文章编号：0253-374X(2020)04-0610-10

DOI：10. 11908/j. issn. 0253-374x. 19179

考虑排班的人力资源投入问题的建模与优化

陆志强，许则鑫，任逸飞

（同济大学 机械与能源工程学院，上海 201804）

摘要：结合实际生产或项目中的排班情况，提出考虑排班的人力资源投入问题。针对该问题建立了以最小化人力资源投入为目标的数学模型。根据资源投入量与排班约束的性质，将原问题数学模型简化，证明简化后问题的数学模型与原问题最优解一致，并通过CPLEX软件求解过程，说明简化后的数学模型在求解速度上表现出很大的优越性。对于大规模问题，由于排班约束会导致班次间资源占用，使用传统任务列表编码方式难以获得较优的解。为此，提出了一种新型编码方式的遗传算法。该算法采用对作业延迟时间进行编码的方式，对作业开始时间进行搜索。为了提升算法的局部搜索能力，对作业延迟时间和开始时间进行局部优化。最后，通过数值实验与CPLEX和文献的算法比较，表明该算法的有效性。

关键词：资源投入；排班；延迟时间；遗传算法

中图分类号：F273　　**文献标志码**：A

Modeling and Optimization of Resource Investment Problem Based on Employee-Timetabling

LU Zhiqiang, *XU Zexin*, *REN Yifei*

(School of Mechanical Engineering, Tongji University, Shanghai 201804, China)

Abstract: In this paper, a resource investment problem is addressed based on employee-timetabling and according to the employee timetabling in practical production systems. A mathematical model aimed at minimizing resource investment is proposed for this problem. In order to solve this problem more efficiently, the mathematical model of the original problem is simplified according to the resource investment and the properties of employee-timetabling constraints. The model proposed is proved to have the same optimal solution as the original mathematical model, and its great advantage in solving speed is verified through the CPLEX software solution process. Meanwhile, for the large-scale problem, as the constraints of employee-timetabling lead to resource occupancy between shifts, it is difficult to obtain a good solution by using the traditional activity list encoding method. A genetic algorithm with a new coding method is designed in this paper, which encodes the job delay time to search the starting time of the job. Moreover, two local optimization methods are proposed which can optimize the delay time and the starting time of jobs to improve the solution obtained by using the genetic algorithm. A comparison of the numerical experiments with CPLEX and the literature demonstrates the validity of the algorithm.

Key words: resource investment; employee-timetabling; delay time; genetic algorithm

资源投入问题(resource investment problem，RIP)是一种经典的项目调度问题，其目标是优化人力资源的投入，使其成本最小化。然而人力资源作为一种重要的生产资源，在实际生产中，如飞机移动装配线，是以排班的形式进行生产作业的，因此资源的投入会受排班的影响。此时，调度的关键在于合理地安排作业和排班，从而确定每个班次的作业调度计划和资源投入情况，使得总体的人力资源投入最小化。在本文中，这一问题被称之为考虑人力资源排班的人力资源投入问题(resource investment problem based on employee-timetabling，RIP-ET)，其本质上是资源投入和人力资源排班(employee timetabling problem, ETP)的整合问题。与传统的RIP问题相比，RIP-ET问题有以下几项难点：RIP-ET需要考虑工人排班的约束；RIP-ET不仅要决策作业的开始时间，而且还要决策作业投入的每个工人；对于RIP-ET问题，每个班次的资源投入是可变

收稿日期：2019-05-10
基金项目：国家自然科学基金(61473211，71171130)
第一作者：陆志强(1968—)，男，教授，博士生导师，工学博士，主要研究方向为物流与供应链建模与优化等。
E-mail：zhiqianglu@tongji. edu. cn

论文拓展介绍

的，由于排班约束，班次之间存在资源占用情况；与RIP问题优化目标为资源峰值不同，该问题的目标是降低投入整个项目的人数。由于RIP-ET问题与传统RIP的决策范围与约束范围不同，现有的模型和算法无法适用，因此，对RIP-ET问题的建模与算法研究具有重要的理论与实际意义。

RIP是从经典的项目调度问题（resource constrained project scheduling problem，RCPSP）中衍生而来，问题的目标是在给定工期的情况下求解资源投入的最小值。Möhring[1]首先提出了资源投入问题，证明了该问题是NP-hard问题，同时证明了RIP与单模资源约束项目调度问题（single mode resource constrained project scheduling problem，SMRCPSP）的对偶关系。Demeulemeester[2]将RIP转化为多个SMRCPSP，提出了MBA（minimum bounding algorithm）精确算法。Rangaswamy[3]提出了一个分支定界算法，进一步提高了算法的求解效率。由于精确算法在求解大规模问题上具有局限性，很多学者在此问题上设计了不同的启发式算法来进行求解。Yamashita等[4]将RIP转化为RCPSP问题，通过基于Scatter Search的元启发式算法求解该问题；Shadrokh等[5]采用遗传算法求解带有延迟惩罚的资源投入问题，分别对作业优先级和资源容量进行编码。Ranjbar等[6]首次在没有将RIP问题转化为RCPSP的基础上，通过路径重连和遗传算法来求解该问题。但是其算法可能会产生优先级不可行的作业列表，也会产生算法效率问题，而且在对作业进行调度时，并没有考虑到当前选择对全局资源投入的影响。Zhu等[7]在此基础上提出了一种多启动迭代搜索算法（multi-start iterative search method），该算法有效地提高了RIP解的质量。针对资源投入问题，许多学者还在此基础上进行了扩展性的研究[8-10]。本文提出的考虑排班的人力资源投入问题就是其中的一种。

