计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2022,31(9):342−351 [doi: 10.15888/j.cnki.csa.008719]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

转包商选择与单机批调度联合优化①

唐文娜, 刘　乐

(济南大学 商学院, 济南 250002)
通信作者: 刘　乐, E-mail: sm_liul@ujn.edu.cn

摘　要: 面向多转包商的外包选择是当今制造企业面临的一大运营管理挑战. 它与内部作业调度的一体化决策对企业降本增效至关重要. 本文针对有多个转包商可供作业外包选择的单机批调度联合优化问题, 在外包总成本和外包作业最晚交付期均受上限前提下建立以最小化外包总成本与内部批加工总成本之和为目标的 0-1 整数规划模型, 并为该问题设计了改进型遗传算法和贪婪算法. 以某陶瓷企业的外包与批调度联合决策场景为实例, 对比分析了这两种算法的求解性能, 发现改进型遗传算法在求解质量和时间上具有比较优势. 在模型的灵敏度实验分析中发现, 外包作业最晚交付期对作业运营总成本有显著影响, 而作业外包总成本的给定上限值对作业运营总成本的影响不显著.

关键词: 调度优化; 外包; 多转包商; 批处理机; 遗传算法; 批调度

引用格式: 唐文娜,刘乐.转包商选择与单机批调度联合优化.计算机系统应用,2022,31(9):342–351. http://www.c-s-a.org.cn/1003-3254/8719.html

Joint Optimization of Subcontractor Options and Single-machine Batch Scheduling

TANG Wen-Na, LIU Le

(Business School, University of Jinan, Jinan 250002, China)

Abstract: Outsourcing with multiple subcontractors is a major operational management challenge for today's manufacturing firms. The joint decision-making between outsourcing options and in-house scheduling is crucial to the cost reduction and efficiency increase of these firms. To jointly optimize single-machine batch scheduling with multiple subcontractors available for job outsourcing, this study constructs a 0-1 integer programming model, the objective of which is to minimize the sum of total outsourcing cost and total in-house batch processing cost under the premise that both the total outsourcing cost and the latest leading time for outsourcing jobs are subject to upper limits. An improved genetic algorithm and a greedy algorithm are also designed for joint optimization. The study takes the joint decision-making scenario of outsourcing and batch scheduling in a ceramic enterprise as an example and compares the solution performance of the two algorithms. The improved genetic algorithm shows its comparative advantages in terms of solution quality and efficiency. The results of a sensitivity experiment show that the latest leading time for outsourcing jobs has a significant impact on the total operating cost, while the upper limit of the total outsourcing cost does not significantly influence the total operating cost.

Key words: scheduling optimization; outsourcing; multi-subcontractors; batch processing machine; genetic algorithm; batch scheduling

为应对全球化竞争加剧、原材料价格上涨以及外部经济环境的不确定性, 外包 (outsourcing) 成为众多制造型企业不可或缺的运营策略[1]. 但外包策略的执行会增加企业作业调度的难度, 决策者一方面需要从现

① 基金项目: 国家自然科学基金 (71501083); 山东省自然科学基金面上项目 (ZR2020MG007); 中国博士后科学基金面上项目 (2019M662296); 济南大学社科类校级项目 (19YB03)
收稿时间: 2021-12-28; 修改时间: 2022-01-29; 采用时间: 2022-02-18; csa 在线出版时间: 2022-06-16

有待加工作业中确定外包作业, 另一方面还需对内部作业进行有效调度. 近年来, 带外包选择的调度问题 (scheduling problems with outsourcing options, SPOO) 引起了调度学者的关注. 现有针对该问题的研究成果中, 绝大多数都把单个转包商作为作业外包选择的对象. 不过, 企业实际运营中外包选择的对象往往是多个转包商, 并且随着企业经营范围的扩大会吸引越来越多的转包商参与其作业外包的竞争. 因此, 面向多转包商的作业外包选择才是企业外包运营管理的常态. 比起仅有单一转包商可供选择的情况, 企业在面对多个转包商时作业外包的选项更多, 优化决策的难度也更大. 如何得到作业的转包商选择与内部调度联合决策方案已成为当今企业外包运营管理的一项挑战性任务. 为此, 本文瞄准面向多转包商的可外包作业调度问题开展理论与方法研究. 这类问题不仅能为企业外包选择决策提供有效的解决方案, 还能帮助企业有效利用外包市场中的产能资源, 具有实践指导意义和应用价值.

带外包选择的调度问题的研究涌现出了丰硕的研究成果. 这些成果已经涉及单机[2]、并行机[3]、流水车间[4]、作业车间[5] 等多种机器环境. 带多转包商外包选择的调度问题在带外包选择的调度问题的研究中占比较少. Chen 等[6] 较早关注到带多转包商外包选择的调度问题, 他们对多转包商参与作业外包的并行机调度问题进行了深入研究, 并开发一种启发式算法求解该问题. Mokhtari 等[7] 也注意到带多转包商外包选择的并行机调度问题, 构建了以总成本最小化为目标的数学模型, 并为该问题设计出一种团队过程算法. Ahmadiza 等[8] 研究了面向双转包商的双机流水车间调度问题, 构建了以完工时间和外包及运输成本之和最小化为目标的数学模型, 并提出一种蚁群算法求解该问题. Goli 等[9] 研究了带多转包商外包选择的流水车间调度问题, 通过考虑不同机器上加工作业工时的不确定性, 提出了一种鲁棒混合整数线性规划模型, 并通过实例验证了所构建模型的鲁棒性.

在并行批加工 (parallel batch processing) 环境下, 批处理机能同时加工多个作业, 同一批作业中最长的加工时间为此批作业的加工时间[10]. 该环境能帮助企业提高机器利用率、降低能耗, 很多产品加工过程中的瓶颈工序都是在该环境加工完成[11]. 并且在半导体制造、金属冶炼、陶瓷烧制等行业中十分常见[12]. 从实际调研的情况来看, 在采用并行批加工模式的企业中, 加工时间相对较长的作业无论分配到哪个加工批次都会延迟同批次其他作业的完工时间并影响企业的整体交付服务水平. 在该情形下, 企业可考虑把这类作业分配给合适的转包商进行委外加工. 这样做既可以提高企业内部产能的利用效率, 又有助于缩短企业的完工期, 提升其交付服务水平. 可见, 并行批加工环境对外包策略的有效运用有着迫切的需求. 基于此, 本文针对并行批加工环境下带转包商选择的单机批调度问题 (single-machine batch scheduling problem with subcontractor options, SBSP_SO) 开展研究, 结合批处理机容量约束以及外包环节中的外包成本预算约束、外包作业最晚交付期约束构建了整数规划模型, 并设计改进型遗传算法和贪婪算法求解该模型. 最后, 通过对实例进行灵敏度实验分析, 为企业外包运营管理提供相关建议.

