计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2020,29(9):66−74 [doi: 10.15888/j.cnki.csa.007505]

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

考虑能耗与工时恶化作用下的并行机调度优化①

薛　聪[1,2], 郭　鹏[1,2], 陈　宓[1,2], 王丽敏[1,2]

[1](西南交通大学 机械工程学院, 成都 610031)
[2](轨道交通运维技术与装备四川省重点实验室, 成都 610031)
通讯作者: 郭　鹏, E-mail: pengguo318@swjtu.edu.cn

摘　要: 在热处理加工环境中, 工件温度随着其开工时刻的延误不断下降, 为了能够正常加工不得不保温或重新加热. 针对这一现象, 本文考虑了能耗与工时恶化作用下的并行机调度问题, 以最小化总拖期和能耗为目标构建了混合整数规划模型. 由于问题的复杂性, 提出了一种遗传变搜索算法, 其通过遗传操作获得变邻域搜索操作的解集, 而后使用变邻域结构进行寻优操作. 算例测试表明: 较之传统遗传算法以及数学规划器 Gurobi 的计算结果, 所提出的算法可以有效减少综合能耗和拖期成本.

关键词: 能源消耗; 并行机调度; 工时恶化; 遗传算法; 邻域搜索

引用格式: 薛聪,郭鹏,陈宓,王丽敏.考虑能耗与工时恶化作用下的并行机调度优化.计算机系统应用,2020,29(9):66–74. http://www.c-s-a.org.cn/1003-3254/7505.html

Parallel Machine Scheduling with Step-Deteriorating Jobs and Energy Consumption

XUE Cong[1,2], GUO Peng[1,2], CHEN Mi[1,2], WANG Li-Min[1,2]

[1](School of Mechanical Engineering, Southwest Jiaotong University, Chengdu 610031, China)
[2](Technology and Equipment of Rail Transit Operation and Maintenance Key Laboratory of Sichuan Province, Chengdu 610031, China)

Abstract: Jobs' temperature will decrease as their starting times delay in heat treating flow shop. In order to process the jobs normally, the worker has to reheat them or keep them in holding furnace. In response to this situation, this study considers a parallel machine scheduling problem with step-deteriorating jobs for minimizing the total tardiness and energy consumption. Firstly, a mixed integer programming model is proposed for the problem under study. Due to the intractability of the problem, a genetic -variable neighborhood search hybrid algorithm is designed to solve it. The algorithm use genetic operations to generate the solution, and then use the variable neighborhood search to improve the solution. The numerical tests show the proposed algorithm can efficiently reduce the total tardiness and energy consumption compared with standard genetic algorithm and mathematical model with standard solver Gurobi.

Key words: energy consumption; parallel machine scheduling; deteriorating job; genetic algorithm; variable neighborhood search

尿素造粒塔是尿素生产中的重要设备, 也是尿素生产流程中最后一个生产环节[1]. 其造粒原理是将成批次放置的满足加工温度要求的熔融尿素使用泵通过管道打到塔顶后喷出, 从而形成高温射流. 高温射流在下坠过程中通过空气冷却等过程快速断裂成滴, 自动凝成固体小颗粒, 从而形成颗粒状的尿素产品. 而不同批次的熔融尿素在等待被注入泵内的过程中, 其温度会随着等待时间的增加而降低. 当其开始加工时因为在

① 基金项目: 国家自然科学基金 (51405403); 中央高校基本科研业务费专项资金 (2682018CX09)
Foundation item: National Natural Science Foundation of China (51405403); the Fundamental Research Funds for the Central Universities of China (2682018CX09)
收稿时间: 2019-12-05; 修改时间: 2020-01-03; 采用时间: 2020-01-21; csa 在线出版时间: 2020-09-04

