文章编号：0253-374X(2017)11-1713-09 DOI:10.11908/j.issn.0253-374x.2017.11.020

多技能资源投入项目调度问题的建模与优化

任逸飞，陆志强
(同济大学 机械与能源工程学院，上海 201804)

摘要：以大型工业品移动装配线为实际背景，针对多技能人力资源投入成本问题，建立了以最小化资源投入成本为目标的数学模型. 针对该模型，提出了基于全局作业影响的改进调度机制的遗传算法，设计最小费用最大流多技能资源分配算法解决资源分配问题，通过基于全局资源水平影响的作业调度评估策略优化非关键作业的调度位置. 最后利用改造的标准算例库 PSPLIB 进行数值试验，通过与 CPLEX 和文献算法对比，验证了本文算法的有效性.

关键词：多技能；资源投入；项目调度；遗传算法
中图分类号：F273 **文献标志码**：A

Modeling and Optimization of Resource Investment Project Scheduling Problem with Multi-skill

REN Yifei, *LU Zhiqiang*
(School of Mechanical Engineering, Tongji University, Shanghai 201804, China)

Abstract: The resource investment problem with multi-skill was proposed based on the large industrial products moving assembly line. A mathematical model with the objective function of minimizing the resource usage cost was proposed. A genetic algorithm with an improved schedule generation scheme based on the global operations impact was developed to solve this model. The scheduling location for non-critical jobs was optimized by the strategies based on the global operations impact and subsequently the resource allocation problem was effectively solved by the minimum cost-maximum flow multi-skill resource allocation algorithm. Numerical experiment was carried out by using the modified standard example library PSPLIB, and the validity of the algorithm is verified by comparison with CPLEX and other literature algorithm.

Key words: multi-skill; resource investment; project scheduling; genetic algorithm

在实际生产中，人力资源是主导整个生产流程的关键性资源之一，而人力资源可以通过学习锻炼掌握不同的能力，从而适应不同工种，将此类人力资源称为多技能工或多技能资源. 大型工业品如飞机的装配过程中，装配工序多，装配过程复杂，科学安排生产计划和合理使用多技能工，对于提高生产效率、降低成本具有重要的现实意义. 基于上述实际生产背景，本文以资源投入成本为优化目标，引入多技能资源因素，研究多技能资源投入项目调度问题(MSRIPSP)的建模与优化方法.

资源投入项目调度问题(RIP)是通过具有正则目标(makespan)的经典项目调度问题(RCPSP)变形拓展而来，目标是在给定工期内求得可更新资源成本最小值. 由于其任意资源组合下仍是求解一个 RCPSP 问题，因此相较 RCPSP 具有更高的复杂度，而 RCPSP 作为 RIP 的基础，其建模与优化方法对于 RIP 问题的研究具有借鉴意义. Möhring[1] 首先提出了 RIP 问题并证明该问题是 NP-hard (non-deterministic polynomial, NP)问题；Drexl 等[2] 详细介绍了用拉格朗日松弛和列生成方法求解 RIP 下界的算法；Rodrigues 等[3] 设计了一种改进的分支定界算法，计算新的下界以减少解空间，提高了分支方案的有效性. 由于精确算法无法求解大规模问题，因此很多研究针对这类问题设计了不同的元启发式算法进行求解. Yamashita 等[4] 将 RIP 转化为 RCPSP 并通过一种基于 scatter search 的元启发式算法求解该问题；Shadrokh 等[5] 采用遗传算法求解具有延迟惩罚的资源投入问题，其中染色体编码分别对活动列表和资源容量进行编码；Afshar[6] 通过以作业浮动时间为编码的模拟退火方法求解需要招募和释放资源的多模式 RIP 问题. 以上算法主要思想都是将 RIP 转化为 RCPSP，通过对资源投入量进行搜索设

收稿日期：2017-05-03
基金项目：国家自然科学基金(61473211, 71171130)
第一作者：任逸飞(1994—)，男，博士生，主要研究方向为多技能资源投入项目调度.
E-mail: renyifei1201@tongji.edu.cn
通讯作者：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

扫码查看作者独家介绍

定,但是由于所设资源投入量上下界差距较大,使得算法搜索效率低下. 而 Ranjbar 等[7]首次在没有将 RIP 转化为 RCPSP 的基础上对作业序列编码通过路径重链和遗传算法来求解问题,但是其算法在搜索期间需要对优先级不可行的活动列表进行调整,导致算法效率降低,而且在对任务进行调度时没有考虑当前选择位置对全局资源投入的影响.

现有 RIP 问题在考虑资源因素时通常设定资源只具备一种能力,很少考虑多技能资源的情况,但是在 RCPSP 中已有文献考虑资源的多技能性,扩展成多技能资源项目调度问题(MSPSP). Bellenguez 等[8]提出了 MSPSP 问题及其扩展并建立数学模型,并通过分支定界法求解该问题;Li 等[9]针对 MSPSP 只考虑每个活动的每个技能只需要一个资源来执行提出了一种基于混合整数线性规划和约束规划的 Benders 分解法;Montoya 等[10]设计了一个包括列生成和分支定界的混合算法求解该问题. 针对大规模问题,主要设计相应的启发式算法和搜索算法来解决. Firat 等[11]通过一种启发式方法解决技术人员对具有多层次技能需求的活动的分配问题;Almeida 等[12]引入资源权重和活动分组的概念,并使用基于并行调度的启发式框架来解决 MSPSP. 在元启发式方面, Myszkowski 等[13]提出了一种嵌入启发式优先级规则的混合蚁群优化算法,通过启发式规则对资源进行分配;张猛等[14]通过基于不同情形下局部两作业资源需求的处理策略的调度机制和资源置换算法解决多技能资源分配问题. 现有研究对 MSPSP 中资源选择分配到技能时很少同时兼顾作业时间调度和资源分配问题,而基于网络最大流的资源分配方法中对应同一最大流的多种分配方案在选择时具有一定随机性. 而现有 MSPSP 中资源分配方法对于 RIP 中多技能资源的分配具有参考价值.

综上所述,结合大型工业品装配过程的实际问题,在现有对 RIP 和 MSPSP 的问题研究基础上提出了多技能资源投入项目调度问题(MSRIPSP). 针对现有文献的不足,在利用关键路径法(CPM)获得非关键任务集合基础上,通过基于全局资源水平影响的改进调度算法对非关键任务的调度位置进行优化决策,统筹考虑其可排区间对整体资源投入的影响,选择最佳时间为其开始时间. 同时结合最小费用最大流算法分配资源给作业技能,以减少资源选择的随机性,最后在调度完成后通过资源置换算法对目标进行进一步优化.