1 问题描述及数学模型

1.1 问题描述

待加工作业集合 J 中有 n 个作业亟待制造商实施联合调度, 这些作业均在 0 时刻可用. 现有两种生产资源可供选择, 可在其内部单台批处理机上加工或是外包给转包商. 现有外包市场上可供制造商选择的转包商有 H 个. 如果作业 J_j 外包给转包商 S_h, 制造商则须向转包商 S_h 支付 o_{jh} 单位的外包成本, 转包商 S_h 完成对作业 J_j 的加工后会在 l_{jh} 时刻交付给制造商. 制造商需要确定外包给转包商 S_h 的外包作业集和内部作业集 I 在内部批处理机上的调度方案 B.

本文所考虑的 SBSP_SO 问题可描述如下. 在满足外包成本预算约束、外包作业最晚交付期约束、机器容量约束的前提下, 得到最佳联合方案$\pi^*=[\cup_{h=1}^{H}O_h^{\pi^*}, B^*]$. 其中, $O_h^{\pi^*}$表示π^*中外包给转包商 S_h 的外包作业集, 以使得作业外包总成本 OC 与内部批加工总成本 IBC 之和 (以下简称作业运营总成本) 达到最小. 根据 Graham 等[13] 提出的调度问题三参数法和 Qi[14] 提出的外包与调度联合优化问题简记法, 本文所考虑的 SBSP_SO 问题可表示为 $1+H \mid batch, s_j \leqslant Q, Budget \mid OC+IBC$.

1.2 假设条件

(1) 每个作业的尺寸大小、内部加工时间均已知.

(2) 每个作业的尺寸都不超过批处理机的容量.

(3) 所有作业均在 0 时刻可用.

(4) 内部的批处理机采用并行批加工模式.

(5) 批处理机同一时刻仅能处理一个批次, 仅当一个批次中所有作业都完工时才释放机器.

(6) 内部加工批次总数仅当所有批次都构建完成后才能确定.

(7) 转包商根据自身产能和市场条件主动将外包成本报价和交付时间信息提供给制造商.

1.3 符号定义

表 1 列出了 SBSP_SO 问题中若干符号表示及其说明.

表 1 符号表示及其说明

类型	符号	描述
集合	J	作业集合, $J=\{J_1,J_2,\cdots,J_n\}$
	S	转包商集合, $S=\{S_1,S_2,\cdots,S_H\}$
	B	内部加工批次集合, $\boldsymbol{B}=\{B_1,B_2,\cdots,B_k\}$
参数	j	作业编号, $j\in\{1,2,\cdots,n\}$
	h	转包商编号, $h\in\{1,2,\cdots,H\}$
	k	批次编号, $k\in\{1,2,\cdots,n\}$
	λ	内部单台批处理机单位时间的批加工成本
	n	作业数量
	s_j	作业尺寸
	p_j	内部加工时间
	o_{jh}	转包商S_h对作业J_j的外包成本报价
	l_{jh}	转包商S_h对作业J_j的交付时间
	Q	批处理机容量
	$Budget$	外包成本预算
	D	外包作业最晚交付期
变量	y_k	如果内部生产批次B_k存在 (即所含作业数不为0), 则y_k=1, 否则为0
	x_{jk}	如果作业J_j不在生产批次B_k中, 则x_{jk}=0, 否则为1
	v_{jh}	如果作业J_j在转包商S_h处加工, 则v_{jh}=1, 否则为0

1.4 数学模型

根据以上描述, 本文考虑的 SBSP_SO 问题的数学模型如下.

$$\min \lambda\cdot\sum_{k=1}^{n}\max_{1\leqslant j\leqslant n}\left\{x_{jk}\cdot p_j\right\}+\sum_{h=1}^{H}\sum_{j=1}^{n}o_{jh}\cdot v_{jh} \tag{1}$$

$$\text{s.t.}\ \sum_{k=1}^{n}x_{jk}+\sum_{h=1}^{H}v_{jh}=1,\quad j=1,2,\cdots,n,\ h=1,2,\cdots,H \tag{2}$$

$$x_{jk}\leqslant y_k,\quad j=1,2,\cdots,n,\ k=1,2,\cdots,n \tag{3}$$

$$\sum_{j=1}^{n}x_{jk}\cdot s_j\leqslant Q\cdot y_k,\quad j=1,2,\cdots,n,\ k=1,2,\cdots,n \tag{4}$$

$$\sum_{h=1}^{H}\sum_{j=1}^{n}o_{jh}\cdot v_{jh}\leqslant Budget \tag{5}$$

$$\sum_{h=1}^{H}v_{jh}\cdot l_{jh}\leqslant D,\quad j=1,2,\cdots,n \tag{6}$$

$$x_{jk},y_k,v_{jh}\in\{0,1\},\quad j=1,2,\cdots,n,\ k=1,2,\cdots,n,\ h=1,2,\cdots,H \tag{7}$$

其中, 式 (1) 为 SBSP_SO 问题的目标函数; 式 (2) 排除了一个作业参与外包的同时又在制造商内部分批加工的可能性, 并保证当一个作业在制造商内部加工时只能安排到一个加工批次中, 当它参与外包生产时则仅能委托给一个转包商加工; 式 (3) 确保每个内部作业所在的加工批次一定不是空作业集; 式 (4) 保证每个内部加工批次中作业的尺寸之和不超过批处理机的容量Q; 式 (5) 表示作业外包总成本不能超过给定外包成本预算 $Budget$; 式 (6) 表示各转包商对其加工的每个外包作业的交付时间不得超过事先预定的外包作业最晚交付期D; 式 (7) 指明了决策变量 x_{jk}、y_k 和 v_{jh} 的二元属性.