等待过程中的温度变化其温度可能不符合加工温度要求. 也就是说热加工环境中工件的温度会随着其开工时刻的增大而降低, 当温度降到一定数值以后, 将不再满足加工要求, 即工时恶化问题. 此时需要对不满足加工温度要求批次的熔融尿素进行二次加热, 或者监控其温度, 当发现其温度不满足加工温度要求时立即使用保温设施对其进行保温操作. 二次加热会增加此批次熔融尿素的加工时间和消耗额外的能量, 从而增大产品拖期和能耗, 造成产品延期交货和增大生产成本; 而使用保温设施保温虽然不会增加其加工时间但是会耗费更多的能耗成本. 当造粒塔从一台扩展到多台, 此时问题便扩展为考虑不同工作模式与工时恶化的并行机调度问题. 及时交货和能耗控制是生产成本控制的重要组成部分, 因此, 如何有效保证产品及时交货以及精准的控制能耗是管理人员需要解决的关键决策问题[2]. 在此, 本文以多台尿素造粒塔的尿素生产过程为背景, 从并行机加工环境出发, 对上述调度问题进行建模并设计优化算法进行求解.

并行机调度问题长期以来吸引了大量学者的关注和研究[3], 且节能调度方面的研究已有不少. 孟磊磊等以能耗最小化为目标, 提出了 5 个考虑关机/重启策略的不相关并行机调度模型[4]. 周炳海与顾佳颖提出了多目标免疫克隆选择算法处理多资源约束下的非等效并行机节能调度问题[5]. 雷德明等提出了新型帝国竞争算法去处理多目标低碳并行机调度[6]. 与本文研究最为相关的则是在并行机调度问题中同时考虑节能与拖期最小化. Li 等基于分派规则提出了 10 个启发式算法去优化能耗与总拖期[7]. 王永琦等结合问题的性质, 设计了适用于该问题的混合教学算法[8]. 然而上述研究均未考虑工件开工时间延误造成额外能耗或二次加热带来的工时恶化. 本文针对尿素厂造粒塔生产过程, 采用阶梯恶化函数对其生产过程进行描述[9], 并尽量减少生产过程中发生的能耗.

关于工时恶化的并行机调度问题亦开始受到关注, Ji 和 Cheng 考虑了工时线性恶化情况下的并行机调度问题, 并给出了机器数确定的情况下多项式近似算法[10]. Wang 等在此基础上进行了扩展, 考虑机器在开始部分不可用的情况, 并设计了启发式规则[11]. Guo 等为工时阶梯恶化的并行机设计了布谷鸟搜索算法, 在考虑工件间调整时间的基础上上最小化了总拖期[12], 此外 Guo 等还比较了不同的建模方式对问题求解效率的影响[13]. 变邻域搜索[14] 与集划分建模策略[15] 也相继被用于工时阶梯恶化的并行机调度问题. 在考虑处理成本和收益的阶梯恶化并行机调度问题中, Pei 等提出了变邻域搜索的混合算法去最大化净收益[16]. 遗传算法也被改进用于求解不相关并行机带恶化工件的调度问题[17]. 关于工时恶化的并行机调度问题在潜在机器扰动、学习效益与维护活动等方面均有扩展[18–20], 但文献 [18–21] 尚未涉及节能调度方面的拓展.

基于以上分析发现, 目前尚未出现同时考虑工时阶梯恶化与能耗的并行机调度优化研究. 现有并行机调度算法在处理实际约束时存在大规模算例计算时间过长、算法收敛速度慢、求解质量不高等问题, 且难以将工作模式选择考虑进去. 本文针对有温度要求的工件, 在等待加工的过程中有保温或者再加热两种方式, 均需消耗一定能量. 再加热还会导致其加工时间发生恶化. 以尿素厂造粒塔为研究背景, 提出以加权总拖期与总能耗之和最小化为调度目标的工时阶梯恶化并行机调度问题. 通过分析问题构建了混合整数线性规划模型, 由于问题的 NP-hard 特性, 提出了遗传变邻域混合搜索算法 (Genetic Algorithm-Variable Neighborhood Search, GA-VNS). 通过集成遗传算法 (Genetic Algorithm, GA) 与变邻域搜索 (Variable Neighborhood Search, VNS) 的优势有效实现了对问题的求解. 通过算例计算结果, 验证了该算法在解决较大规模算例时的效率.