1 问题描述及数学模型

记一个项目 P 由 n 个作业构成,作业集合为 $J=\{1,2,\cdots,j,\cdots,n\}$,$j$ 为作业编号,$j\in J$,作业的时序约束通过具有 $n+2$ 个节点且没有环路的节点网络图 AON 表示,其中节点 0 和 $n+1$ 表示项目 P 的唯一开始和结束作业,2 个作业均为虚拟作业且作业时间为零. 项目中每一个作业的标准作业时间为 t_j,给定项目工期上限为 $\overline{T}$,设所有作业均不可中断. 项目中所有作业共需要 S 种技能,且所有技能均由 K 种可更新资源提供,资源集合和技能集合分别记为 $K_C=\{1,2,\cdots,k,\cdots,K\}$,$S_C=\{1,2,\cdots,s,\cdots,S\}$,$k$ 为资源编号,$k\in K_C$,s 为技能编号,$s\in S_C$,k 种可更新资源单位时间内的费用分别为 c_k,W_{weight} 为权重系数. 设作业对技能的需求由作业技能矩阵(JSM)表示,定义 $\boldsymbol{G}_{\text{JSM}}=[r_{js}]_{|J|\times|S|}$,$r_{js}$ 表示第 j 项作业对第 s 种技能的需求量,$r_{js}\in\mathbf{N}^*$. 资源技能矩阵(RSM)表示资源和技能之间的关系,定义 $\boldsymbol{G}_{\text{RSM}}=[\delta_{ks}]_{|K|\times|S|}$,其中 δ_{ks} 表示资源 k 是否具备技能 s,$\delta_{ks}\in\{0,1\}$,$\delta_{ks}=1$,表示资源 k 具备技能 s,否则为零. S_j 表示作业 j 需求的技能集合,S_k 表示资源 k 拥有的技能集合,设资源 k 在某时刻 t 只能使用一种技能 s 执行一项作业 j.

P_j 为第 j 项作业的紧前作业集合,i 为 j 的紧前作业,$i\in P_j$,$P_{\text{pred}(j)}$ 表示第 j 项作业的所有前相关作业集合,$S_{\text{succ}(j)}$ 表示第 j 项作业的所有后相关作业集合. t_{EST_j}、t_{EFT_j}、t_{LST_j}、t_{LFT_j} 分别表示通过 CPM 法得到的作业 j 的最早开始时间、最早结束时间、最晚开始时间和最晚结束时间;t_{AST_j} 和 t_{AFT_j} 分别表示作业 j 在调度时的开始时间和结束时间;t_{ST_j} 和 t_{FT_j} 分别为作业 j 的最终实际开始时间和实际结束时间,d 为离散时间点,$d\in D$,$D=\{1,2,\cdots,d,\cdots,T\}$,$T$ 为项目实际完工总时间.

求解目标函数为满足各种约束下的最小化项目总费用 A. 决策变量如下:① x_{jd}. 为 0、1 变量,作业 j 在 d 时刻处于执行状态则为 1,否则为 0. ② φ_{ij}. 为 0、1 变量,作业 j 在作业 i 结束后开始则为 1,否则为 0. ③ y_{jkd}. 为整数变量,d 时刻执行作业 j 的资源 k 的数量. ④ h_{jkds}. 为 0、1 变量,资源 k 在 d 时刻使用技能 s 执行作业 j 则为 1,否则为 0.

目标函数为

$$\min A=\sum_{k\in K}c_k\max_{d\in[1,T]}\sum_{j\in J}y_{jkd} \tag{1}$$

约束为

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

$$d \times x_{jd} \leqslant T, \forall d \in D, \forall j \in J \tag{3}$$

$$t_p x_{jd} \leqslant \sum_{i=1}^{d-1} x_{pi}, \forall p \in P_j, \forall j \in J, \forall d \in D \tag{4}$$

$$\varphi_{ij} + \varphi_{ji} \leqslant 1, \forall i,j \in J, i > j \tag{5}$$

$$0 \leqslant T \leqslant \overline{T} \tag{6}$$

$$t_j x_{jd} - t_j x_{j(d+1)} + \sum_{i=d+2}^{T} x_{ji} \leqslant t_j, \quad \forall j \in J, \forall d \in D \tag{7}$$

$$t_j \times h_{jkds} = \sum_{d=1}^{T} h_{jkds}, \quad \forall j \in J, \forall d \in D, \forall s \in S, \forall k \in K \tag{8}$$

$$\sum_{k\in K} y_{jkd} = r_{js} \times x_{jd}, \forall j \in J, \forall d \in D, \forall s \in S \tag{9}$$

$$x_{jd}, h_{jkds} = \{0,1\}, \forall j \in J, \forall d \in D, \quad \forall k \in K, \forall s \in S \tag{10}$$

$$\varphi_{ij} = \{0,1\}, \forall i,j \in J, i \neq j \tag{11}$$

$$c_k = W_{\text{weight}} \times \sum_{s=1}^{S} \delta_{ks}, \forall k \in K \tag{12}$$

模型中各式含义如下：式(1)表示目标函数，最小化资源投入成本；式(2)表示确保每一个作业在标准作业工期内都被执行完成；式(3)表示项目中所有作业的结束时间都小于项目的结束时间；式(4)表示每一项作业在其所有紧前作业完成后才能开始，即时序约束；式(5)表示两作业不能同时在对方开始之前结束，辅助限定两作业间优先关系；式(6)表示项目实际完成工期不超过给定工期上限；式(7)表示作业一旦开始就不可中断；式(8)表示资源是非抢占式的；式(9)表示满足各个作业对技能的需求；式(10)和式(11)表示决策变量的可行域；式(12)表示各个资源成本与其所掌握技能数量之间的权重关系.