SBSP_SO 问题的计算复杂性可通过一类特殊情形 A_1 获知. 其中, 各转包商 S_h (h=1, 2, ⋯, H) 对每个作业 $J_j\in J$ 的外包成本报价 o_{jh} 均相等且远远大于 $\lambda\cdot\sum_{j=1}^{n}P_j$. 由于作业的外包成本足够高, 导致无任何作业参与外包, 于是此时 SBSP_SO 问题退化为差异尺寸作业在单台批处理机上的调度问题, 即 $1\mid batch, s_j\leqslant Q\mid C_{\max}$ ($C_{\max}$ 表示最大完工时间). 该问题已被证明是强 NP-hard 问题[15], SBSP_SO 问题的计算复杂度不会低于该问题, 故 SBSP_SO 问题是强 NP-hard 问题.

SBSP_SO 问题的求解过程包含两个决策环节: 一是确定出外包给各个转包商的作业集合, 即作业外包决策; 二是对制造商内部批处理机上加工的作业进行分批调度, 即内部批调度决策. 作业外包决策是 SBSP_SO 问题的首要环节, 包含外包与否和转包商选择两个任务. 事实上, 可根据下列性质直接判断符合特定条件的作业是否参与内部分批调度.

性质 1. 如果作业 J_j 满足不等关系$o_{jh}>\lambda\cdot p_j$ ($h\in\{1,2,\cdots,H\}$), 则它在最优联合方案$\pi^*=[\bigcup_{h=1}^{H}O_h^{\pi^*},B^*]$中不会分配给转包商 S_h 完成加工, 即 $J_j\notin O_h^{\pi^*}$.

性质 2. 如果作业 J_j 满足不等关系$\min\left\{l_{jh};h=1,2,\cdots,H\right\}>D$, 则它在最优联合方案$\pi^*=[\bigcup_{h=1}^{H}O_h^{\pi^*},B^*]$

中参与内部分批调度, 即 $J_j \in B^*$.

以上性质均可通过反证法证明, 鉴于篇幅, 证明过程不在此详述.

2 改进型遗传算法和贪婪算法

本节为具备强 NP-hard 特性的 SBSP_SO 问题分别设计了改进型遗传算法 (improved genetic algorithm, IGA) 和贪婪算法 (greedy method, GM). 本节先对 IGA 算法中染色体编码与初始种群生成、适应度函数与选择机制、交叉算子、变异算子和迭代终止条件这 5 个重要组成部分进行逐一详述, 然后给出 IGA 算法的总体流程. 最后对 GM 算法的设计思路和执行步骤进行表述.

2.1 染色体编码与初始种群生成

根据 SBSP_SO 问题的描述, 联合方案π须给出外包给转包商 S_h 的作业集和内部批处理机上的调度方案. 任意一个作业 $J_j \in J$, 首先要决定其是否外包, 若是外包, 则须为其寻找合适的转包商; 若是在内部批处理机上完成加工, 则须确定其所属的内部加工批次. 因此, 在 IGA 算法中, 对染色体编码时需要兼顾单个作业可选择的外包对象与内部加工批次. 其中, 每条染色体表示一个可行联合方案π, 为每个作业设置两个基因位, 分别用 g_1 和 g_2 表示. g_1 表示作业可选择的外包对象, g_1 可能取值为{0, 1, 2, ⋯, H}, g_1=0 表示作业在内部单台批处理机上加工, g_1=h 则表示外包给转包商 S_h 加工; g_2 表示作业的内部加工批次, 考虑到一批作业的尺寸大小均接近 Q 的极端情况, g_2 可能的取值为{0, 1, 2, ⋯, n}, 每条染色体的长度为 2n. 任意一个作业要么参与外包, 要么在制造商内部加工, 故它的两个基因位不能同时取 0 且规定当 g_1 不等于 0 时 g_2 一定等于 0.

以一个包含 8 个作业、3 个转包商的问题为示例, 各作业的信息 (即内部加工时间 p_j、尺寸大小 s_j) 和转包商的信息 (即外包成本报价 o_{jh}、交付时间 l_{jh}) 如表 2 所示. 此外, 示例中统一取 Q=10, $Budget$=7, D=15. 图 1 为该示例的一个可行联合方案的染色体表示结构图. 其中, J_4 外包给转包商 S_2, OC=5<$Budget$=7, l_{42}=13<D=15; 内部加工批次分为 5 个批次, J_2、J_8 属于同一批次, J_1、J_7 属于同一批次, J_3、J_5、J_6 分别单独属于一个批次, 各内部加工批次中作业的尺寸之和也未超过机器容量 Q.

表 2 示例中作业与转包商的信息

编号	p_j	s_j	S_1		S_2		S_3	
			o_{j1}	l_{j1}	o_{j2}	l_{j2}	o_{j3}	l_{j3}
J_1	4	1	5	13	7	10	5	13
J_2	5	5	8	8	7	10	7	10
J_3	1	4	7	10	6	11	8	8
J_4	8	8	5	13	5	13	7	10
J_5	7	7	7	10	8	8	8	8
J_6	1	9	8	8	6	11	6	11
J_7	8	8	8	8	8	8	7	10
J_8	8	5	8	8	6	11	8	8

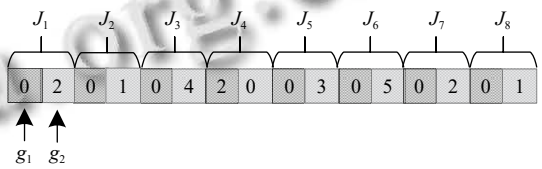

图 1 染色体表示结构示意图

IGA 算法中的每一代种群由 $popSize$ 条染色体组成, 每条染色体包含 2n 个基因位. 初始种群中的每条染色体均在每个基因位的有效范围内随机生成. 为进一步提高染色体的质量, 在染色体的生成时充分结合性质 1 和性质 2, 将满足性质 1、性质 2 中不等式条件的作业从外包作业集中排除.

2.2 适应度函数与选择机制

对于编码后的染色体, 在计算其适应度值之前需要判断它的合法性. 染色体合法性的判断需考虑以下 4 个方面:

(1) 外包成本预算约束 $Budget$.

(2) 外包作业最晚交付期约束 D.

(3) 机器容量约束 Q.

(4) 每个作业的两个基因位不能同时为 0 且至少有一个为 0.