人力资源排班是将具有特殊技能的人力资源分配到特定的班次，以满足特定时间段的服务需求。随着人力资源排班用于各种领域，人力资源排班的形式也呈现多样化。由于工作内容的不同，Baker[11]首次提出了人力资源分配的分类方法，将其分为轮班调度、日程安排和行程安排三类。Kletzander等[12]针对排班问题中的各种约束提出了一种适用性框架，并在此框架上提出了一种通用的模拟退火算法。但是在现有的文献中，大多只关注于人力资源的轮班顺序或工作时间表，很少将人力资源调度与其他调度（例如机器调度、车辆调度、生产调度、手术室调度等）结合在一起研究。然而，在一个实际的生产活动中，人力资源的排班往往需要与生产项目的调度结合起来统筹考虑，正如Bergh等[13]提到的，这也是未来一个主要的研究方向。Daniels等[14]假定每个作业必须由作业人员操作机器才能执行，将流水作业调度与人力资源排班整合起来。Drezet等[15]在资源受限的项目调度环境中考虑了人力资源排班，并提出一种预测算法来处理该问题。Artigues等[16]和Guyon等[17]采用了不同的精确算法对车间调度和人力资源排班的整合问题进行了求解。Smet等[18]将作业调度与排班问题结合起来，将任务与班次同时分配给员工。Eeckhout等[19]提出了一种新的迭代局部搜索方法，解决资源受限项目调度中的人员配置问题。由于车间调度、作业调度与资源投入项目调度具有一定的相似性，这对研究项目调度与人力资源排班的整合问题具有借鉴意义。不过，目前还没有关于RIP与人力资源排班的整合问题的研究。

RIP作为RCPSP的一个衍生问题，也是一种经典的项目调度问题，经常应用于实际项目决策中，它本身也对RIP-ET的研究具有很重要的意义。本文研究的RIP-ET除了考虑RIP的特性之外，还需要考虑人力资源排班约束，具有较高的复杂度，精确算法在求解这类大规模的问题时效率较低。遗传算法由于具有较好的全局搜索能力，在RIP中得到了很好的使用[20-23]。由于排班约束会导致班次间的资源抢占，故采用作业优先级和资源编码的方式对于该问题无法取得较优解。因此，本文采用了对作业开始时间进行搜索的遗传算法。

在分析现有文献中人力资源投入问题与人力资源排班问题和遗传算法的基础上，本文以最小化人力资源投入作为目标，从实际生产活动的决策需求出发，建立了RIP-ET的数学模型。通过分析，将该整合问题拆分为了人力资源投入和人力资源排班问题，并且设计了一种新型编码方式的遗传算法，通过对作业延迟时间进行编码的方式，对作业的开始时间进行全局搜索，此外还对作业延迟时间和开始时间进行局部优化。

1　问题描述及建模

记一个项目由若干项作业构成，$j=\{1,2,3,\cdots,n\}$为项目的作业集合。其中，1、n为虚任务不占用时间和资源，$j\in J$为作业编号，其标准作

业时间为t_j;全部作业共需要K种人力资源进行作业,资源集合$R=\{1,2,\cdots,K\}$,$k\in R$为人力资源种类编号,其中第k种人力资源对应的单价为c_k;定义$m\in M_k$为第k种人力资源中的工人编号,集合$M_k=\{1,2,\cdots,M\}$表示第k种人力资源的集合,同时假设同类资源下的所有资源不具有差异性。

项目的总工期为$\overline{T}$,每个班次8 h,将项目分为U个班次,定义班次集合为$W=\{1,2,\cdots,U\}$,$w\in W$为班次编号,对于每个工人,规定在连续的3个班次中只能工作1个班次。假设作业从开始到结束不得中断,当班次结束时,若当前作业未完成,不允许工人加班,必须换上相同数目的同类工人继续该工作。在实际的人力资源排班中,很多情况下使用的是三班制排班。在文本中,不妨也使用三班制排班,假设每个班次8 h,将1 d分为3个班次。

P_j为作业j的紧前作业的集合,$i\in P_j$表示作业i为作业j的紧前作业;r_{jk}表示作业j对第k种人力资源的需求量。对时间进行离散化,$d\in D$为离散时间点,$D=\{1,2,\cdots,T\}$,T为项目的实际完工时间。

定义决策变量如下:

x_{jd}——0,1变量,作业j在d时刻处于执行状态为1,否则为0;

h_{jdkm}——0,1变量,第k种人力资源中的第m号工人在时间d执行作业j则为1,否则为0。

定义中间变量如下:

λ_{wkm}——0,1变量,第k种人力资源中的第m号工人在第w班次中进行作业则为1,否则为0;

ℓ_{km}——0,1变量,第k种人力资源中的第m号工人投入了整个项目则为1,否则为0。

RIP-ET问题P_1的数学模型如下:

目标函数为

$$\min A=\sum_{k\in R}\left(c_k\sum_{m\in M_k}\ell_{km}\right) \tag{1}$$

传统RIP约束为

$$\sum_{d\in D}x_{jd}=t_j,\ \forall j \tag{2}$$

$$t_p x_{jd}\leqslant\sum_{i=1}^{d-1}x_{pi},\ \forall p\in P_j,\forall j,d \tag{3}$$

$$0\leqslant dx_{jd}\leqslant\overline{T},\ \forall j,d \tag{4}$$

$$t_j x_{jd}-t_j x_{j(d+1)}+\sum_{i=d+2}^{T}x_{ji}\leqslant t_j,\ \forall j,d \tag{5}$$

$$\sum_{m\in M_k}h_{jdkm}\geqslant r_{jk}x_{jd},\ \forall j,d,k \tag{6}$$

排班约束为

$$\max_{d\in[8(w-1),8w]}\sum_{j\in J}h_{jdkm}\leqslant\lambda_{wkm},\ \forall k,w,m \tag{7}$$

$$\lambda_{wkm}+\lambda_{(w+1)km}+\lambda_{(w+2)km}\leqslant 1,\ \forall k,w,m \tag{8}$$

$$\lambda_{wkm}\leqslant\ell_{km},\ \forall k,w,m \tag{9}$$

决策变量可行域为

$$x_{jd},h_{jdkm},\lambda_{wkm},\ell_{km}=\{0,1\},\ \forall j,k,d,m \tag{10}$$

其中,式(1)为最小化人力资源的投入;式(2)表示作业的执行时间等于作业工期;式(3)为优先关系约束;式(4)为项目工期约束;式(5)表示任务一旦开始就不能中断;式(6)为资源约束;式(7)为决策变量h_{jdkm}与中间变量λ_{wkm}的关系;式(8)为人力资源排班约束,单个个体的人力资源在连续3个班次中只能工作1个班次;式(9)表示中间变量λ_{wkm}与中间变量ℓ_{km}的关系;式(10)表示定义所有决策变量中间变量的可行域。