1 问题描述与数学建模

给定机器集合$M=\{1,\cdots,m\}$, 其中 M 为包含所有机器的集合, m表示机器的总数, 给定工件集合$N=\{1,\cdots,n\}$, 其中 N 为包含所有工件的集合, n为加工工件的总数. 每个工件均可在任意一台机器上加工. 假设所有的工件在时刻 0 都是可用的, 即工件的准备时间为 0. 在加工过程中机器不会因其他因素而中断. 工件集 N 中的所有工件均需在机器上加工. 每台机器最多同时加工一个工件. 每个工件$j(j\in N)$都有相应的基本加工时间a_j, 交货期d_j, 恶化工期h_j. 如果工件 j 的开工时刻s_j早于恶化工期h_j, 则其实际加工时间p_j等于基本加工时间a_j. 在等待加工的过程中, 工件的温度会逐渐下降, 其开工时刻s_j也在逐渐增大. 而当工件j的开工时刻s_j晚于其恶化工期h_j时, 工件的温度将不再符合加工要求, 为了使工件温度满足加工要求, 有两种模式可以选择: (1) 对工件进行二次加热, 将发生额外的惩罚时

间b_j, 使得$p_j=a_j+b_j$, 并且还会有消耗额外的能量$(b_j\times q_j)$, 其中q_j为工件j加工时的单位时间能耗. 此时, 工件j的实际加工时间p_j是其开工时刻的阶梯函数, 额外消耗的能量为多出的加工时间内消耗的能量; (2) 将工件放在保温设施里对其进行保温处理, 等到其开始被加工时再取出 (将工件放进和取出保温设施的时间忽略不计). 保温模式可使工件j不增加额外的惩罚时间, 但使用保温设施对工件进行保温操作将会发生额外的能量消耗. 在保温模式下, 工件的加工时间仍等于其基本加工时间a_j, 额外的能量损耗e_j则可表示为:

$$e_j=e_0\times\max\{0,s_j-h_j\} \tag{1}$$

式中, e_0表示使用保温设备对工件进行保温操作时单位时间发生的能耗.

调度决策就是指派工件到各台机器, 同时选择恰当的工作模型以保证工件能够顺利完工, 以求最小化总拖期时间与加工能耗. 表 1 解释了本文数学模型中的变量和符号. 所提出的混合整数规划模型如下:

目标函数为:

$$F(X,Y,Z,U)=\alpha\sum_{j\in N}T_j+\beta\sum_{j\in N}E_j \tag{2}$$

约束条件为:

$$\sum_{k\in M}y_{jk}=1 \quad \forall j\in N \tag{3}$$

$$y_{ik}+y_{jk}\le 1+x_{ij}+x_{ji} \quad \forall i,j\in N,i\ne j,k\in M \tag{4}$$

$$c_i-s_j\le M(1-x_{ij}) \quad \forall i,j\in N \tag{5}$$

$$s_j+p_j\le c_j \quad \forall j\in N \tag{6}$$

$$c_j-d_j\le T_j \quad \forall j\in N \tag{7}$$

$$a_j\le p_j\le a_j+b_j \quad \forall j\in N \tag{8}$$

$$s_j\le h_j+M(1-z_j) \quad \forall j\in N \tag{9}$$

$$u_j\le 1-z_j \quad \forall j\in N \tag{10}$$

$$a_j+b_j(1-u_j-z_j)\le p_j \quad \forall j\in N \tag{11}$$

$$e_0\times(s_j-h_j)-M(1-u_j)\le e_j \quad \forall j\in N \tag{12}$$

$$e_j+p_j\times q_j\le E_j \quad \forall j\in N \tag{13}$$

$$x_{ij},y_{ik},x_{ik}\in\{0,1\} \quad \forall i,j\in\Omega,k\in\Omega \tag{14}$$