文中规定不满足上述约束或条件的染色体为非法染色体, 记其适应度值为 0; 反之, 将合法染色体转化为联合方案 3, 并按式 (1) 计算联合方案 3 的目标函数值 $z(\pi)$. 染色体的适应度值按式 (8) 计算:

$$fitness(\pi)=\begin{cases}0, \text{如果染色体不合法}\\ \dfrac{M-z(\pi)}{M}, \text{如果染色体合法}\end{cases} \tag{8}$$

其中, $M=\lambda\cdot\sum_{j=1}^{n}p_j$.

本文采用结合精英保留策略的锦标赛选择机制.

该机制既保留了锦标赛选择简单易行、无需对所有适应度值进行有序排列的优点, 又能防止当前种群中的最优个体的基因在下一代丢失, 确保遗传算法随着迭代次数的增加一直收敛逐渐靠近最优解[16]. IGA 算法中选择操作的具体步骤如下:

步骤 1. 从上一代完成交叉、变异操作的种群 $Pop(t-1)$ 中选择适应度值最大的 *elitismCount* 个染色体直接进入当代种群 $Pop(t)$.

步骤 2. 从上一代种群 $Pop(t-1)$ 中随机选取 *tournamentSize* 个染色体进入选择池 *tournament*.

步骤 3. 将选择池 *tournament* 中的个体按其适应度值从小到大排序, 从中选出适应度值最大的个体, 置于当代种群 $Pop(t)$ 中.

步骤 4. 步骤 2、步骤 3 重复执行 *popSize*–*elitismCount* 次, 直到形成 $Pop(t+1)$.

2.3 交叉算子

在上述染色体的编码结构中, 因每个作业对应两个基因位, 在执行交叉操作时需要两个基因位一起操作. 为了保证执行交叉操作后能以相对较高的概率获得合法染色体, 在 IGA 算法中借鉴文献 [17] 中的均匀交叉方法实施逐代的染色体交叉操作. IGA 算法中交叉操作的具体实施步骤如下.

首先, 从本代种群 $Pop(t)$ 中随机选择 2 个亲代染色体, 分别表示为 Parent1、Parent2. 然后, 判断是否满足交叉实施条件. 若交叉实施条件满足, 即 $p_c>random(a)$ (p_c 表示交叉率, $random(a)\in[0, 1]$), 为交叉实施后的后代染色体 offspring 分配 $2n$ 个基因位, offspring 中每个作业对应的两个基因位来自于 Parent1 或者 Parent2 的概率各有 50%. 最后, 染色体 offspring 取代 Parent1 进入本代种群 $Pop(t)$ 中. 图 2 为上述示例的均匀交叉过程的示意图.

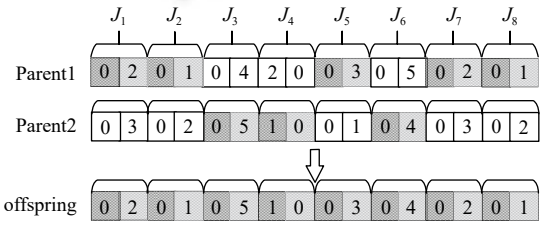

图 2 均匀交叉算子示意图

2.4 变异算子

为尽量保证执行变异操作后的染色体为合法染色体, 在 IGA 算法中选用均匀变异的方式实施逐代的染色体变异操作, 以一个较小的概率来替换染色体原有的基因对. IGA 算法中变异操作的具体实施步骤如下.

首先, 从本代种群 $Pop(t)$ 中按顺序选择出一个染色体, 记作染色体 1, 为保证有新的基因对进入现有种群, 按照初始种群中生成染色体的方式 (见第 2.1 节) 随机生成另一条染色体, 记作染色体 2. 对于染色体 1 中的每个基因对, 若满足当前的变异实施条件, 即 $p_m>random(b)$ (p_m 表示交叉率, $random(b)\in[0, 1]$), 则将其替换为染色体 2 中对应的基因对, 否则, 基因对的值保持不变. 由此得到的染色体作为变异后的染色体进入种群 $Pop(t)$. 图 3 为上述示例的均匀变异过程的示意图.

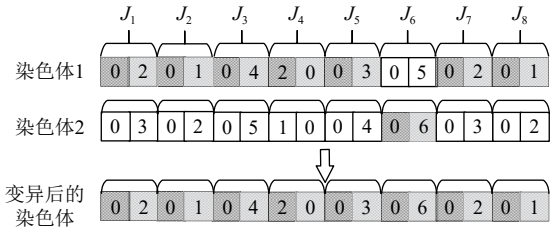

图 3 均匀变异算子示意图

2.5 终止条件

遗传算法中可选用的终止条件有最大迭代数或无改进最大迭代数终止、选取偏差度终止、评价准则终止等. 本文将无改进最大迭代数 (t_{max}) 作为所设计 IGA 算法的迭代终止条件, 令 $t_{max}=200n$. 具体操作如下. IGA 算法的每一次迭代中都记录并更新迄今最好解, 当迄今最好解的适应度值在连续的 $200n$ 次迭代中都未改进时 IGA 算法终止.

2.6 算法步骤

本文所设计的 IGA 算法的具体步骤如下所示.

步骤 1. 进行参数设置, 包括种群规模 *popSize*、交叉率 p_c、变异率 p_m、精英保留个数 *elitismCount*、锦标赛规模 *tournamentSize*, 并设定迭代终止条件.

步骤 2. 初始种群 $Pop(0)$ 的生成按照第 2.1 节中初始种群生成的相关内容执行, 设置变量 t, 其初始值设定为 1.

步骤 3. 先对种群 $Pop(t-1)$ 执行结合精英保留策略的锦标赛选择操作, 具体的步骤按第 2.2 节的相关内容执行.

步骤 4. 然后对选中的染色体进行均匀交叉操作,

具体的步骤按照第 2.3 节的相关内容执行.

步骤 5. 最后, 对选中的染色体执行均匀变异操作, 具体的步骤按照第 2.4 节的相关内容执行, 生成新的种群 $Pop(t)$.

步骤 6. t=t+1, 检验是否满足迭代终止条件, 即每一代记录的迄今最好解的适应度值是否在 200n 代内未得到改进; 若不满足迭代终止条件, 则返回步骤 3.

步骤 7. 输出联合方案π的目标函数值及求解时间、联合调度方案等信息.