2　算法设计原理

多技能资源是一种特殊资源，在本文模型中设定：每个作业需要多个不同技能，每种资源可以掌握多种不同技能，同时每个资源在同一时间点只能使用一种技能执行一项作业. 结合问题描述可见，在作业技能矩阵 $\boldsymbol{G}_{\text{ISM}}$ 中，对于任意作业 $j\in J$，所需技能数量为 $M_j = \sum_{s=1}^{S} r_{js}$. 定义技能因素($E_{\text{SF}}$)表示作业所需技能的平均水平，获得 $E_{\text{SF}} = \frac{1}{s}\sum_{j=1}^{n}\sum_{s=1}^{s} r_{js} = \sum_{j=1}^{n}\frac{M_j}{s}$. 例如：设每个活动需求的技能数量从集合{1,2,3}中选择，$E_{\text{SF}}=0.3$ 表示每个活动需要 1 种技能；$E_{\text{SF}}=1$ 表示每个活动需要 3 种技能；E_{SF} 为随机表示每个活动需要的技能数量从集合中随机选择. 当作业所需技能数量越多时，所需资源也就越多，因此 E_{SF} 是影响该作业调度复杂度的原因之一. 资源柔性度 F 表示所有资源掌握的技能数量总和与技能数和资源种类乘积的比值，即：$F = \sum_{S\in S_C}\sum_{k\in K_C}\delta_{ks}/(|S|\times|K|)$. 当每种资源掌握的技能数量越多时，$F$ 越大，则资源分配技能的组合越多，灵活度高，因此对于 MSRIPSP 问题很难采用传统多模式项目调度问题的搜索算法[6]. 同时如果将其转化为多技能资源约束项目调度问题[4-5]求解，则由于资源投入量上下界间距较大，使得搜索空间变大，导致算法效率降低. 为解决本文问题的难题，算法设计核心思想如下：以遗传算法为框架，在只考虑作业时序约束下，通过 CPM 获得关键作业集合 C_r、非关键作业集合 F_r 以及项目完成工期 T_{cpm}($T_{\text{cpm}}<\overline{T}$)，对非关键作业优先级列表和 $\Delta T=\overline{T}-T_{\text{cpm}}$ 进行编码，通过改进的调度算法进行解码：首先调度关键作业，确定各个关键作业的开始时间，然后在确定非关键作业可排区间基础上，利用任意作业的最优位置判断基准对非关键作业在可排区间内的位置进行评测，优化确定非关键任务的最终开始时间，确定满足各种约束条件下使得目标函数值最小的调度方案，最后通过局部资源调整策略对调度结果进行进一步优化，得到最终目标函数值.

2.1　遗传算法

2.1.1　遗传算法框架

采用双链表编码的遗传算法框架，对各项非关键作业和 CPM 得到的工期与工期上限 $\overline{T}$ 的差值 ΔT 进行编码. 设种群规模 N，最大迭代次数为 $\lambda_{\max}$，第 λ 代群体中第 ν 个染色体用 $x_{\nu,\lambda}$ 表示，$A(x_{\nu,\lambda})$ 表示对应的调度方案的目标函数值，$x_{\text{best},\lambda}$ 表示第 λ 代群体中的最优解，全局最优解为 $x_{\text{best},A}$，算法步骤如下：

(1)初始化染色体种群，$\lambda=0$.

(2)判断若 $\lambda<\lambda_{\max}$，则转步骤(3)；否则跳转步骤(6).

(3)将每条染色体 $x_{\nu,\lambda}$ 中的优先级列表转化为非关键作业执行列表集合 L，调用基于全局作业影响的改进调度算法进行解码，得到各染色体对应调度方案的目标函数值 $A(x_{\nu,\lambda})$，更新 $x_{\text{best},\lambda}$；如果 $A(x_{\text{best},\lambda})<A(x_{\text{best},A})$，则 $A(x_{\text{best},A})=A(x_{\text{best},\lambda})$，$x_{\text{best},A}=x_{\text{best},\lambda}$.

(4)计算群体中所有个体的适应值，根据轮盘赌

的方式选择新的个体.

(5)对选择出来的个体进行交叉变异操作,得到新的种群,$\lambda=\lambda+1$,跳转步骤(2).

(6)满足终止条件,输出当前最优个体对应的目标函数值 $A(x_{\nu,\lambda})$,即为整个项目资源投入的最小值.

2.1.2 编码

遗传算法采用双链表编码,对象为非关键作业和 ΔT,每组编码的第1层对应各非关键作业优先级列表,采用实数编码,即如果有 a 项非关键作业,则编码长度为 a,若编码第 m 位为 j,则代表非关键作业 j 的作业调度优先级为 m. 第2层对应 ΔT 的拆分数量和大小同样采用实数编码,即根据关键作业集合 C_r 内元素个数 n 将 ΔT 随机拆分成 $n+1$ 份,每一份大小为 ΔT_{n+1},$\Delta T \geqslant \Delta T_{n+1} \geqslant 0$ 且 $\Delta T_1+\Delta T_2+\cdots+\Delta T_{n+1}=\Delta T$. 如图1所示:编码第1层为6项非关键作业优先级列表,编码第2层为将 $\Delta T=4$ 随机拆分成5份,每份大小依次为0、2、0、1、1,且 0+2+0+1+1=4.

非关键作业	8	4	5	7	3	10
ΔT=4	0	2	0	1	1	

图1 编码

Fig. 1 Coding

2.2 基于全局资源水平影响的改进调度算法

2.2.1 调度关键作业

通过CPM获得的项目完成工期为理论上的最短完成工期 T_{cpm},由RCPSP问题可得当 $T=T_{cpm}$ 时项目对各资源的需求量为最大值. 随着延迟各作业的开始时间,原本并行执行的作业 i,j 可通过延迟其中一个作业的开始时间,使得两作业可串行执行,则可以达到降低各个资源需求量峰值的目的.

性质1 在资源分配到作业确定的情况下,对于RIP问题,当 $T=\overline{T}$ 时,各个资源需求量的峰值最低,资源投入成本最小.

证明 对于RIP问题,设当完成工期为 T 时的最优调度计划为 S_{sche},各资源使用峰值为 R_k. 由文献[12-13]可得:在资源 R_k 约束下的RCPSP的最优调度计划 S'_{sche} 也是RIP的最优调度计划,可得相应项目完成工期 $T_{RCPSP} \leqslant T \leqslant \overline{T}$. 对于RCPSP而言,可知项目完成工期为 T_{RCPSP} 时各资源使用的峰值不小于 $\overline{T}$ 时的峰值,因此在目标完成工期为 $\overline{T}$ 时RCPSP对应的最优调度计划为 S''_{sche},资源约束为 $\overline{R}_k$,对应的RIP最优调度也为 S''_{sche},此时当 $T=\overline{T}$ 时,各资源需求量的峰值为 $\overline{R}_k$ 最低,资源投入成本最小.

因此,按照CPM得到的关键作业的 t_{EST_j} ($j \in C_r$)调度关键作业,即 $t_{AST_j}=t_{EST_j}$,并对 ΔT_{n+1} 编码.

定义1 ΔT_{n+1} 为 $n+1$ 个虚拟时间作业,即该作业只占用时间,不消耗任何资源和空间.

将 ΔT_{n+1} 解码后插入到已调度关键任务的前后,更新关键作业 $t_{AST_j}=t_{AST_j}+\Delta T_n$,$T=\overline{T}$,如图2所示. 在保持关键作业调度不变的情况下,再对非关键作业进行调度.