$$e_j,T_j,E_j,s_j\ge 0 \quad \forall j\in\Omega \tag{15}$$

上述模型中, 目标函数 (2) 为最小化所有工件的加权总拖期与总能耗之和, 其中α和β表示所有工件在加工过程中的总拖期和总能耗的加权系数, 在本文数学模型计算过程中, α和β均取 1. 因此该模型的优化目标为最小化所有工件的总拖期和总能耗之和. 约束条件 (3) 确保每个工件只能指派给一台机器. 约束条件 (4) 强调如果工件 i 和工件 j 被分配到同一台机器上加工, 则它们不能同时加工. 约束条件 (5) 定义了工件的开工时刻并确保工件 i 的完工时刻c_i、工件j的开工时刻s_j和x_{ij}的取值保持一致, 其中 M 为一个足够大的数, 取$M=\sum_{j\in N}a_j+\sum_{j\in N}b_j+\max\{d_j\}(j\in N)$. 约束条件 (6) 定义了工件 j 的完工时刻c_j. 约束条件 (7) 给出了工件j的拖期T_j, $T_j=\max(c_j-d_j,0)$. 约束条件 (8) 确定了工件 j 的实际加工时间p_j的取值范围. 约束条件 (9) 强调若工件j的开工时刻s_j早于或等于其恶化工期$h_j(p_j\le s_j)$, 则二进制变量$z_j=1$, 否则等于 0. 约束条件 (10) 确保当工件 j 的开工时刻s_j早于其恶化期h_j时, 不会再使用保温设施进行保温操作. 即当$z_j=1$时, $u_j=0$. 约束条件 (11) 确保当工件 j 使用保温设施保温时, 其实际处理时间p_j等于a_j. 约束条件 (12) 定义了当工件 j 使用保温设施时额外消耗的能量e_j. 约束条件 (13) 确定了工件 j 从时刻 0 开始到加工完成消耗的总能量E_j. 约束条件 (14) 和 (15) 为决策变量的取值约束.

表 1 数学模型中的符号

符号	含义
q_j	工件j在加工过程中的单位时间能耗
a_j	工件j的基本加工时间
h_j	工件j的恶化工期
d_j	工件j的交货期
s_j	工件j的开工时刻
p_j	工件j的实际加工时间
c_j	工件j的完工时刻
T_j	工件j的拖期
e_0	工件j使用保温设备保持加工温度条件时单位时间能耗
e_j	工件j在等待加工过程中的能耗
E_j	工件j的总能耗
M	足够大的正整数
y_{jk}	当且仅当工件j在机器k上加工时等于1, 否则为0
x_{ij}	当且仅当工件i的完工时刻小于或等于工件j的开工时刻等于1, 否则为0
z_j	当且仅当工件j的开工时刻小于等于其恶化工期时等于1, 否则为0
u_j	当且仅当工件j使用保温设备维持温度时等于1, 否则为0

2 遗传变邻域混合搜索算法

由于考虑工时阶梯的单机总拖期问题已被证明为

NP-hard 的[22], 本文所考虑的问题为并行机加工环境且考虑能耗, 因此其也是 NP-hard 的. 随着工件数的增加, 很难通过精确的算法获得问题的最优解. 在大规模问题求解寻找近优解时, 群集智能优化算法求解效率明显高于精确求解方法. 遗传算法能够利用群集效应进行寻优, 具有先天的并行搜索能力, 故本文利用遗传算法的这一特性进行大规模问题的求解. 前期计算结果表明, 若仅仅使用基本遗传算法求解本文所提并行机调度问题, 易陷入局部最优解且求解效率相对较差. 在此采用混合变邻域搜索策略来提高算法性能. 使用遗传算法经过一定迭代次数操作后的种群输出的最优染色体做为初始解, 进一步实施变邻域搜索, 以达到提高算法质量, 提升运算效率的目的. 因此, 本文针对所提并行机调度问题的特性, 集成考虑遗传算法和变邻域搜索的优势, 提出了基于遗传算法框架的遗传变邻域混合搜索算法.