2 问题分析与简化

上节给出问题P_1的数学模型,是对作业调度与人力资源排班进行同时决策,需要在满足排班约束下求出人力资源投入的最小值。然而,通过分析发现,由于同种人力资源中每个个体之间不存在差异性,因此对于k类人力资源,总有:

性质1 考虑排班约束下的总投入总是等于不考虑排班约束下最大的连续3个班次的总投入。即$\max\limits_{\forall w\in W}(l_{kw}+l_{k(w+1)}+l_{k(w+2)})=\sum\limits_{m\in M_k}\ell_{km}$,其中$l_{kw}$为第$k$种人力资源在班次$w$的投入数量。

证明 假设所有作业的开始和结束时间已经确定,对于k类人力资源,假设有$l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}=\max\limits_{\forall w\in M}(l_{kw}+l_{k(w+1)}+l_{k(w+2)})$,其中$w'$为一个特定的班次,令$m_k=\sum\limits_{m\in M_k}\ell_{km}$。首先证明$l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}$的人数在满足工人休息的情况下能够满足作业要求,即$l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}\geqslant m_k$。由已知可得:$l_{kw'}\geqslant l_{k(w'+3)}$,表示对于$w'+3$班次对人力资源的需求可以由$w'$班次释放的工人来满足,从而可以推理得到$l_{k(w'+4)}\leqslant l_{kw'}+l_{k(w'+1)}-l_{k(w'+3)}$,$l_{k(w'+5)}\leqslant l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}-l_{k(w'+3)}-l_{k(w'+4)}$,由数学归纳法可以得到,对于$w\in W$,$l_{k(w)}\leqslant l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}-l_{k(w-1)}-l_{k(w-2)}$,即$l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}\geqslant m_k$,得证。第二步,证明$l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}\leqslant m_k$,使用反正法证明。若$l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}>m_k$,由于不等式两边都是正整数,可以假设$m_k=l_{kw'}+l_{k(w'+1)}+(l_{k(w'+2)}-1)$,此时在$w'+2$

班次中由于人力资源不足，无法进行作业，则$l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}>m_k$是错误的，即$l_{kw'}+l_{k(w'+1)}+l_{k(w'+2)}\leqslant m_k$，得证。从而可得$\max\limits_{\forall w\in M}(l_{kw}+l_{k(w+1)}+l_{k(w+2)})=\sum\limits_{m\in M_k}\ell_{kw}$。

根据性质1，发现求解原问题P_1可以转化为：先根据项目调度计算每个班次需要投入人力资源的数量，进而计算最终的人力资源投入。本文将这个问题称之为问题P_2。得到最优的人力投入之后，再根据每个班次的需求量对人力资源进行排班，这个问题称之为问题P_3。因此，对原问题P_1的求解，变成了求解问题P_2与问题P_3。针对问题P_2，建立如下的数学模型。

定义决策变量如下：

x_{jd}——0，1变量，作业j在d时刻处于执行状态为1，否则为0；

y_{kd}——整数变量，第k种人力资源在时间d的投入数量。

定义中间变量如下：

l_{kw}——整数变量，第k种人力资源在班次w的投入数量。

目标函数：

$$\min A=\left(c_k\max_{\forall w\in W}(l_{kw}+l_{k(w+1)}+l_{k(w+2)})\right) \tag{11}$$

$$\sum_{d\in D}x_{jd}=t_j,\ \forall j\in J \tag{12}$$

$$t_p x_{jd}\leqslant\sum_{i=1}^{d-1}x_{pi},\ \forall p\in P_j,\ \forall j,d \tag{13}$$

$$0\leqslant dx_{jd}\leqslant\overline{T},\ \forall j,d \tag{14}$$

$$t_j x_{jd}-t_j x_{j(d+1)}+\sum_{i=d+2}^{T}x_{ji}\leqslant t_j,\ \forall j,d \tag{15}$$

$$\sum_{j\in J}r_{jk}x_{jd}=y_{kd},\ \forall k,d \tag{16}$$

$$\max_{d=(8(w-1),\min(T,8w))}y_{kd}=l_{kw},\ \forall k,w \tag{17}$$

$$x_{jd}=\{0,1\},y_{kd},l_{kw}\in\{0,1,2,\cdots\},\ \forall j,d,k,w \tag{18}$$

其中，式(11)为最小化人力资源投入；式(12) 表示作业的执行时间等于作业工期；式(13)为优先关系约束；式(14)为工期约束；式(15)表示任务一旦开始就不能中断；式(16)为资源约束；式(17)表示中间变量l_{kw}与决策变量y_{kd}之间的关系；式(18)表示决策变量x_{jd}、y_{kd}和中间变量l_{kw}的定义域。

问题P_3是对工人的工作班次进行分配，在不考虑资源均衡的情况下，并没有目标函数。问题的决策变量和模型如下：

λ_{wkm}——0，1变量，第k种工人中的第m号工人在第w班次中工作则为1，否则为0。

$$\lambda_{wkm}+\lambda_{(w+1)km}+\lambda_{(w+2)km}\leqslant 1,\ \forall k,w,m \tag{19}$$

$$\sum_{m\in M_k}\lambda_{wkm}\leqslant l_{kw},\ \forall k,w \tag{20}$$

其中，式(19)表示单个工人在连续3个班次内只能工作1个班次；式(20)表示决策变量λ_{wkm}与问题P_2中间变量l_{kw}的关系。

为了验证问题简化的有效性，使用CPLEX对测试问题库中的小算例进行求解，来对比两种建模方式求解的速度。设定资源种类$K=4$，任务工期$\overline{T}$设置为由关键链方法(critical path method，CPM)得出的项目工期的1.2倍。表1和表2分别表示在10 jobs、14 jobs下两种建模方式求解结果。其中，jobs表示项目的作业数量。A_1和A_2分别为对问题P_1和转化后问题P_2、P_3计算所得的目标函数值。T_1为求解P_1花费的时间，T_2为求解P_2和P_3一共花费的时间，设置算法的最大运行时间为3 600 s。表2中，A列中“—”表示CPLEX没有求出最优解，T列中“—”表示CPLEX运行时间超过3 600 s自动停止。