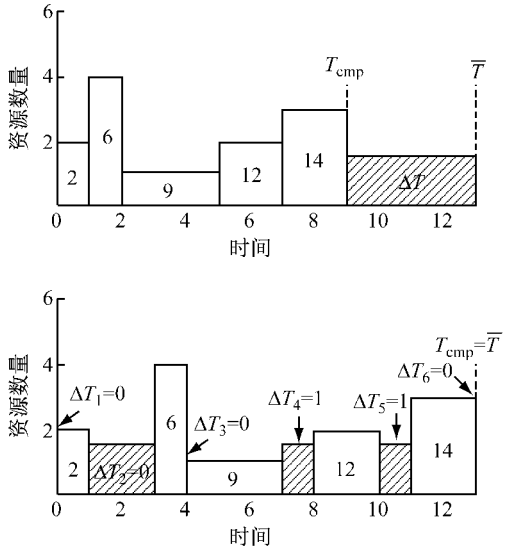

图2 关键作业调度方案

Fig. 2 Scheduling of critical jobs

2.2.2 确定非关键作业的决策区间

在非关键作业集合 F_r 中,选择优先级最高的非关键作业 i 进行调度,因为关键作业的 t_{AST_j} 更新改变,所以当前调度非关键作业的可排区间受到已调度作业的决策位置影响,因此后期调度作业 j 的开始时间 t_{AST_j} 的决策区间是动态变化的. 设集合 E 为已调度作业集合,U 为未调度作业集合,D 为将要调度作业集合.

定义2 作业 i 与作业 j 具有时序关系且能够影响作业 j 的开始时间,则称作业 i 为作业 j 的前相关作业,记 $i<j$,集合 $X=\{i \in E \mid i<j\}$ 为作业 j 的前相关作业集合.

定义3 作业 i 为作业 j 的前相关作业,$i<j$,两作业的相关距离为任务 i 到任务 j 的最长路径的长度,记为 d_{ij},$d_{ij}=-d_{ji}$.

因此,可以得到作业 j 的 t_{AST_j} 的决策区间$[t_{ES_j}, t_{LS_j}]$的更新公式如下:

$$t_{ES_j}=\max\{d_{0j},\max_{i\in X}(t_{ST_i}+d_{ij})\} \quad (13)$$

$$t_{LS_j}=\min\{\overline{T}-d_{j,n+1},\min_{i\in X}(t_{ST_i}-d_{ji})\} \quad (14)$$

确定好作业 j 的决策区间$[t_{ES_j}, t_{LS_j}]$后，需要通过基于全局资源水平影响的作业调度评估策略得到作业 j 的最优 t_{AST_j}.

2.2.3　任意作业 j 的最优位置判断基准

由作业时序约束关系得到将要被调度作业 j 的前相关作业集合 $P_{pred(j)}$ 和后相关作业集合 $S_{succ(j)}$，则 $t_{AST_x} < t_{AST_j}, x \in P_{pred(j)}$；$t_{AST_y} > t_{AST_j}, y \in S_{succ(j)}$. 依据作业 j 在调度时其 t_{AST_j} 和 t_{AFT_j} 会与已调度作业 $i \in E$、$i' \in E$ 的开始时间和结束时间有不同的重叠关系，归纳以下 4 种不同情况以及相应的评估策略.

2.2.3.1　情况 1

情况为：$t_{AST_j} = t_{FT_i}$ 或 $t_{AST_j} = t_{ST_i}$，$t_{AFT_j} = t_{FT_{i'}}$ 或 $t_{AFT_j} = t_{ST_{i'}}$.

此情况下包括 4 种小情况：$t_{AST_j} = t_{FT_i}$，$t_{AFT_j} = t_{FT_{i'}}$；$t_{AST_j} = t_{FT_i}$，$t_{AFT_j} = t_{ST_{i'}}$；$t_{AST_j} = t_{ST_i}$，$t_{AFT_j} = t_{FT_{i'}}$；$t_{AST_j} = t_{ST_i}$，$t_{AFT_j} = t_{ST_{i'}}$，但是对应的全局作业影响策略一样，因此划分到同一种情况，此时已调度作业 i：$t_{FT_i} = t_{AST_j} \parallel t_{ST_i} \geqslant t_{AFT_j} \parallel t_{ST_i} \geqslant t_{AST_j} \& t_{FT_i} \leqslant t_{AFT_j}$. 此时将工期 $\overline{T}$ 划分成 3 部分 $\mathbb{A}, \mathbb{B}, \mathbb{C}$，其中 $\mathbb{A}_t = [0, t_{AST_j})$，$\mathbb{B}_t = [t_{AST_j}, t_{AFT_j}]$，$\mathbb{C}_t = (t_{AFT_j}, \overline{T})$，作业$\{x \mid x \in P_{pred(j)} \& x \neq i\}$和$\{i \mid i \in E \& t_{FT_i} \leqslant t_{AST_j}\}$在 $\mathbb{A}$ 部分，作业 j 和作业$\{i \mid i \in E \& t_{ST_i} \geqslant t_{AST_j} \& t_{FT_i} \leqslant t_{AFT_j}\}$在 $\mathbb{B}$ 部分，作业$\{y \mid y \in S_{succ(j)} \& y \neq i\}$和$\{i \mid i \in E \& t_{ST_i} \geqslant t_{AFT_j}\}$在 $\mathbb{C}$ 部分. 由于所有作业未完全调度，因此无法求出各部分的各资源使用峰值，设在此处定义各资源的预估峰值 $H_{\mathbb{I}k}, k \in K_C, \mathbb{I} = \mathbb{A}, \mathbb{B}, \mathbb{C}$. 随着作业不断调度，$x \in P_{pred(j)}$ 和 $y \in S_{succ(j)}$ 中会有作业成为已调度作业，因此要注意避免重复计算的作业.

定义 4　各部分各资源的预估峰值 $H_{\mathbb{I}k}$ 等于各部分中各作业对该资源需求量与该作业工期乘积的和与各部分时间的比值，即

$$H_{\mathbb{I}k} = \frac{\sum_{j \in \mathbb{I}_J} t_j r_{jk}}{|\mathbb{I}_t|}, \quad k \in K_C \tag{15}$$

式中：$|\mathbb{I}_t|$ 为各部分时间长度，即 $|\mathbb{A}_t| = |t_{AST_j}|$，$|\mathbb{B}_t| = |t_{AFT_j} - t_{AST_j}| = |t_j|$，$|\mathbb{C}_t| = |\overline{T} - t_{AFT_j}|$；$\mathbb{I}_J$ 为各部分内的所有作业.