2.1 编码和解码方式

编码是整个算法的第一步, 对算法效果有巨大影响. 本文的编码方式必须满足两方面的需求, 一是要确定机器的指派问题, 另一个则是要确定每台机器上工件的加工顺序. 本算法采用实数编码方式. 每个解序列为从 1 到$n+m-1$随机排序的序列. 其中大小从 1 到 n 的实数分别表示 n 个工件的编号. 大小从$n+1$到$n+m-1$的数字为分隔符来区分不同机器, 也就是说明其左边和右边的工件不在同一个机器上加工. 例如 6 个工件, 3 台机器的算例. 染色体序列(1,5,7,6,3,8,4,2) 表示机器一加工工件 1、5 且加工先后顺序为 (1, 5), 机器二加工工件 6、3, 且加工先后顺序为 (6, 3), 依次类推, 此序列解码结果如图 1 所示. 关于当工件开始加工时其温度不满足加工温度要求时的工作模式选择问题. 本文采取随机的策略选择其开工时刻超过恶化工期时的工作模式, 即当工件的开工时刻大于其恶化工期时, 其选择施加惩罚时间和使用保温设施保温的概率均为 50%.

2.2 遗传算法

2.2.1 遗传算法种群初始化

初始化种群是本文整个算法的开端, 其组成结构对整个算法影响很大. 本文采用随机生成的方式生成初始种群, 以保证染色体可取到所有的可能调度序列.

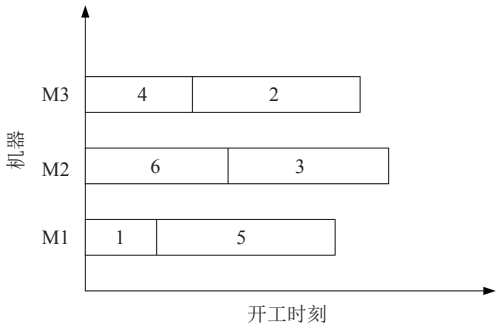

图 1 染色解码方式甘特图表示

2.2.2 适应度函数

模型目标是最小化所有工件的加权总拖期与总能耗之和, 因此本文的适应度函数为目标函数的倒数:

$$f(\pi_i)=1/(\alpha T(\pi_i)+\beta E(\pi_i)) \tag{16}$$

其中, π_i表示一个可行调度序列, $\alpha T(\pi_i)+\beta E(\pi_i)$表示其目标值.

2.2.3 选择策略以及精英保留

选择的作用是可以从当前群体中选出优良个体, 使得好的个体有更大的机会保留下来并将其优良的信息传递给下一代, 因而可以逐步地向最优解逼近. 本文的选择操作选用轮盘赌选择方法, 染色体被选中的概率 L 为:

$$L(\pi_i)=\frac{\alpha T(\pi_i)+\beta E(\pi_i)}{\sum_{i=1}^{p_{\text{num}}}\alpha T(\pi_i)+\beta E(\pi_i)} \tag{17}$$

在式 (17) 中, p_{num}为遗传算法部分的种群数量. 为防止产生新个体的同时破坏已有的优良个体, 本文采用了精英保留策略. 精英保留就是在一次迭代过程中对种群进行选择、交叉、变异等操作之后选出当前种群中适应值最大的个体, 直接遗传到下一代, 下一次迭代时不对其进行遗传操作. 目的是保留最好的染色体, 避免遗传算子破坏其优良性.

2.2.4 交叉操作

交叉操作是种群中产生新个体的主要步骤, 好的交叉方法对于遗传算法的效果也有显著影响. 由于顺序交叉能够在保留原有排列的基础上融合不同排列的有序结构单元, 本文选用两点顺序交叉 (类 OX), 顺序交叉的具体流程 (图 2) 如下:

步骤 1. 父代 1 和父代 2 配对, 从父代 1 中随机选择一段连续的基因, 并将这段基因按顺序放在子代 1

的起始部分;

步骤 2. 去除父代 2 中包含父代 1 被选基因段的基因;

步骤 3. 将父代 2 中去除父代 1 备选基因段后剩余的基因按顺序依次填入子代 1 的剩余位置. 此时, 获得了完整的子代 1 的基因.