表1　10 jobs 实验结果

Tab.1　Scheduling result of 10 jobs

算例	A_1	A_2	T_1/s	T_2/s
1	122	122	21.948	0.256
2	104	104	427.027	0.595
3	143	143	254.443	0.681
4	118	118	29.963	0.312
5	166	166	135.183	0.601
6	131	131	52.430	0.729
7	109	109	42.087	0.424
8	158	158	73.634	0.614
9	154	154	163.570	0.984
10	106	106	28.313	0.492

表2　14 jobs 实验结果

Tab.2　Scheduling result of 14 jobs

算例	A_1	A_2	T_1/s	T_2/s
1	160	160	69.245	0.682
2	149	149	72.540	0.288
3	—	151	—	3.825
4	215	215	140.438	1.286
5	118	118	—	2.342
6	165	165	334.535	0.856
7	123	123	36.262	0.487
8	177	177	1 135.165	2.598
9	175	175	297.973	1.123
10	182	182	79.260	0.836

从实验对比可以发现，将问题转换之后，使用CPLEX求解的速度显著提升，因此将原问题转换为问题P_2与问题P_3在小规模的求解中是很有必要的。

3 算法设计

针对RIP-ET的特点,本文设计了一种对作业开始时间进行搜索的遗传算法,可以有效避免RIP在考虑工人排班时带来的难点。采用这种编码方式可以通过解码直接求出各个作业的开始时间,进而根据式(11)求出人力资源的投入量,解码过程中并不需要通过控制可用资源和作业的优先级来决策作业的开始时间。

Najafi等[24]通过CPM求出所有作业的最早开始时间,提出了对作业最早开始时间的浮动时间的编码方法。这种编码方法解决了基于现金折现的RIP,同时还提出了如何在解生成中避免出现不可行解。对作业开始时间的浮动时间进行编码,虽然可以有效避免排班带来的资源抢占问题,但是在生成下一代解时,变动性较小,容易陷入局部最优解。因此,本文采用对作业延迟时间进行编码的方式,并对作业延迟时间和开始时间同时进行局部优化。基于该问题设计了新的遗传算法,有效地解决了该问题。对比文献[24]的编码方式,这种编码方式能够有效地避免解陷入局部最优。需要强调的是,下文算法优化的是总人力资源投入,即针对问题P_2所设计的算法,由于问题P_3是一个简单问题,因此求解不加以赘述。

3.1 染色体编码

本文采用实数编码方式,对每个作业的延迟开始时间进行编码,编码长度为n,分别对应每一个作业的延迟开始时间,如图1所示。通过确定每个作业的延迟开始时间调度项目中的所有作业。

0	2	1	1	0	1	0	0

图1 作业延迟时间编码

Fig. 1 Coding with work delay time

关于延迟开始时间的相关定义如下:

定义1 在作业1~(i-1)已经调度的基础上,作业i的延迟时间等于作业i的最早开始时间$\max\limits_{j\in P_i}(t_{\mathrm{FT}_j})$与实际开始时间$t_{\mathrm{ST}_i}$的差值,即

$$D[i]=t_{\mathrm{ST}_i}-\max_{j\in P_i}(t_{\mathrm{FT}_j}) \tag{21}$$

式中:t_{ST_i}、t_{FT_i}表示作业i实际的开始、结束时间。

定义2 在所有作业的延迟时间D给定的情况下,作业i的最早开始时间t_{ES_i}等于其最晚紧前任务的结束时间,即令$D[i]=0$时,作业i的开始时间,即

$$t_{\mathrm{ES}_i}=\max_{j\in P_i}(t_{\mathrm{FT}_j}) \tag{22}$$

定义3 在所有作业的延迟时间D给定的情况下,作业i的最晚开始时间t_{LS_i}等于令最后一个任务的结束时间为项目工期$\overline{T}$,倒序所求出的作业i的开始时间,即

$$t_{\mathrm{LS}_i}=\min_{j\in S_{\mathrm{succ}(i)}}(t_{\mathrm{ST}_j}-D[j]) \tag{23}$$

式中:$S_{\mathrm{succ}(i)}$表示作业i紧后任务的集合。

当所有的延迟时间D都为0时,或者未赋值时,所有作业的最早(晚)开始时间t_{ES_i}(t_{LS_i})为不考虑资源使用的情况下,由关键链方法所求得的最早(晚)开始时间,当某个作业延迟时间D给定时,其他作业的最早(晚)开始时间也会发生改变。

对于图2所示的例子,假设作业的延迟时间为(0,2,1,1,0,1,0,0),可以得到如图3所示的项目调度。

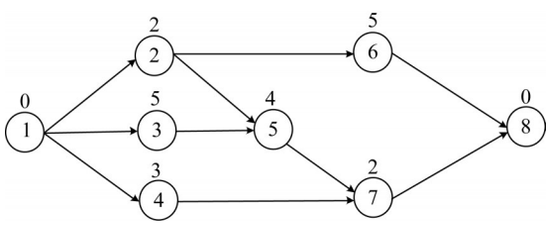

图2 项目的AON网络的一个实例

Fig.2 An example of AON network of a project

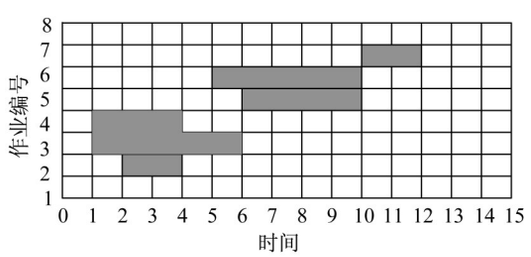

图3 项目调度时间图

Fig.3 Scheduling time map of a given project

假设此项目的工期为15,通过逆向调度,将截止时间视为开始时间,结束任务作为开始任务,可得到项目逆向调度图,如图4所示。图中作业的开始时间,即为上文所定义的最晚开始时间。