选择各资源在 $\mathbb{A}, \mathbb{B}, \mathbb{C}$ 各部分的预估峰值的最大值为作业 j 开始时间在 t_{AST_j} 时的各个资源的峰值 $H_{max(k)}$，即 $H_{max(k)} = \max\{H_{\mathbb{I}k} \mid \mathbb{I} = \mathbb{A}, \mathbb{B}, \mathbb{C}\}$，进而求出此时的目标函数值：$A_{(t_{AST_j})} = \sum_{k \in K_C} H_{max(k)} c_k$. 因为由式(13)或式(14)得出作业 j 的 $t_{AST_j} \in [t_{ES_j}, t_{LS_j}]$，因此更新 t_{AST_j}：$t_{ES_j} \rightarrow t_{LS_j}$，计算每一个时间点的目标函数值 $A_{(t_{AST_j})}$，从中选择最小的一个值为最终的目标函数值 $A = \min\{A_{(t_{AST_j})}\}$，此时刻的 t_{AST_j} 为作业 j 最终的开始时间 t_{ST_j}，$t_{ST_j} = t_{ST_j} + t_j$.

2.2.3.2　情况 2

情况为：$t_{FT_i} > t_{AFT_j} > t_{ST_i}$，$t_{AFT_j} = t_{FT_{i'}}$ 或 $t_{AFT_j} = t_{ST_{i'}}$.

此时，存在一个已调度作业 $i \in E$，$t_{ST_i} < t_{AST_j} \& t_{FT_i} > t_{AST_j}$，当按照情况 1 的方式将工期 $\overline{T}$ 划分成 3 部分时，$\mathbb{A}, \mathbb{B}$ 两部分内都有作业 i，$\mathbb{C}$ 部分与情况 1 相同. 当计算 $\mathbb{A}, \mathbb{B}$ 两部分的各资源峰值时，针对特殊划分的作业 i 分别计算到 2 部分中，则可得

$$H_{\mathbb{A}k} = \frac{r_{ik} \mid t_{AST_j} - t_{ST_i} \mid + \sum_{j \in \mathbb{A}_J} t_j r_{jk}}{|\mathbb{A}_t|}, \quad k \in K_C \tag{16}$$

式中：$|\mathbb{A}_t|$ 为各部分时间长度，即 $|\mathbb{A}_t| = |t_{AST_j}|$；$\mathbb{A}_J$ 表示除作业 i 以外 $\mathbb{A}$ 部分内的所有作业.

$$H_{\mathbb{B}k} = \frac{r_{ik} \mid t_{FT_I} - t_{AST_j} \mid + \sum_{j \in \mathbb{B}_J} t_j r_{jk}}{|\mathbb{B}_t|}, \quad k \in K_C \tag{17}$$

式中：$|\mathbb{B}_t|$ 为各部分时间长度，即 $|\mathbb{B}_t| = |t_{AFT_j} - t_{AST_j}| = |t_j|$；$\mathbb{B}_J$ 表示除作业 i 以外 $\mathbb{B}$ 部分内的所有作业.

$H_{\mathbb{C}k}$ 的求解公式和式(15)一样，同情况 1 求出最终的目标函数值 $A = \min\{A_{(t_{AST_j})}\}$，此时刻的 t_{AST_j} 为作业 j 最终的开始时间 t_{ST_j}，$t_{FT_j} = t_{ST_j} + t_j$.

2.2.3.3　情况 3

情况为：$t_{AST_j} = t_{ST_i}$ 或 $t_{AST_j} = t_{FT_i}$，$t_{FT_{i'}} > t_{AFT_j} > t_{ST_{i'}}$.

此时，存在一个已调度作业 $i \in E$，$t_{ST_i} < t_{AST_j} \& t_{FT_i} > t_{AST_j}$，当按照情况 1 的方式将工期 $\overline{T}$ 划分成 3 部分时，会存在将作业 i 划分到 $\mathbb{B}, \mathbb{C}$ 两部分内，$\mathbb{A}$ 部分划分与情况 1 相同. 当计算 $\mathbb{B}, \mathbb{C}$ 两部分的各资源使用峰值时，针对特殊划分的作业 i 要分别计算到 2 部分中，则可得

$$H_{\mathbb{B}k} = \frac{r_{ik} \mid t_{AFT_j} - t_{ST_i} \mid + \sum_{j \in \mathbb{B}_J} t_j r_{jk}}{|\mathbb{B}_t|}, \quad k \in K_C \tag{18}$$

$$H_{\mathbb{C}k} = \frac{r_{ik} \mid t_{FT_j} - t_{AFT_j} \mid + \sum_{j \in \mathbb{C}_J} t_j r_{jk}}{|\mathbb{C}_t|}, \quad k \in K_C \tag{19}$$

$\mathbb{C}_j$ 表示除作业 i 以外 $\mathbb{C}$ 部分内的所有作业. 同理可以比较得出各资源的使用量峰值，并最终的目标函数值 $A = \min\{A_{(t_{AST_j})}\}$，此时刻的 t_{AST_j} 为作业 j

最终的开始时间 t_{ST_j}.

2.2.3.4 情况4

情况为:$t_{FT_i}>t_{AST_j}>t_{ST_i}$,$t_{FT_{i'}}>t_{AFT_j}>t_{ST_{i'}}$.

此时会出现将2个已调度作业 i,i' 都划分到2个部分中,其中作业划分到 A,B 两部分内,作业 i' 划分到 B,C 两部分内,则此时经分析可知:H_{Ak} 求解与情况2相同,即 H_{Ak} 求解公式为式(16);H_{Ck} 求解情况与情况3相同,即 H_{Ck} 的求解公式为式(19).

$$H_{Bk}=(r_{ik}\mid t_{FT_i}-t_{AST_j}\mid+r_{i'k}\mid t_{AFT_j}-t_{ST_{i'}}\mid+\sum_{j\in B_J}t_j r_{jk})/\mid B_t\mid \tag{20}$$

式中:B_J 表示除作业 i,i' 以外 B 部分内的所有作业.同理,可以得出作业 j 最终的开始时间 t_{ST_j}.

则基于全局作业影响的改进调度算法步骤如下:

(1) 初始化各数据,由CPM得出关键作业集合 C_r 和非关键作业集合 F_r 以及项目完成工期 T_{cpm},计算 $\Delta T=\overline{T}-T_{cpm}$.

(2) 依次对关键作业 i 按照 t_{EST_i} 进行调度并顺次建立流量网络图 $G(i)$,通过基于最小费用最大流的多技能资源分配模型为作业 i 分配资源.

(3) 将 ΔT 按照2.2.1的拆分方法拆分后,更新各关键作业的 t_{ST_i} 和 t_{FT_i}.

(4) 从非关键作业列表 L 中顺序调度非关键作业 j,由式(13)或(14)确定作业 j 的 t_{AST_j} 决策区间 $[t_{ES_j},t_{LS_j}]$.

(5) 通过基于全局资源水平影响的作业调度评估策略确定作业 j 的 t_{AST_j} 和预估最小目标函数值 $A=\min\{A_t\}$,则 $t_{AST_j}=t$,更新 $t_{ST_j}=t_{AST_j}$.