2.7 贪婪算法

贪婪算法在求解复杂组合优化问题时, 通常不从整体最优解出发, 而总是基于当前条件做出最好的选择. 这种做法往往能在很多组合优化问题的求解中获得全局最优解或全局近似最优解[18]. 本节应用所设计的 GM 算法完成 SBSP_SO 问题中的外包与否和转包商选择两个任务, 从而得到内部作业集I'和面向每个转包商的外包作业集$O_h^{\pi'}$. 在外包作业集$\bigcup_{h=1}^{H} O_h^{\pi'}$确定后, 可基于 BFLPT (best-fit longest processing time) 规则得出面向内部作业集I'的批调度方案B'.众所周知, BFLPT 规则能为差异尺寸作业批调度问题快速提供一个近似最优批调度解[19], 文献 [19] 中给出了该规则的具体步骤, 在此不再赘述. 当外包作业集$\bigcup_{h=1}^{H} O_h^{\pi'}$和内部作业集$I'$均确定后, 可根据式 (1) 计算出批调度方案$\pi'$=$[\bigcup_{h=1}^{H} O_h^{\pi'}, B']$的目标函数值$z(\pi')$.

用于确定 SBSP_SO 问题中内部作业集I'和外包作业集$\bigcup_{h=1}^{H} O_h^{\pi'}$的 GM 算法步骤如下:

步骤 1. 对于任意作业$J_j \in J$, 按式 (1) 计算对应的比值R:

$$R_j = \frac{\lambda \cdot p_j}{\min\left\{o_{jh}; h = 1,2,\cdots,H\right\}} \tag{9}$$

将作业集 J 中的全部作业按照对应的R_j (j=1, 2, …, n) 值降序排序, 形成作业排序方案τ.并且, 执行变量初始化操作: e←1、OC←0.

步骤 2. 如果 e>n, 那么输出内部作业集I'和外包作业集$\bigcup_{h=1}^{H} O_h^{\pi'}$, GM 算法结束. 否则, 取出当前序列$\tau$中$R_j$值最大的作业$J_j$, 并找出对该作业外包成本报价最低的转包商$S_h$.

步骤 3. 确定转包商$S_{h'}$对作业$J_{j'}$的交付时间$l_{j'h'}$. 如果$l_{j'h'}$>D, 将作业$J_{j'}$归入I'中, 从序列τ中剔除作业$J_{j'}$, 置 e←e+1, 并返回步骤 2. 否则, 确定转包商$S_{h'}$对作业$J_{j'}$的外包成本报价$o_{j'h'}$.

步骤 4. 更新当前的 OC 值: 置 OC←OC+$o_{j'h'}$.如果 OC >$Budget$, 将作业$J_{j'}$归入I'中, 从序列τ中剔除作业$J_{j'}$, 置 e←e+1, 并返回步骤 2. 否则, 将作业$J_{j'}$归入$O_h^{\pi'}$中, 从序列τ中剔除作业$J_{j'}$, 置 e←e+1, 并返回步骤 2.

3 实例分析

下面以某陶瓷制造企业的作业外包与批调度联合决策场景为例进行仿真实验分析, 整个实验分为两部分. 在第 1 部分, 通过对比实验考察 IGA 算法的优化性能, 以验证 IGA 算法对 SBSP_SO 问题的求解有效性. 在第 2 部分, 旨在运用 IGA 算法对 SBSP_SO 问题实例中的可控参数进行灵敏度实验分析, 以探寻影响作业运营总成本的关键可控参数及其影响效果.

3.1 实例介绍

所选取的实例来自于某陶瓷企业中进行胚体烧制的批加工生产线. 经实地调研发现: 该生产线采用并行批加工模式, 生产线烧制所使用的电炉的容量 Q 为 25 m^3; 企业内部单台批处理机单位时间的批加工成本λ为 1; 外包市场上可供陶瓷厂选择的转包商有 4 个; 外包成本预算$Budget = \eta \cdot \sum_{j=1}^{n} \max\left\{o_{jh}\right\}$, 其中$\eta \in [0, 1]$, 被称作外包成本容许率. 所关注的联合调度决策场景如下所述: 该批加工生产线共有 35 个待烧制作业; 各作业的内部加工时间 p_j (以 h 为单位)、尺寸大小 s_j (以 m^3 为单位) 已知, 如表 3 所示; 各转包商对每个作业的外包成本报价 o_{jh} (以 RMB 为单位)、交付时间 l_{jh} (以 h 为单位) 也已知, 如表 3 所示. 由于该厂自身的产能有限, 难以按时完成所有待烧制作业的加工任务, 不得不考虑将一部分待烧制作业外包, 以牺牲一定数额的外包成本, 使作业运营总成本下降.为保证完工的陶瓷品能按时送达客户, 该企业限定各个转包商在 60 h 内完成对委外烧制任务的交付.

3.2 改进型遗传算法的性能实验分析

下面选用商业优化软件 IBM ILOG CPLEX 12.8 和 GM 算法作为比较对象, 考察所设计 IGA 算法对 SBSP_SO 问题的优化性能表现. 运行环境如下: 计算机系统为 Windows 10, 处理器为 Intel®Core™i7-7500@ 2.70 GHz 2.90 GHz, RAM 内存为 4.0 GB. 另外, 实验中用到的各种算法、程序均采用 Java 语言编程实现. 在预实验中, 通过对所设计 IGA 算法的控制参数进行校准实验, 得出各参数的建议取值如下: $popSize$=50, p_c = 0.95, p_m=0.01, $elitismCount$=5, $tournamentSize$=5.