于是,项目中所有作业的最早开始时间、实际开始时间等见表3。

此外,考虑到解的可行性,对于一组延迟时间D,它必须满足:对于任意作业i,该作业的延迟时间不能超过该作业的最晚开始时间与最早开始时间之差,即

$$D[i]\leqslant t_{\mathrm{LS}_i}-t_{\mathrm{ES}_i},\ \forall i\in J \tag{24}$$

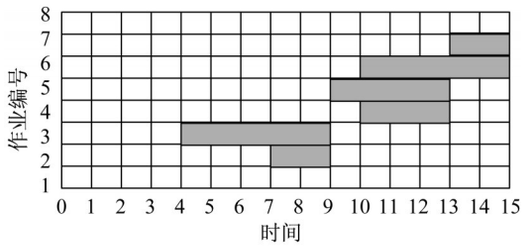

图4　项目逆向调度时间图

Fig.4　A project time map with reverse scheduling

表3　项目中各作业的时间

Tab.3　Time of all works

作业	延迟时间	最早开始	实际开始	最晚开始	最晚结束
1	0	0	0	3	3
2	2	0	2	7	9
3	1	0	1	4	9
4	1	0	1	10	13
5	0	6	6	9	13
6	1	4	5	10	15
7	0	10	10	13	15
8	0	12	12	15	15

对于染色体的评估,可以根据目标函数计算染色体的适应值。本文所采用的对作业延迟时间编码的方式,可以有效地搜索所有作业可能的开始时间,通过确定所有作业的开始时间,来确定每个班次需要分配的资源。

3.2　初始解生成

采用正向和逆向两种方法生成初始的染色体群。根据上文所提到的,要保证解的可行性,基因$D[i]\in[0,t_{\mathrm{LS}_i}-t_{\mathrm{ES}_i}]$。于是,需要先求得作业$i$的最早开始时间与最晚开始时间,然后再给基因$D[i]$随机赋值。为了防止先生成的$D[i]$过大,导致后续$D[i]$的取值受限。本文采用如图5所示的三角分布对基因进行随机赋值。下文中采用三角分布都是这个原因。

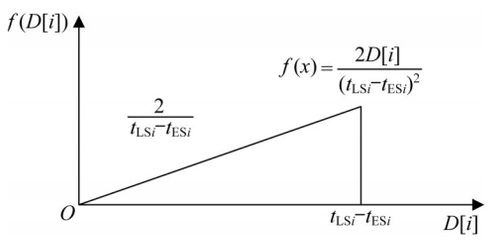

图5　产生基因i的三角分布图

Fig.5　Triangular distribution for gene *i* generation

3.2.1　正向生成染色体

从作业1到作业n顺序生成每个基因的值。算法步骤如下:

步骤1　初始化$D[i]=0,\forall i\in J$,计算所有的t_{ES_i}、t_{LS_i}。

步骤2　令$i=1$,在$[0,1]$中随机选取一个小数θ,$D[1]=\lfloor\theta^2\cdot t_{ES_1}\rfloor$。

步骤3　$i=i+1$,计算作业i的最早开始时间t_{ES_i},在$[0,1]$中随机选取一个小数θ,令$D[i]=\lfloor\theta^2\cdot(t_{LS_i}-t_{ES_i})\rfloor$。

步骤4　若$i=n$,则停止运算,否则转步骤3。

3.2.2　逆向生成染色体

从作业n到作业1逆向生成每个基因的值,算法的具体操作与正向生成染色体的操作类似。

3.3　染色体交叉

选取父代染色体P_1、P_2进行交叉,生成子代染色体C_1、C_2。子代染色体中的一部分直接复制父代的染色体,另外一部分通过2条父代染色体交叉得到。

定义4　在给定的调度基础上,作业延迟时间的可减少值等于作业的实际开始时间减去作业最早的开始时间,即

$$N[i]=t_{\mathrm{ST}_i}-t_{\mathrm{ES}_i} \tag{25}$$

定义5　在给定的调度基础上,作业延迟时间的可增加值等于作业的最晚开始时间减去作业的实际开始时间,即

$$F[i]=t_{\mathrm{LS}_i}-t_{\mathrm{ST}_i} \tag{26}$$

染色体交叉的算法步骤如下:

步骤1　令$C_1[i]=P_1[i]$,$C_2[i]=P_2[i]$,$\forall i\in J$。

步骤2　对于父代P_1、P_2,计算所有作业延迟时间的可减少值和可增加值,记为N_{P_1}、F_{P_1}、N_{P_2}、F_{P_2}。

步骤3　在范围$[2,n-1]$内随机生成一个整数q。

步骤4　在范围$[0,1]$随机生成一个小数ρ。若$\rho>0.5$,转步骤5,反之转步骤9。

步骤5　令$i=q$。

步骤6　令$i=i+1$,对于子代C_1、C_2,计算作业i延迟时间的可减少值和可增加值,记为$N_{C_1}[i]$、$F_{C_1}[i]$、$N_{C_2}[i]$、$F_{C_2}[i]$。

步骤7　染色体的交叉步骤,根据上面得到的结果不同,可以分为几下几种情形。

情形1　当$N_{P_2}[i]>0$,$F_{P_2}[i]>0$,即父代P_2中作业i的可减少与可增加时间都大于0,令

$$D_{C_1}[i]=\left\lfloor F_{P_2}/(N_{P_2}[i]+F_{P_2}[i])\times(F_{C_1}[i]+N_{C_1}[i])\right\rfloor$$

情形2　不满足情形1,并且$N_{C_2}[i]>0$,

$F_{C_2}[i]>0$时。此时与C_2交叉。即令

$$D_{C_1}[i]=\left\lfloor F_{C_2}/(N_{C_2}[i]+F_{C_2}[i])\times(F_{C_1}[i]+N_{C_1}[i])\right\rfloor$$

情形3 即不满足情形1与情形2,此时$D_{C_1}[i]$取$[0,N_{C_1}[i]+F_{C_1}[i]]$中的随机一个整数。

按照生成$D_{C_1}[i]$的交叉操作生成子代C_2的第i个基因$D_{C_2}[i]$。

步骤8 若$i=n$停止运算,否则转步骤6。

步骤9 令$i=q$。

步骤10 对于子代C_1、C_2,计算作业i延迟时间的可减少值和可增加值,记为$N_{C_1}[i]$、$F_{C_1}[i]$、$N_{C_2}[i]$、$F_{C_2}[i]$。

步骤11 重复步骤7的操作。

步骤12 若$i=1$则停止运算,否则令$i=i-1$转步骤10。

3.4 染色体变异

选取染色体C进行变异,随机选择该染色体一部分的基因进行变异,其他部位的基因保持不变。与染色体的生成相似,计算作业i的$F[i]$、$N[i]$。采用三角分布对$D[i]$重新赋值。

染色体变异的算法步骤如下:

步骤1 从$[1,n-1]$随机选择两个整数q_1、q_2,其中$q_2>q_1$。

步骤2 从$[0,1]$随机选取一个小数ρ,若$\rho>0.5$,转步骤3,否则转步骤7。

步骤3 令$i=q_1$。

步骤4 $i=i+1$,对于染色体C,计算所有任务延迟时间的可减少值和可增加值,记为$N_C[i]$、$F_C[i]$。

步骤5 在$[0,1]$中随机生成一个小数θ,$D[i]=\theta^2\cdot(F_C[i]+N_C[i])$。

步骤6 若$i=q_2$,运算结束,否则转步骤4。

步骤7 令$i=q_2$。

步骤8 对于染色体C,计算作业i延迟时间的可减少值和可增加值,记为$N_C[i]$、$F_C[i]$。

步骤9 在$[0,1]$中随机生成一个小数θ,$D[i]=\left\lfloor(1-\theta^2)\cdot(F_C[i]+N_C[i])\right\rfloor$。

步骤10 若$i=q_1$,则停止运算,否则令$i=i-1$,转步骤8。

3.5 局部优化

为了提高解的适应性,本节采用两种局部优化方法对作业的延迟时间和作业的开始时间进行优化。

3.5.1 对作业延迟时间优化

对于一个已经给定的调度,每个作业都有一个延迟开始时间,改变作业延迟时间可以改变目标函数的值。例如,对于图3所示的项目调度时间图,作业的延迟时间为(0,2,1,1,0,1,0,0),作业2延迟时间的可增加值和可减少值分别为5和2。减少或者增加作业2的延迟时间,可以得到一个新的目标函数值。

该局部优化的算法步骤如下:

步骤1 令$i=1$,计算当前调度下目标函数值为A。

步骤2 计算作业i延迟时间的可增加值和可减少值$F[i]$、$N[i]$。

步骤3 从$D[i]\in[0,F[i]+D[i]]$中选择使得目标函数值最优的$D[i]$,更新$D[i]$,A。若$i=n$,停止运算,否则令$i=i+1$转步骤2。

3.5.2 对作业开始时间优化

对于一个给定的调度,提前或推迟作业的开始时间可以改变目标函数的值,通过对每个任务开始时间的优化可以优化整个项目的资源投入。

定义6 表示作业i在不影响其他作业的基础上最早的可开始时间,等于所有紧前任务最晚的完成时间,即

$$t_{\mathrm{es}_i}=\max_{j\in P_i}(t_{\mathrm{FT}_j}) \tag{27}$$

式中:t_{FT_j}表示作业的实际完成时间。

定义7 t_{ls_i}表示作业i在不影响后续作业开始时间的基础上最晚可开始的时间,等于所有紧后任务最早的开始时间减去作业i的持续时间,即

$$t_{\mathrm{ls}_i}=\max_{j\in S_{\mathrm{succ}(i)}}(t_{\mathrm{ST}_j})-t_i \tag{28}$$

该局部优化的算法步骤如下:

步骤1 令$i=1$,计算当前调度下目标函数值为A。

步骤2 计算作业i最早开始时间和最晚开始时间t_{es_i}、t_{ls_i}。

步骤3 从$t_{\mathrm{ST}_i}\in[t_{\mathrm{es}_i},t_{\mathrm{ls}_i}]$中选择使得目标函数值最优的$t_{\mathrm{ST}_i}$,更新$t_{\mathrm{ST}_i}$、$A$。若$i=n$,停止运算,否则令$i=i+1$转步骤2。

4 数据实验

为了验证本文设计的采用对作业延迟时间编码的遗传算法(DTGA)的有效性,选取10 jobs、14

jobs、18 jobs、30 jobs、60 jobs、90 jobs，各10个算例进行数值实验，与文献中对采用作业浮动时间进行编码的遗传算法(FTGA)[24]进行比较。数值实验在C#(Visual Studio 2017)语言环境下编程实现，测试平台为Intel Core i5 4th处理器，2.40 GHz主频，4G内存，结果如表4至表7所示。其中，A_D、A_F、A_C分别为该组算例在DTGA、FTGA和CPLEX下所求得平均目标函数值，T_D、T_F、T_C分别为平均运算时间，G_1、G_2分别为DTGA、FTGA所求目标函数值与CPLEX最优解的差距，G为DTGA和FTGA之间的差距。遗传算法的种群数为50，交叉概率为0.8，变异概率为0.3。

表4　10 jobs 实验结果

Tab.4　Scheduling result of 10 jobs

案例数	A_C	T_C/s	A_D	T_D/s	G_1/%	A_F	T_F/s	G_2/%
1	122	0.256	123	0.552	0.82	125.0	0.186	2.46
2	104	0.595	104	0.779	0.00	105.2	0.284	1.15
3	143	0.681	143	0.492	0.00	143.0	0.181	0.00
4	118	0.312	118	0.283	0.00	118.8	0.104	0.68
5	166	0.601	166	0.806	0.00	173.1	0.284	4.28
6	131	0.729	131	0.799	0.00	131.0	0.257	0.00
7	109	0.424	109	0.501	0.00	111.0	0.120	1.83
8	158	0.614	158	0.658	0.00	158.0	0.210	0.00
9	154	0.984	154	0.884	0.00	164.2	0.320	6.62
10	106	0.492	106	0.469	0.06	111.3	0.125	5.00
平均值					0.09			2.20

表4至表6显示了小规模的算例结果，算例规模分别为10 jobs、14 jobs、18 jobs，每组包含10个案例。从表中可得到：当作业数量为10 jobs和14 jobs时，本文所设计的算法求得的最优解基本等于CPLEX得到的最优解，而对比算法的结果与最优解有明显的偏差，在求解时间上3个算法都在同一个数量级范围。当作业数量为18 jobs时，本文所设计的算法与CPLEX得到的最优解仅为1.6%，对比算法的偏差却达到了10%，而求解时间上本文算法与对比算法在同一个数量级内，远远小于CPLEX的求解时间。因此，得出以下结论：在小规模的算例中，本文所提出的遗传算法在一定误差范围内均能求解出较好的结果。