(6) 判断非关键作业列表 L 是否为 $\varnothing$,如果 $L\neq\varnothing$,则跳转步骤(4);否则转步骤(7).

(7) 输出此时调度计划 $S_{sche(best)}$ 和全局目标函数最优解 A_{best},调用局部资源置换算法对资源分配进行优化,输出最终目标函数值 A.

2.3 基于最小费用-最大流的多技能资源分配算法

在资源选择分配时,由于多技能资源组合复杂多样,当能够满足作业技能需求的资源分配方案不止一种时,网络最大流算法选择具有随意性,为了避免资源的随机选择和分配,通过定义权重 w_k 与资源相关联,并构造最小费用最大流的多技能资源分配算法,在考虑对目标函数影响的基础上,选择最优的资源分配方案.

定义5 各资源的权重等于其掌握技能的数量与该资源的单价及其所掌握技能的稀有程度的乘积.

$$w_k=c_k\sum_{s=1}^{S}\delta_{ks}\max_{s\in S_k}\left\{\frac{K}{\mid K_s\mid}\sum_{j\in J_s}r_{js}\right\}$$

式中:S_k 表示资源 k 拥有的技能集合;$\mid K_s\mid$ 表示掌握技能 s 的所有资源的数量和;J_s 表示需求技能 s 的所有作业集合.该资源权重规则是"静态的",即可以计算出来并在执行期间保持不变.

设有向网络图 $G=(V,E,C,\sigma)$,其中 V 表示节点集,$V=\{X,V_k,V_s,J\}$,E 表示有向弧集,$E=\{E_{xk},E_{ks},E_{sj}\}$,$C$ 表示容量集合,$C=\{C_{xk},C_{ks},C_{sj}\}$,$\sigma$ 为节点间有向弧的费用,如图3所示.在节点集中,X 为源点,是一虚拟点;J 为汇点,即作业节点(作业 j);V_k 表示资源节点集合,$V_k=K_C$;V_s 表示技能节点集合,$V_s=S_C$.在有向弧集中,E_{xk} 表示资源供应关系,即资源 k 在其调度时可用;E_{ks} 表示资源 k 可以执行技能 s;E_{sj} 表示作业 j 对技能 s 的需求情况;$C(e)$ 表示各个有向弧 e 的容量,在容量集 C 各子集中,C_{xk} 表示对应的资源在某一时刻只能执行一种技能,其各元素值为1;C_{ks} 中元素值为1表示该资源 k 可以使用对应的技能 s,C_{sj} 表示作业 j 对技能 s 的需求量,其值为 r_{js}.设在网络图中资源节点 V_k 和技能节点 V_s 间的有向弧在 d 时刻的费用为 $\sigma_{ks}=w_k$,其余各处有向弧处费用为零.

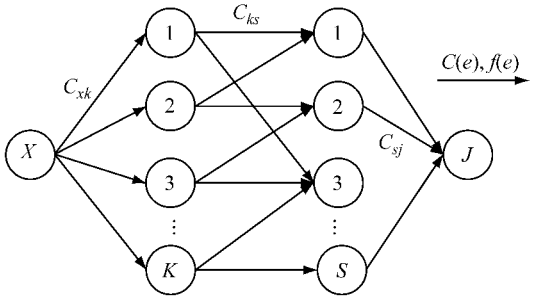

图3 最小费用-最大流多技能资源分配模型

Fig. 3 The minimum cost-maximum flow multi-skill resource allocation model

2.4 局部资源置换算法

由于多技能资源可以执行不同技能,因此当作业调度完成时,通过调整各资源使用峰值点的各作业的资源分配情况,更新各资源的峰值,最大限度达到降低峰值的目的.局部资源置换算法步骤如下:

(1) 确定资源 K 使用量的峰值 $\overline{R}_K$ 和峰值点 t,得到在此时刻下调度的作业集合 J_p 和其他资源的使用量值 R_K.

(2) 确定 J_p 中作业 j 所需技能集合 S_j,资源 K 分配执行作业 j 的 $n_{s'}$ 个技能 s' 及其集合 S'_j,$S'_j\subseteq S_j$,得到技能 s' 对应的资源集合 K'_s.

(3) 选择 K'_S/K 中 c_k 最小的资源 K' 分配给技能 s'，若 $n_{s'} \leqslant |\bar{R}_{K'} - R_{K'}|$，则将资源 K' 全部分配给技能 s'，转步骤(5). 若 $n_{s'} > |\bar{R}_{K'} - R_{K'}|$，跳转步骤(4).

(4) 将 $|\bar{R}_{K'} - R_{K'}|$ 个资源 K' 分配给技能 s'，剩下的技能 s' 仍由资源 K 执行，转步骤(5).

(5) 更新 t 时刻下资源 K 的使用量 R'_K 和峰值 $\bar{R}_K$，如果 $R'_K < \bar{R}_K$，则 $\bar{R}_K = R'_K$，转步骤(6).

(6) 对 J_p 中下一个作业 j' 进行资源置换，跳转步骤(2)；如果 J_p 所有作业更新完毕，则转步骤(7).

(7) 对下一个资源 K 进行置换分析，跳转步骤(1)；如果所有的资源都更新完毕，则更新目标函数值 A，判断是否降低，输出结果.

综上所述，基于全局作业影响的改进调度机制的遗传算法流程如图 4 所示.