表 3 实例中的作业与转包商信息

编号	p_j	s_j	S_1		S_2		S_3		S_4	
			o_{j1}	l_{j1}	o_{j2}	l_{j2}	o_{j3}	l_{j3}	o_{j4}	l_{j4}
J_1	14	8	38	24	28	34	18	44	10	52
J_2	12	6	26	36	23	39	11	51	18	44
J_3	20	16	14	48	28	34	16	46	10	52
J_4	10	6	10	52	6	56	10	52	8	54
J_5	12	14	6	56	4	58	14	48	11	51
J_6	10	8	22	40	25	37	16	46	17	45
J_7	24	16	6	56	16	46	20	42	9	53
J_8	16	14	13	49	14	48	28	34	12	50
J_9	9	6	6	56	4	58	18	44	14	48
J_{10}	25	16	15	47	12	50	12	50	13	49
J_{11}	18	8	8	54	18	44	15	47	6	56
J_{12}	19	10	20	42	18	44	6	56	12	50
J_{13}	7	4	14	48	13	49	16	46	10	52
J_{14}	17	12	20	42	20	42	4	58	16	46
J_{15}	15	10	6	56	24	38	32	30	28	34
J_{16}	26	16	15	47	20	42	9	51	12	50
J_{17}	11	10	6	56	10	52	8	54	8	54
J_{18}	14	14	20	42	30	32	28	34	16	46
J_{19}	11	4	15	47	16	46	13	49	20	42
J_{20}	6	8	26	36	26	36	34	28	28	34
J_{21}	13	4	32	30	18	44	28	34	12	50
J_{22}	10	12	19	43	14	488	12	50	16	46
J_{23}	27	16	24	38	30	32	34	28	20	42
J_{24}	12	8	20	42	8	54	14	48	10	52
J_{25}	9	4	22	38	14	48	20	42	20	42
J_{26}	13	12	6	36	10	52	6	56	8	54
J_{27}	10	3	16	46	2	60	4	58	7	55
J_{28}	6	5	4	58	5	57	3	59	4	58
J_{29}	11	6	5	57	7	55	5	57	8	54
J_{30}	15	8	11	51	10	52	16	46	15	47
J_{31}	10	6	13	49	15	47	16	46	14	48
J_{32}	13	8	14	48	19	43	17	45	21	41
J_{33}	15	12	8	54	6	56	5	57	11	51
J_{34}	13	9	5	57	7	55	8	54	10	52
J_{35}	16	10	13	49	17	45	15	47	16	46

当前性能实验中的 6 个测试算例源于第 3.1 节所述实例. 算例 1 中的待烧制作业数为 30, 这 30 个作业的相关信息如表 3 中前 30 个作业所示; 算例 2 中的待烧制作业数为 31, 这 31 个作业的相关信息如表 3 中前 31 个作业所示, 以此类推. 此外, 这 6 个算例中其他参数取值统一为: η=0.1, D=48 h , Q=25 m^3. 随着作业规模的增大, CPLEX 短时间内得不到精确解的可能性会提升. 为了便于比较 IGA 算法、GM 算法和 CPLEX 的求解结果, 在当前实验中将 CPLEX 求解时间上限设置为 3 600 s, 将迄今最好解的适应度值在连续 200n 次迭代中都未改进作为 IGA 算法的迭代终止条件. IGA 算法本质上属于元启发式算法, 在 IGA 算法的性能实验中需要记录独立运行多次的求解结果 (目标函数值和求解时间). 在本性能实验中, 为测算 IGA 算法和 GM 算法在求解能力上的差距, 特引入一个求解质量偏差度量指标 *Gap*, 两算法的 *Gap* 指标值按下式计算:

$$Gap_{HA}=\frac{z_{HA}(\pi)-z^*(\pi)}{z^*(\pi)}\cdot 100 \tag{10}$$

其中, *HA* 为 IGA 算法和 GM 算法的通称变量, 即 $HA\in$ {IGA, GM}. $z_{\mathrm{IGA}}(\pi)$ 是 IGA 算法独立运行 15 次所得目标函数值的均值, $z_{\mathrm{GM}}(\pi)$是 GM 算法所得解的目标函数值. $z^*(\pi)=\min\{z_{\mathrm{IGA}}(\pi),z_{\mathrm{GM}}(\pi)\}$.

表 4 给出的是改进型遗传算法、贪婪算法与 CPLEX 求解测试算例的统计结果. 其中, Obj1 和 State 分别表示 CPLEX 单独运行的求解结果、对应目标函数值的时间状态; Min、Max、AVG、SD、Time 和 *Gap* 分别为改进型遗传算法独立运行 15 次的最小值、最大值、均值、标准差、求解时间以及 Gap_{ABA}的值; Obj2 、Time 和 *Gap* 分别表示贪婪算法单独运行的求解结果、对应目标函数值的求解时间以及Gap_{GM}的值.

表 4 改进型遗传算法、贪婪算法与 CPLEX 的实例求解结果比较

编号	n	CPLEX		IGA						GM		
		Obj1	State	Min	Max	AVG	SD	Time (s)	*Gap* (%)	Obj2	Time (s)	*Gap* (%)
1	30	197	—	197	197	197.00	0.00	7.00	0	235	2.001	19.29
2	31	201	~	201	201	201.00	0.00	7.01	0	236	2.001	17.41
3	32	205	~	205	205	205.00	0.00	8.00	0	239	2.005	16.59
4	33	213	~	212	212	212.00	0.00	8.00	0	248	2.001	16.98
5	34	220	~	218	219	218.33	0.47	8.01	0	254	2.071	15.45
6	35	227	~	225	225	225.00	0.00	8.01	0	261	2.009	16.00

注: 其中, "—"表示获取的是 3 600 s 内的目标函数值, "~"表示获取的是 3 600 s 外的目标函数值.

由于 CPLEX 是基于数学规划法求解测试算例, 其耗时会随作业数 n 的增大而指数增长. 从表 4 可以看出, 对于作业数 $n\geqslant 31$ 的算例, CPLEX 在设定的 3 600 s 内无法得到精确最优解, 只能求得近似最优解. 而 IGA

算法却能在更短的时间得出更高质量的联合调度解, 运行时间仅为 CPLEX 所耗时间的 1/500. 在 n=30、31、32 这 3 个算例的求解中, IGA 算法全部 15 次实验中均得到和 CPLEX 相等的目标函数值. 在 n=33、34、35 这 3 个算例的求解中, 改进型遗传算法在全部 15 次实验中均达到或优于 CPLEX 求解结果水平. 可见, 与 CPLEX 相比, IGA 算法在测试算例上的优化质量和时间上的表现更好.

另外, 从 IGA 算法和 GM 算法对上述 6 个算例的求解结果来看, 虽然 IGA 算法比 GM 算法花费的时间多, 但是这些时间并不是徒劳的, 而是有效提升了解的质量. IGA 算法的 *Gap* 值均为 0, 而 GM 算法的 *Gap* 值均在 15% 以上. 这说明 IGA 算法的优化质量明显优于 GM 算法, 进一步体现出 IGA 算法的收敛能力.

