 Part 1:根据 $s_{[i-1][i]}s_{[i][i+1]} + p_{[i]} \geqslant s_{[i-1][k]} + s_{[k][i+1]} + p_{[k]}$,$[i+1]$到$[k-1]$位置的作业能够被较早调度。对于位置$[i]$和$[k]$的作业,由于 $T_{[k]}' \leqslant T_{max}(K|u)$ 和 $T_{[i]}' \leqslant T_{max}(K|u)$,有 $T_{max}(K|u) \geqslant T_1'$。

在 Part 2:因为 $s_{[i-1][i]} + s_{[i][i+1]} + s_{[k-1][k]} - (s_{[i-1][k]} + s_{[k][i+1]} + s_{[k-1][i]}) = ds$,且 $s_{[k]u} + ds \geqslant s_{[i]u}$,有 $C_u \geqslant C_u'$。因此,任何开始于作业 u 的未调度作业序列能够在 S' 中较早处理,即 $T_2 \geqslant T_2'$。

最后可得 $T = \max(T_0, T_1, T_2) \geqslant T' = \max(T_0', T_1', T_2')$,所以 $\sum(K|u)$ 被占优。

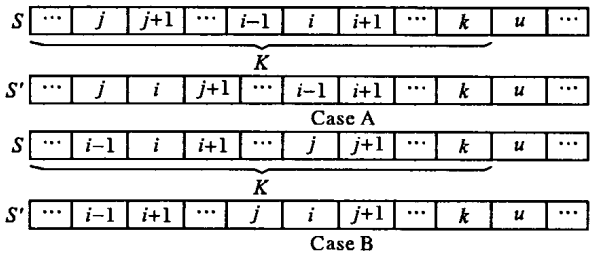


图 4　最大拖期作业$[i]$插入作业后$[j]$

定理 3　设存在 i 有 $T_{[i]} = \max_{[x] \in J(K)} T_{[x]}$,若 $\exists j,[j] \in J(K)$,满足下列条件:

1) $s_{[i-1][i]} + s_{[i][i+1]} + s_{[j][j+1]} \geqslant s_{[i-1][i+1]} + s_{[j][i]} + s_{[i][j+1]}$

2) $C_{[j]}' + s_{[j][i]} + p_{[i]} \leqslant T_{[i]} + d_{[i]}$

3) $T_{[max]}' + \Delta C \leqslant T_{[i]}$

其中:

$$T_{[max]}' = \begin{cases} \max_{x \in [j+1, i-1]} T_x' & Case\ A \\ \max_{x \in [i+1\ j]} T_x' & Case\ B \end{cases}$$

$$\Delta C = \begin{cases} (s_{[j][i]} + p_{[i]} + s_{[i][j+1]}) - s_{[j][j+1]} & Case\ A \\ s_{[i-1][i+1]} - (s_{[i-1][i]} + p_{[i]} + s_{[i][i+1]}) & Case\ B \end{cases}$$

那么 $\sum(K|u)$ 被占优。

证明:考虑调度序列 $S = \sum(K|u)$。把序列 K 中具有最大拖期的作业$[i]$放在作业$[j]$之后,构造新调度方案 S'(图 4)。

考虑未调度作业集合,根据:

$$s_{[i-1][i]} + s_{[i][i+1]} + s_{[j][j+1]} \geqslant s_{[i-1][i+1]} + s_{[j][i]} + s_{[i][j+1]}$$

我们有 $C_u \geqslant C_u'$ 和 $T_u \geqslant T_u'$,因此任何以作业 u 开始的未调度作业序列能在调度 S' 中较早处理。

对作业$[i]$,由 $C'[j] + s_{[j][i]} + p_{[i]} \leqslant T_{[i]} + d_{[i]}$,可得 $C_i \geqslant C_i'$ 和 $T_i \geqslant T_i'$。

对于调度序列中的其他作业,存在两种情况:

Case A:作业$[i]$在作业$[j]$之后

$$T_{[max]}' = \max_{x \in [j+1, i-1]} T_x'$$

$\Delta C = (s_{[j][i]} + p_{[i]} + s_{[i][j+1]}) - s_{[j][j+1]}$

Case B:作业[i]在作业[j]之前

$T_{[\max]}' = \max_{x \in [i+1,j]} T_x'$

$\Delta C = s_{[i-1][i+1]} - s_{[i-1][i]} + (p_{[i]} + s_{[i][j+1]})$

当 $T_{[\max]}' + \Delta C \leqslant T_{[i]}$,可得作业[$j+1$]和作业[$i-1$]之间(*Case A*)或者作业[$i+1$]和作业[$j$]之间(*Case B*)的拖期都小于 $T_{[i]}$。

至此有 $\max_{x \in S}(T_{[x]}) \geqslant \max_{x \in S'}(T_{[x]'})$,即 $\sum(K|u)$ 被占优。

4 应用仿真实验

通常车间加工过程中存在着关键和瓶颈设备,该设备直接影响否能满足合同订单规定的交货期限。当该设备同时需要处理多个规定交货期的合同时,就需要对这些合同涉及的加工作业的先后顺序进行调度,避免出现过大的合同拖期,达到优化经济效益的目的。本文提出的算法将对这样的调度问题进行优化,使可能出现的最大合同拖期达到最小。