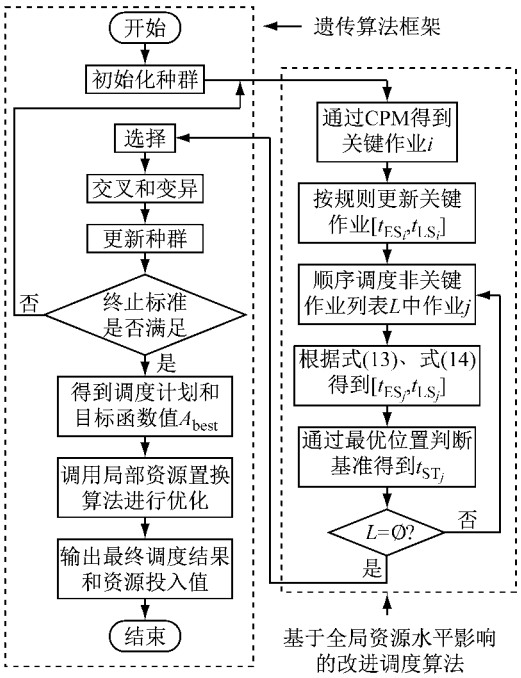

图 4　算法流程

Fig. 4　Flow chart

3　数值试验

为了验证本文设计的基于全局作业影响的改进调度算法(ISGS)求解 RIP 的有效性，选择测试问题库 PSPLIB 中的 10jobs，30jobs，60jobs，90jobs 算例进行数值试验，工期上限 $\bar{T}$ 设置为由 CPM 得出的 makespan 的 1.1，1.2，1.5 倍，并对算例进行改造，设定参数如下：资源种类 K，技能数 S，技能因素 E_{SF}，资源柔性度 F，其中 $K=4$，$S=3$，E_{SF} 为随机，$F=0.4$，同时将本文算法与 CPLEX(版本号为 IBM ILOG CPLEX Optimization Studio V12.6.3)和改进 Ranjbar 等[7]算法(HSGS)进行对比，数值试验在 C#(Visual Studio 2013)语言环境下编程实现，测试平台为 Intel Core i5 7th处理器，2.40 GHz 主频，4G 内存，结果如表 1～5 所示. 其中表 1 中：CPLEX 列、ISGS 列和 HSGS 列分别表示用该法求得算例的平均目标函数值；g_1 为 ISGS 与 CPLEX 数值之差除以 ISGS 值，g_2 为 HSGS 与 CPLEX 之差除以 HSGS 值. 表 2 中：每一列表示该法求解算例的平均运算时间. 表 3～5 中：ISGS 和 HSGS 列分别代表各算法所求目标函数值；g 为 HSGS 与 ISGS 之差除以 HSGS 值.

由表 1～5 的数据试验结果可见：在小规模算例试验时，本文设计的算法求得的最优解基本等于 CPLEX 得到的最优解，而对比算法却有较多算例与最优解有偏差. 在求解时间上本文算法与对比算法在同一个数量级范围内远远小于 CPLEX 的求解时间. 在大规模算例试验时，随着问题规模扩大，在保持 $\bar{T}$ 一定的情况下，本文算法与对比算法 HSGS 的求解数值间的 g 保持在 5%左右，证明本文算法在求解大规模 MSRIPSP 问题时更有效.

图 5 中，30 ISGS 表示在 30jobs 规模算例下使

表 1　10jobs 试验结果

Tab. 1　Results of 10 jobs

算例	$\bar{T}=1.1T_{cpm}$					$\bar{T}=1.2T_{cpm}$					$\bar{T}=1.5T_{cpm}$				
	CPLEX	ISGS	g_1/%	HSGS	g_2/%	CPLEX	ISGS	g_1/%	HSGS	g_2/%	CPLEX	ISGS	g_1/%	HSGS	g_2/%
1	54.0	54.0	0	54.0	0	54.0	54.0	0	54.0	0	38.0	38.0	0	38.0	0
2	58.0	59.0	1.7	59.0	1.7	57.0	57.0	0	58.0	1.8	47.0	47.0	0	47.0	0
3	65.0	66.0	1.5	66.0	1.5	64.0	64.0	0	65.0	1.6	57.0	59.0	3.5	59.0	3.5
4	76.0	76.0	0	76.0	0	67.0	67.0	0	68.0	1.5	62.0	63.0	1.6	63.0	1.6
5	51.0	51.0	0	51.0	0	51.0	51.0	0	51.0	0	43.0	43.0	0	43.0	0
6	62.0	63.0	1.6	64.0	3.2	60.0	61.0	1.7	61.0	1.7	58.0	58.0	0	58.0	0
7	60.0	60.0	0	62.0	3.3	58.0	59.0	1.7	59.0	1.7	53.0	53.0	0	53.0	0
8	57.0	57.0	0	59.0	3.5	51.0	51.0	0	52.0	2.0	37.0	37.0	0	37.0	0
9	67.0	67.0	0	67.0	0	66.0	66.0	0	66.0	0	59.0	59.0	0	60.0	1.7
10	68.0	68.0	0	68.0	0	63.0	63.0	0	63.0	0	57.0	57.0	0	57.0	0
平均值	61.8	62.1	0.5	62.6	1.3	59.1	59.3	0.3	59.7	1.0	51.1	51.4	0.5	51.5	0.7

表2 10jobs试验运算时间

Tab. 2 Operation time of 10 jobs

算例	$\overline{T}=1.1T_{cpm}$			$\overline{T}=1.2T_{cpm}$			$\overline{T}=1.5T_{cpm}$		
	CPLEX/s	ISGS/s	HSGS/s	CPLEX/s	ISGS/s	HSGS/s	CPLEX/s	ISGS/s	HSGS/s
1	152.9	10.7	4.8	230.8	16.8	6.8	284.5	35.3	22.3
2	73.7	5.6	6.1	122.3	6.9	5.9	145.1	10.9	8.5
3	126.4	8.4	8.8	149.2	9.6	8.1	166.6	15.1	10.1
4	85.9	5.4	4.2	176.5	6.4	4.6	156.2	10.7	7.8
5	100.2	7.7	8.4	93.0	9.7	7.9	161.2	12.2	9.6
6	48.8	4.1	1.9	87.5	4.6	4.0	134.0	7.5	5.1
7	65.9	5.8	3.4	90.5	6.3	9.0	117.9	14.7	9.4
8	91.5	6.3	2.1	102.9	7.8	5.6	185.4	12.1	9.5
9	43.3	2.5	1.1	52.3	2.7	2.4	83.7	4.9	4.5
10	17.6	1.3	0.7	13.1	1.4	1.6	41.2	2.5	2.2
平均值	80.6	5.8	4.2	111.8	7.2	5.6	147.6	12.6	8.9

表3 30jobs试验结果

Tab. 3 Results of 30 jobs

算例	$\overline{T}=1.1T_{cpm}$			$\overline{T}=1.2T_{cpm}$			$\overline{T}=1.5T_{cpm}$		
	ISGS	HSGS	g/%	ISGS	HSGS	g/%	ISGS	HSGS	g/%
1	95.0	101.0	5.94	94.0	96.0	2.08	77.0	82.0	6.10
2	81.0	84.0	3.57	79.0	82.0	3.66	70.0	74.0	5.41
3	86.0	92.0	6.52	85.0	87.0	2.30	67.0	73.0	8.22
4	83.0	87.0	4.59	78.0	83.0	6.02	66.0	72.0	8.33
5	100.0	105.0	4.76	97.0	102.0	4.90	78.0	84.0	7.14
6	80.0	86.0	6.98	76.0	82.0	7.32	73.0	78.0	6.41
7	83.0	86.0	3.49	80.0	84.0	4.76	70.0	70.0	0
8	90.0	90.0	0	79.0	81.0	2.47	77.0	79.0	2.53
9	100.0	111.0	9.91	97.0	101.0	3.96	88.0	92.0	4.34
10	107.0	115.0	6.96	100.0	110.0	9.09	91.0	97.0	6.19
平均值	91.0	96.0	5.27	87.0	91.0	4.66	76.0	80.0	5.47