表5　14jobs 实验结果

Tab.5　Scheduling result of 14 jobs

案例数	A_C	T_C/s	A_D	T_D/s	G_1/%	A_F	T_F/s	G_2/%
1	160	0.682	160	0.555	0.00	160.5	0.141	0.31
2	149	0.288	149	0.713	0.00	154.4	0.191	3.62
3	151	3.825	152.1	1.519	0.72	154.6	0.617	2.38
4	215	1.286	215.2	0.459	0.09	222.0	0.096	3.25
5	118	2.342	118.8	2.000	0.68	124.6	0.710	0.60
6	165	0.856	165.0	0.510	0.00	173.7	0.115	4.66
7	123	0.487	123.0	0.525	0.00	127.4	0.142	3.57
8	177	2.598	177.5	0.734	0.28	179.6	0.176	1.46
9	175	1.123	175.1	0.596	0.06	176.8	0.159	0.10
10	182	0.836	182.0	0.524	0.00	183.2	0.109	0.06
平均值					0.18			2.00

表6　18 jobs 实验结果

Tab.6　Scheduling result of 18 jobs

案例数	A_C	T_C/s	A_D	T_D/s	G_1/%	A_F	T_F/s	G_2/%
1	192	27.766	192.9	1.512	0.47	216	0.380	12.50
2	149	91.628	151.4	2.041	1.61	171.7	0.637	15.23
3	178	5.022	182.0	3.749	2.24	192.0	1.351	7.86
4	162	4.412	171.2	1.313	5.67	185.5	0.384	14.50
5	255	4.338	255.0	0.882	0.00	278.6	0.149	9.25
6	127	4.564	130.1	2.587	2.44	136.9	1.215	7.79
7	186	4.238	187.7	0.694	0.91	204.1	0.138	9.73
8	183	46.302	187.1	1.390	2.24	202.3	0.462	10.54
9	126	41.528	136.3	2.512	0.81	142.2	0.948	12.85
10	141	57.170	143.4	2.328	0.17	145.3	0.773	3.05
平均值					1.66			10.33

表7　中规模、大规模案例实验结果

Tab.7　Scheduling result of middle and large scale

算例数	30 jobs				60 jobs				90 jobs			
	A_D	A_F	A_C	G/%	A_D	A_F	A_C	G/%	A_D	A_F	A_C	G/%
1	240.3	252.6	230	5.12	278.4	293.5	UB=311	5.42	335.5	354.4	UB=395	5.63
2	249.7	265.7	238	6.41	341.5	360.1	UB=374	5.45	258.1	279.7	UB=331	8.37
3	221.4	241.0	215	8.85	320.0	347.8	UB=369	8.69	357.1	382.1	UB=423	7.00
4	265.8	269.6	UB=276	1.43	258.9	284.2	UB=307	9.77	315.8	338.7	UB=377	7.25
5	233.1	251.1	230	7.72	273.3	292.4	UB=315	6.99	265.2	287.5	UB=323	8.41
6	224.4	244.5	211	8.96	345.5	368.0	UB=392	6.51	358.5	386.2	UB=421	7.72
7	156.3	183.8	142	17.59	325.3	347.8	UB=376	6.92	307.9	329.5	UB=366	7.02
8	190.6	201.6	183	5.77	296.7	315.9	UB=334	6.47	331.5	355.1	UB=394	7.12
9	263.1	285.2	291	8.40	265.8	290.9	UB=323	9.44	283.2	315.8	UB=363	11.51
10	232.3	251.7	219	8.35	263.2	278.9	UB=301	5.96	290.4	311.8	UB=352	7.37
平均值				7.86				7.16				7.74

为了对比两种算法的优劣性,在其他设置参数相同的情况下,对比两种算法在不同的迭代次数下的求解结果。在3种不同规模下,各取10个案例,记录每个案例在不同迭代次数下的平均值,再对10个案例取平均值。图6至图8表示作业数目为30 jobs、60 jobs、90 jobs的情况下,两种不同算法的比较。横坐标为迭代次数,纵坐标为10个算例的平均值。通过图可以发现,随着迭代次数增加,两种算法的值逐渐降低,并且在不同的迭代次数下,本文设计的算法DTGA都明显优于对比算法FTGA。

表7显示了30 jobs、60 jobs、90 jobs 3种情况下本文设计算法与CPLEX及对比算法之间的对比。两种遗传算法的设置参数相同,迭代次数均为200。每组选择10个案例,为了实验结果更具代表性,每个案例求解10次并取平均值。针对大部分大规模算例,由于CPLEX无法求得精确解,因此与CPLEX在7 200 s内所求上界UB进行对比。从表7中的数据可以看出,在作业数目为30 jobs情况下,CPLEX能求得一部分算例的精确解,本文算法与精确解之间相差百分比平均约为4.75%,而对比算法与精确解之间相差百分比平均约为12.15%。与对比算法相比,本文算法在求解大规模算例问题上更加有效。