3.3 实例的灵敏度实验分析

本节力图利用灵敏度实验分析法找出影响企业作业运营总成本的关键可控参数, 通过对实例中关键可控参数的有效调整, 使企业所花费的作业运营总成本进一步降低. 实例中, 转包商提供的外包成本报价 o_{jh} 和交付时间 l_{jh}、作业数 n 等属于不可控因素. 由于企业内部单台批处理机单位时间的批加工成本λ主要受人员工资、机器设备等影响, 在现实中改变的可能性不大, 也属于实例中的不可控参数. 实例中的可控参数主要有外包成本预算 *Budget* 和外包作业最晚交付期 D, 而外包成本预算 *Budget* 的实际值由外包成本容许率 η 决定. 因此, 针对该实例的灵敏度实验主要围绕外包成本容许率 η 和外包作业最晚交付期 D 展开.

下面将使用 IGA 算法进行实例的灵敏度实验, 基于上述实例、基于外 D 和 η 的不同组合 (D, η), 进而得到不同的作业运营总成本的值, 并将 IGA 算法独立运行 15 次的均值作为实验分析的数据. 其中, 作业运营总成本用 TC 表示, 外包作业最晚交付期上限用 $l_{\max}$ 表示. 考虑到企业预算不够充足更符合现实情况, 在进行灵敏度实验时, η 的范围取 [0.05, 0.5], 单位浮动数值为 0.05. 把 $l_{\max}$ 作为基准, 将提前 50%、25% 作为间隔点, 将 D 划分为早 (0, 30]、中 (30, 45]、晚 (45, 60] 三个阶段. 在 D 范围的选取上, 考虑到 D 过早在现实中难以实现, 在此选取 $D\in$[20, 60] 的范围进行实验, 单位浮动数值为 3. 图 4 给出的是不同参数组合 (D, η) 所对应的作业运营总成本 TC.

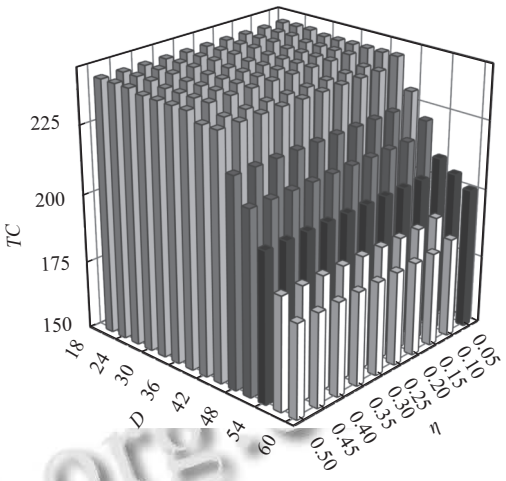

图 4 不同参数组合 (D, η) 下的作业运营总成本

由图 4 可发现以下 4 个现象: (1) 外包作业最晚交付期 D 对作业运营总成本 TC 的影响较为明显, 而外包成本容许率 η 对作业运营总成本 TC 的影响不明显; (2) 当 η>0.15 时, 无论 D 取何值, TC 的值几乎与 η=0.15 时 TC 的值持平. (3) 不建议企业将 D 定至早 (0, 30] 或中 (30, 45] 两个阶段. 究其原因在于外包作业最晚交付期的硬约束使得几乎所有作业在内部批处理机上加工, 企业无法从外包策略中获益. 四是在外包成本预算相对充足的情况下, D 的取值越接近 $l_{\max}$, TC 降低效果越明显. 当 D=54, 57, 60 h 时, 与早 (0, 30]、中 (30, 45] 两个阶段相比, TC 分别降低 15.70%、21.07%、23.97%.

为验证上述第一个现象是否有统计学意义, 取外包成本容许率 η 和外包作业最晚交付期 D 作为自变量, 作业运营总成本 TC 作为因变量, 使用上述灵敏度实验得出的数据进行双因素方差分析检验. 表 5 给出的是双因素方差分析的检验结果. 从表 5 可以看出, 外包成本容许率 η 和外包作业最晚交付期 D 组成的交互项的 p 值大于 0.05, 故没有统计学意义. 值得注意的是, 外包成本容许率 η 和外包作业最晚交付期 D 的 p 值分别为 0.991、0.000, 可认为外包成本容许率 η 的不同对 TC 没有显著影响, 而外包作业最晚交付期 D 对 TC 有显著影响.

下面针对外包作业最晚交付期D接近 $l_{\max}$ 的 4 种情况即 D=48, 51, 54, 57 h 观测并分析外包成本容许率 η 在 [0.01, 0.15] 范围内对作业运营总成本 TC 的影响趋势, η 的单位浮动值取 0.01. 首先, 运用所设计的 IGA 算法在 60 种参数组合 (D, η) 下分别独立运行

15 次. 然后, 统计出每种参数组合 (D, η) 所得的 15 个目标函数值的均值, 并将其绘于图 5 中.

表 5 双因素方差分析检验结果

指标	自由度	均方和	F值	p值
校正模型	29	1 496.010	16.372	0.000
截距	1	7 859 281.500	8 609.720	0.000
η	9	17.856	0.195	0.991
D	2	215.420	235.294	0.000
$\eta\times D$	18	12.376	0.135	1.000
误差	120	91.377	—	—
合计	150	—	—	—
校正合计	149	—	—	—

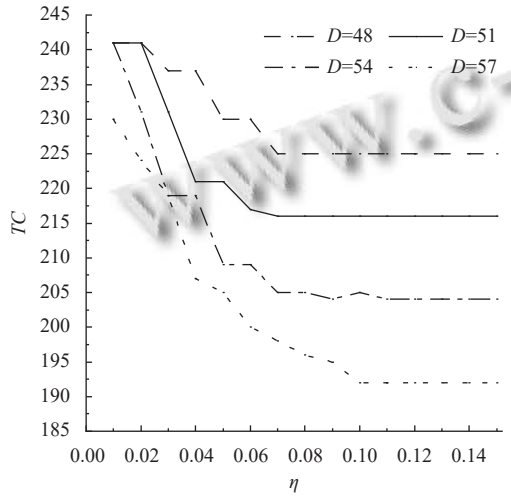

图 5 外包作业最晚交付期接近上限时外包成本容许率的变化对作业运营总成本的影响

从图 5 可以看出, 当 $\eta\leqslant 0.15$ 时, η 对 TC 的影响明显, 并且外包作业最晚交付期 D 越接近 $l_{\max}$, IGA 算法的作业运营总成本优化效果越显著. 此外, 在外包作业最晚交付期 D 的 4 种不同取值情况下, TC 都是随着 η 的增大而先快速下降后又保持稳定. 在 D 的 4 种不同取值情况下, η 的适宜取值范围也有所不同. 当 D=48, 51, 54 h 时, 建议外包成本容许率 η 的设定值不应低于 0.07. 当 D=57 h, η 的取值维持在 0.10 以上较优.