表4 60jobs试验结果

Tab. 4 Results of 60 jobs

算例	$\overline{T}=1.1T_{cpm}$			$\overline{T}=1.2T_{cpm}$			$\overline{T}=1.5T_{cpm}$		
	ISGS	HSGS	g/%	ISGS	HSGS	g/%	ISGS	HSGS	g/%
1	130.0	138.0	5.79	126.0	133.0	5.26	107.0	112.0	4.46
2	146.0	152.0	3.95	129.0	138.0	6.52	104.0	112.0	7.14
3	140.0	144.0	2.78	128.0	129.0	0.78	107.0	113.0	5.31
4	130.0	137.0	5.11	112.0	121.0	7.44	102.0	108.0	5.56
5	156.0	163.0	4.29	155.0	157.0	1.27	122.0	132.0	7.58
6	138.0	145.0	4.83	128.0	136.0	5.88	108.0	122.0	11.48
7	146.0	152.0	3.95	130.0	137.0	5.11	112.0	121.0	7.44
8	140.0	149.0	6.04	134.0	139.0	3.60	126.0	133.0	5.26
9	120.0	120.0	0	106.0	116.0	8.62	100.0	104.0	3.85
10	143.0	148.0	3.38	124.0	136.0	8.82	112.0	118.0	5.08
平均值	139.0	145.0	4.01	127.0	134.0	5.33	110.0	118.0	6.32

用本文算法ISGS,30 HSGS表示在30jobs规模算例下使用对比算法HSGS,其他类似.在$\overline{T}=1.5T_{cpm}$的情况下,随着算例规模扩大,本文算法ISGS比对比算法HSGS求得的目标函数值好且g值维持在5%左右,并逐渐有扩大趋势.在同组规模算例试验中,随着$\overline{T}$的增加,更多的原本"并行"执行的作业可以"串行"执行,资源消耗自然减少,但是本文算法仍与对比算法保持稳定的g值,进一步验证了本文算法的有效性.

图6中,1.1 ISGS表示在$\overline{T}=1.1T_{cpm}$时使用本文算法ISGS,1.1 HSGS表示在$\overline{T}=1.1T_{cpm}$时使用对比算法HSGS,其他类似.在90jobs的算例试验中,在同组算例中随着$\overline{T}$变大,目标函数值逐渐变小,但是本文算法ISGS依然优于对比算法HSGS.

表 5　90jobs 试验结果

Tab. 5　Results of 90 jobs

算例	$\overline{T}=1.1T_{cpm}$			$\overline{T}=1.2T_{cpm}$			$\overline{T}=1.5T_{cpm}$		
	ISGS	HSGS	g/%	ISGS	HSGS	g/%	ISGS	HSGS	g/%
1	150.0	157.0	4.46	140.0	155.0	9.68	121.0	138.0	12.32
2	173.0	181.0	4.42	160.0	166.0	3.61	125.0	134.0	6.72
3	157.0	168.0	6.55	144.0	149.0	3.36	118.0	131.0	9.93
4	163.0	170.0	4.12	148.0	160.0	7.50	132.0	142.0	7.04
5	175.0	189.0	7.41	149.0	159.0	6.29	134.0	140.0	4.29
6	179.0	191.0	6.28	158.0	167.0	5.39	139.0	147.0	5.44
7	180.0	188.0	4.26	168.0	175.0	4.00	140.0	151.0	7.28
8	180.0	193.0	6.74	160.0	167.0	4.19	141.0	155.0	9.03
9	200.0	208.0	3.85	184.0	197.0	6.60	147.0	155.0	5.16
10	160.0	168.0	4.76	150.0	158.0	5.06	133.0	140.0	5.00
平均值	172.0	181.0	5.28	156.0	165.0	5.57	133.0	143.0	7.22

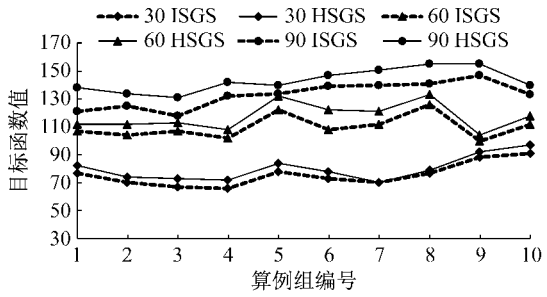

图 5　$\overline{T}=1.5T_{cpm}$ 时算法对比

Fig. 5　Algorithm comparison in $\overline{T}=1.5T_{cpm}$

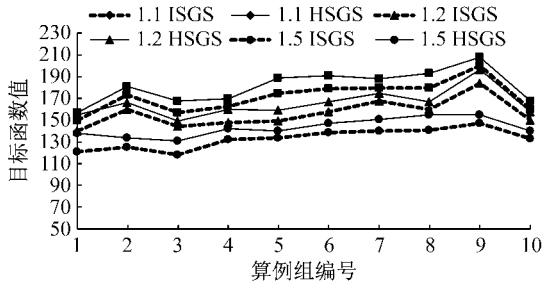

图 6　90jobs 算法对比

Fig. 6　Algorithm comparison in 90jobs

4　总结与展望

在考虑实际工业生产线中存在多技能工人的背景下，以生产投入成本最小为目标，抽象出考虑多技能可更新资源的资源投入项目调度问题，并建立了以项目资源投入成本最小化为目标函数的数学模型. 本文算法以遗传算法为框架，对非关键作业和 ΔT 进行编码，解码时提出了基于全局作业影响的改进调度算法，确定非关键作业的最优调度位置，并结合最小费用最大流模型解决资源分配问题，在得到调度方案的基础上通过局部资源调整策略进行进一步优化，从而得到最佳目标函数值. 算例试验数据表明本文算法无论在给定项目工期下不同规模算例试验还是在同种规模算例下不同项目工期时都具有较好的有效性.

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] DREXL A, KIMMS A. Optimization guided lower and upper bounds for the resource investment problem[J]. Journal of the Operational Research Society, 2001, 52(3): 340.

[3] RODRIGUES S B, YAMASHITA D S. An exact algorithm for minimizing resource availability costs in project scheduling[J]. European Journal of Operational Research, 2010, 206(3): 562.

[4] YAMASHITA D S, ARMENTANO V A, 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] AFSHAR N B. Multi-mode resource availability cost problem with recruitment and release dates for resources[J]. Applied Mathematical Modelling, 2014, 38(21): 5347.

[7] 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.

[8] BELLENGUEZ O, NÈRON E. Lower bounds for the multi-skill project scheduling problem with hierarchical levels of skills [C]//International Conference on the Practice and Theory of Automated Timetabling. Springer: Berlin Heidelberg, 2004: 229-243.

[9] LI H, WOMER K. Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm[J]. Journal of Scheduling, 2009, 12(3): 281.

（下转第 1730 页）