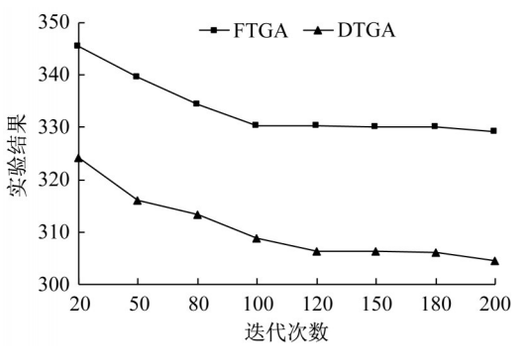

图6 30 jobs实验数据对比

Fig.6 Comparison of scheduling result of 30 jobs

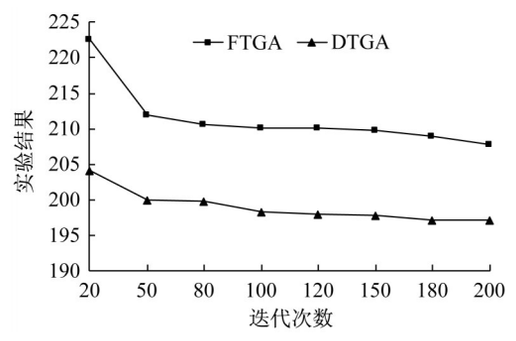

图7 60 jobs实验数据对比

Fig.7 Comparison of scheduling result of 60 jobs

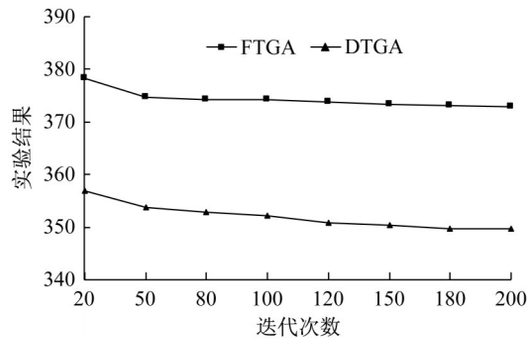

图8 90 jobs实验数据对比

Fig.8 Comparison of scheduling result of 90 jobs

5 总结与展望

本文在考虑实际生产过程中存在的换班情况下,提出了以最小化人力成本投入为目标的考虑排班的人力资源投入问题,建立了人力成本投入最小化为目标的数学模型。此外,本文将所提出的数学模型进行拆分,使得小规模问题可以直接用CPLEX进行求解。为了解决大规模问题,本文又提出了对作业延迟开始时间进行解码的遗传算法,并且对作业开始时间和延迟时间进行局部优化,从而得到最佳目标值。算例实验结果表明,无论是小规模的模型拆分还是大规模算法的构建都取得不错的效果。

在实际中,同一种人力资源之间,由于个体的不同也可能存在差异,然而本文中并没有考虑这种差异性,这也是本文未来的研究方向之一。

参考文献:

[1] MÖHRING R H. Minimizing costs of resource requirements in project networks subject to a fixed completion time [J]. Operations Research, 1984, 32(1): 89.

[2] DEMEULEMEESTER E. Minimizing resource availability costs in time-limited project networks [J]. Management Science, 1995, 41(10): 1590.

[3] RANGASWAMY B. Multiple resource planning and allocation in resource-constrained project networks [D]. Boulder: Graduate School of Business, University of Colorado, 1998.

[4] YAMASHITA D, ARMENTANO V, LAGUNA M. Scatter search for project scheduling with resource availability cost [J]. European Journal of Operational Research, 2006, 169(2): 623.

[5] SHADROKH S, KIANFAR F. A genetic algorithm for resource investment project scheduling problem, tardiness permitted with penalty [J]. European Journal of Operational Research, 2007,181(1): 86.

[6] RANJBAR M, KIANFAR F, SHADROKH S. Solving the

resource availability cost problem in project scheduling by path relinking and genetic algorithm[J]. Applied Mathematics and Computation, 2008,196(2): 879.

[7] ZHU X, RUI Z R, LI S, *et al*. An effective heuristic for project scheduling with resource availability cost[J]. European Journal of Operational Research, 2017,257(3): 746.

[8] 陆志强,周皓雪.带资源空窗期的资源投入型问题的建模与优化[J].同济大学学报(自然科学版),2019,47(10):1520.
LU Zhiqiang, ZHOU Haoxue. Modeling and optimization of resource investment problem with resource window[J]. Journal of Tongji University(Natural Science), 2019, 47(10):1520.

[9] 任逸飞,陆志强.多技能资源投入项目调度问题的建模与优化[J].同济大学学报(自然科学版),2017,45(11):1713.
REN Yifei, LU Zhiqiang. Model and optimization of resource investment project scheduling problem with multi-skill [J]. Journal of Tongji University(Natural Science), 2017, 45(11): 1713.

[10] SU C T, SANTORO M C, MENDES A B. Constructive heuristics for project scheduling resource availability cost problem with tardiness[J]. Journal of Construction Engineering and Management, 2018, 144(8): 04018074.

[11] BAKER K R. Workforce allocation in cyclical scheduling problems: a survey[J]. Operational Research Quarterly, 1976, 27(1): 155.

[12] KLETZANDER L, MUSLIU N. Solving the general employee scheduling problem [J]. Computers & Operations Research, 2020, 113: 104794.

[13] BERGH J V D, BELIËN J, BRUECKER P D, *et al*. Personnel scheduling: a literature review[J]. European Journal of Operational Research, 2013, 226(3): 367.

[14] DANIELS R L, MAZZOLLA J B, SHI D. Flow shop scheduling with partial resource flexibility [J]. Management Science, 2004,50(5): 658.

[15] DREZET L E, BILLAUT J C. Predictive and proactive approaches for RCPSP with labour constraints [J]. IFAC Proceedings Volumes,2006, 39(3): 125.

[16] ARTIGUES C, GENDREAU M, ROUSSEAU L M, *et al*. Solving an integrated employee timetabling and job-shop scheduling problem via hybrid branch-and-bound [J]. Computers & Operations Research, 2009, 36(8): 2330.

[17] GUYON O, LEMAIRE P, PINSON É, *et al*. Cut generation for an integrated employee timetabling and production scheduling problem [J]. European Journal of Operational Research, 2010, 201(2): 557.

[18] SMET P, ERNST A T, BERGHE G V. Heuristic decomposition approaches for an integrated task scheduling and personnel rostering problem [J]. Computers & Operations Research, 2016, 76: 60.

[19] VAN DEN EECKHOUT M , MAENHOUT B, VANHOUCKE M. A heuristic procedure to solve the project staffing problem with discrete time/resource trade-offs and personnel scheduling constraints [J]. Computers & Operations Research, 2019, 101: 144.

[20] LI H, XIONG L, LIU Y, *et al*. An effective genetic algorithm for the resource levelling problem with generalized precedence relations [J]. International Journal of Production Research, 2018, 56(5): 2054.

[21] SELVAM G, TADEPALLI T C M. Genetic algorithm based optimization for resource leveling problem with precedence constrained scheduling [J/OL]. International Journal of Construction Management, 2019. https://doi. org/10.1080/15623599.2019.1641891.

[22] REN Y, LU Z. A flexible resource investment problem based on project splitting for aircraft moving assembly line [J]. Assembly Automation, 2019, 39(4): 532.

[23] LU Z, REN Y, Wang L, *et al*. A Resource investment problem based on project splitting with time windows for aircraft moving assembly line [J]. Computers & Industrial Engineering, 2019, 135: 568.

[24] NAJAFI A A, NIAKI S T A. A genetic algorithm for resource investment problem with discounted cash flows [J]. Applied Mathematics and Computation, 2006,183(2): 1057.