4 结论与展望

本文研究了带转包商选择的单机批调度问题. 通过考虑该问题的特征并结合相关优化性质设计了改进型遗传算法和贪婪算法. 从算法性能实验结果可知, 贪婪算法的求解质量偏差即 Gap 值均超过 15%. 这说明改进型遗传算法的优化质量明显优于贪婪算法. 在验证所设计改进型遗传算法有效性的基础上, 使用改进型遗传算法对实例中的可控参数进行灵敏度实验, 以找出关键可控参数并对其进行有效调整, 最大限度降低该企业的作业运营总成本, 为企业日常运营提供决策支持. 实例研究发现:

(1) 相较于外包成本容许率, 外包作业最晚交付期对作业运营总成本的影响更显著, 合理的外包作业最晚交付期设置可节省超过 20% 的作业运营总成本.

(2) 制造商不宜将外包作业最晚交付期设定太早, 否则无法从外包中获益, 在条件允许的情况下, 应让其设定值尽量接近外包作业最晚交付期上限.

(3) 外包成本容许率不宜低于 0.1, 制造商可通过融资或银行贷款等方式增加外包成本预算, 以确保降本效果显著.

在下一步的研究中, 力求在外包运营实践方面为企业提供更多的管理启示, 并进一步寻找该问题的应用场景, 在其他行业的相关实例中验证所设计改进型遗传算法的适用性.

参考文献

1 Wang SJ, Cui WL. Approximation algorithms for the min-max regret identical parallel machine scheduling problem with outsourcing and uncertain processing time. International Journal of Production Research, 2021, 59(15): 4579–4592. [doi: 10.1080/00207543.2020.1766721]

2 Lu LF, Zhang LQ, Zhang J, *et al*. Single machine scheduling with outsourcing under different fill rates or quantity discount rates. Asia-Pacific Journal of Operational Research, 2020, 37(1): 1950033. [doi: 10.1142/S02117595919500337]

3 Wang HB, Alidaee B. Unrelated parallel machine selection and job scheduling with the objective of minimizing total workload and machine fixed costs. IEEE Transactions on Automation Science and Engineering, 2018, 15(4): 1955–1963. [doi: 10.1109/TASE.2018.2832440]

4 Tirkolaee EB, Goli A, Weber GW. Fuzzy mathematical programming and self-adaptive artificial fish swarm algorithm for just-in-time energy-aware flow shop scheduling problem with outsourcing option. IEEE Transactions on Fuzzy Systems, 2020, 28(11): 2772–2783. [doi: 10.1109/TFUZZ.2020.2998174]

5 Safarzadeh H, Kianfar F. Job shop scheduling with the option of jobs outsourcing. International Journal of Production Research, 2019, 57(10): 3255–3272. [doi: 10.1080/00207543.2019.1579934]

6 Chen ZL, Li CL. Scheduling with subcontracting options. IIE Transactions, 2008, 40(12): 1171–1184. [doi: 10.1080/07408170801975057]

7 Mokhtari H, Abadi INK, Amin-Naseri MR. Production scheduling with outsourcing scenarios: A mixed integer programming and efficient solution procedure. International Journal of Production Research, 2012, 50(19): 5372–5395. [doi: 10.1080/00207543.2011.627687]

8 Ahmadizar F, Amiri Z. Outsourcing and scheduling for a two-machine flow shop with release times. Engineering Optimization, 2018, 50(3): 483–498. [doi: 10.1080/0305215X.2017.1325483]

9 Goli A, Tirkolaee EB, Soltani M. A robust just-in-time flow shop scheduling problem with outsourcing option on subcontractors. Production & Manufacturing Research, 2019, 7(1): 294–315. [doi: 10.1080/21693277.2019.1620651]

10 唐红涛, 杨志鹏, 刘家毅. 考虑工序并行的差异工件批调度研究. 工业工程, 2021, 24(3): 68–76, 114. [doi: 10.3969/j.issn.1007-7375.2021.03.009]

11 蒋小康, 张朋, 吕佑龙, 等. 基于混合蚁群算法的半导体生产线炉管区调度方法. 上海交通大学学报, 2020, 54(8): 792–804. [doi: 10.16183/j.cnki.jsjtu.2018.232]

12 吕顺风, 马科. 蚁群-鱼群混合算法在差异工件批调度中的应用. 计算机系统应用, 2018, 27(1): 162–167. [doi: 10.15888/j.cnki.csa.006136]

13 Graham RL, Lawler EL, Lenstra JK, *et al*. Optimization and approximation in deterministic sequencing and scheduling: A survey. Annals of Discrete Mathematics, 1979, 5: 287–326. [doi: 10.1016/S0167-5060(08)70356-X]

14 Qi XT. Coordinated logistics scheduling for in-house production and outsourcing. IEEE Transactions on Automation Science and Engineering, 2008, 5(1): 188–192. [doi: 10.1109/TASE.2006.887159]

15 Uzsoy R. Scheduling a single batch processing machine with non-identical job sizes. International Journal of Production Research, 1994, 32(7): 1615–1635. [doi: 10.1080/00207549408957026]

16 李雪, 何正文, 王能民. 不确定环境下基于时间、费用及鲁棒性权衡的多目标项目调度优化. 运筹与管理, 2019, 28(1): 6–16. [doi: 10.12005/orms.2019.0002]

17 Mokhtari H, Hasani A. An energy-efficient multi-objective optimization for flexible job-shop scheduling problem. Computers & Chemical Engineering, 2017, 104: 339–352. [doi: 10.1016/j.compchemeng.2017.05.004]

18 郑斐峰, 梅启煌, 刘明, 等. 基于遗传算法与贪婪策略的多港口集装箱配载研究. 运筹与管理, 2018, 27(5): 1–7. [doi: 10.12005/orms.2018.0104]

19 Dupont L, Dhaenens-Flipo C. Minimizing the makespan on a batch machine with non-identical job sizes: An exact procedure. Computers & Operations Research, 2002, 29(7): 807–819. [doi: 10.1016/S0305-0548(00)00078-2]

(校对责编: 牛欣悦)