本文通过数据仿真方法来对文中提出的算法有效性进行测试。车间进行这样的调度需要如下的输入数据:合同交货期 d_i,作业处理时间 p_i 和 *Setup* 值 s_{ij}。交货期由两个控制参数产生:交货期范围(Due Date Range, DDR)和拖期因子(Tardiness Factor, TF)。设 T' 为处理所有作业总的时间长度,则两个控制参数定义为:

$$DDR = \frac{d_{\max} - d_{\min}}{T'} \qquad TF = 1 - \frac{\sum d_j}{T'}$$

DDR 控制交货期的分布范围,*TF* 定义交货期相对于平均交货期的分布情况,指明了作业交货期的松紧程度。在本实验中提出了一种新的 *Setup* 比例因子(Setup Proportion Factor, SPF),*SPF* 说明了 *Setup* 对处理时间的相对比例情况。

处理时间和 *Setup* 值满足均匀分布,$p \in [10, 100]$,$s \in [10, SPF \times p_{\max}]$。交货期在之内(假定不允许交货期为负)。然而在确定调度序列之前,不能按最优解序列来严格确定 S_{ij},用平均 *Setup* 值来近似,则有 $T' = \sum(p_j + average(s))$。

表 1 生成随机实例的控制参数

问题规模	*TF*&*DDR*	*SPF*
{10,15,20,25,30}	{0.1,0.5,1.0}	{0.2,0.6,1.0,1.4,1.8}

根据表 1 给出的控制参数随机产生 5×225 组数据实例进行计算实验。算法 VC++ 6.0 实现,在 Pentium Ⅱ CPU 600MHz、256M 内存的 PC 机上运行。

表 2 算法总体性能

问题规模	运行时间/s	扩展节点	探索节点	存储节点	可解率(%)
10	1.461	685.47	131.82	43.68	100
15	84.513	24630	3586.5	1158	97.333
20	344.65	61395	9697.1	5528	68.889
25	520.97	19521	4142.7	3201.5	46.667
30	593.15	13514	3964.3	3436.5	37.333

将算法用于求解随机产生所有数据实例(共 5×225 个),在表 2 里列出了实验的平均结果。每一个实例的 CPU 运行时间设定为 15 分钟。从表 2 可以看出有 10 项和 15 项作业的平均求解时间少于 85s。随着作业量的增加,可解实例百分比逐步减少,如该算法能够解决 25 个作业问题的 46%、30 个作业问题的 37%。因为随着作业量的增加,下界、上界和优势规则计算逐渐困难。30 个作业比 25 个作业实例的节点数少是由于每个作业 CPU 计算时间的限制。

5 结语

序列相关 Setup 满足交货期的调度问题是工业应用和科学研究领域的重要问题,本文针对该类型调度问题研究了最小化最大拖期的分枝定界算法。在算法的研究中提出和证明了下界、上界方法和一些优势规则。实验结果证明了本文所提出算法的有效性,说明了本文的下界、上界和优势规则对算法性能的提高具有实质的意义。采用本文提出的算法,能够求解 100% 的 10 个作业规模的实例、97% 的 15 个作业规模的实例和至多 37.33% 的 30 个作业规模的实例。

参考文献:

[1] GUPTA SK. N jobs M machines job - shop problems with sequence dependent setup times[J]. International Journal of Production Research, 1982, 20: 643 – 656.

[2] KRAJEWSKI LJ, KING BE, RITZMAN LP, *et al*. Kanban, MRP and shaping the manufacturing environment[J]. Management Science, 1987, 33(1): 39 – 57.

[3] KIM SC, BOBROWSKI PM. Impact of sequence - dependent setup times on job shop scheduling performance[J]. International Journal of Production Research, 1994, 32: 1503 – 1520.

[4] BAKER KR. Introduction to sequencing and scheduling[M]. Wiley New York, 1974.

[5] OVACIK IM, UZSOY R. Rolling horizon procedures for dynamic parallel machine scheduling with sequence-dependent setup times[J]. International Journal of Production Research, 1995, 33(11): 3173 – 3192.

[6] SUN XQ, NOBLE JS, KLEIN CM. Single-machine scheduling with sequence dependent setup to minimize total weighted squared tardiness[J]. IIE Transaction, 1999, 31(2): 113 – 124.

[7] RAGATZ GL. A branch-and-bound method for minimum tardiness sequencing on a single processor with sequence dependent setup times[A]. Proceedings: 27 twenty-fourth annual meeting of the Decision Sciences Institute[C]. 1993. 1375 – 1377.

[8] TAN KC, NARASINMHAN R, RUBIN PA, *et al*. A comparison of four methods for minimizing total tardiness on a single processor with sequence dependent setup times[J]. Omega, 2000, 28(3): 313 – 326.

[9] RUBIN PA, RAGATZ GL. Scheduling in a sequence dependent setup environment with generic search[J]. Computers & Operations Research, 1995, 22(1): 85 – 99.

[10] TAN KC, NARASIMHAN R. Minimizing tardiness on a single processor with sequence-dependent times: a simulated annealing approach[J]. Omega, 1997, 25(6): 619 – 634.

[11] LEE YH, BHASKARAN K, PINEDO M. A heuristic to minimize the total weighted tardiness with sequence-dependent setup[J]. IIE Transaction, 1997, 21(9): 45 – 52.

[12] PARK Y, KIM S, LEE YH. Scheduling jobs on parallel machines applying neural network and heuristics rules[J]. Computers & Industrial Engineering, 2000, 38(1): 189 – 202.

[13] GAGNE C, PRICE WL, GRAVEL M. Scheduling a single machine with sequence dependent setup times using ant colony optimization [R]. Work report 2001 – 2003.

[14] GLOVER G, GUTIN G, YEO A, *et al*. Construction heuristics for the asymmetric TSP[J]. European Journal of Operation Research, 2001, 129: 555 – 568.