同理, 按照以上步骤, 可从父代 2 中随机选取一部分基因段和父代 1 中的出去被选基因段部分, 按照上述步骤插入得到子代 2.

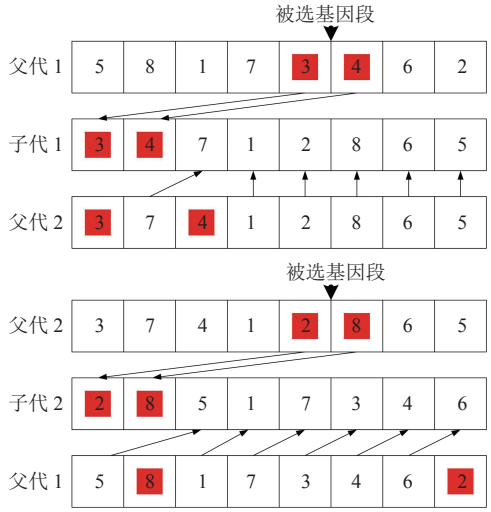

图 2 类 OX 交叉示意图

2.2.5 变异操作

当遗传算法通过交叉算子已接近最优邻域解时, 利用变异操作可以提高其随机搜索能力, 且可以有效维持群体多样性. 本文选用随机位置交换完成变异操作. 随机选中两个元素进行位置调换即完成变异操作. 如图 3 示例: 父代$P1(5,8,7,1,3,4,6,2)$. 选中第 3 个元素 7 和第 6 个元素 4 进行位置调换, 即生成子代$Q1(5,8,4,1,3,7,6,2)$.

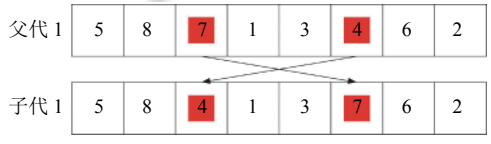

图 3 随机选择变异示意图

2.3 变邻域搜索

变邻域搜索算法[14] 通过搜索多个邻域结构能够较好的避免陷入局部最优解, 是一种改进型的局部搜索算法. 本文对使用遗传算法经过一定迭代次数操作后的最优染色体进行变邻域搜索操作, 以进一步改善解的质量.

变邻域搜索算法的工作原理是通过各种邻域结构对当前解实施变换, 以产生可行解集合. 目前关于调度问题的邻域结构大多数是基于插入和交换操作. 除了这些基本的邻域结构, 本文还引入了逆序操作. 因此, 本节使用插入和交换操作构造 4 种邻域结构, 并使用逆序定义了一种邻域结构. 以下将分别对这 5 种邻域结构进行介绍.

邻域搜索结构 (Neighborhood Search) NS1 和 NS2 原理相似. 两者均是其中利用交换对当前解实施微小的变动. 区别在于 NS1 是从整个解序列的第一个元素开始到最后一个元素结束, 依次随机选择一个与其不相等的元素进行交换, 也就是把解序列内的所有元素都与其他不重复的随机位置进行交换, 而 NS2 则是在解序列中随机选择一个元素, 将其插入到另一个不重复的随机选定的元素上. 在 NS1 和 NS2 的操作过程中只要解的质量比原来的解好, 立即接受其为当前解, 并重新开始搜索; 否则继续搜索直至所有可能的组合均已完成.

邻域搜索结构 NS3 和 NS4 有着类似的操作. 两者和 NS1, NS2 不同的是 NS3 和 NS4 所进行的交换操作是两两交换, 即同时交换的元素有 4 个, 成对交换. NS3 从解序列的前两个元素开始到最后两个元素结束, 依次选择两个元素与另外随机选择的两个不重复的元素分别进行交换. 与 NS1 和 NS2 的区别一样, NS4 则是随机的取两个元素与另外一对两个随机的不重复的元素分别进行交换. 其实施过程与 NS3 相同. 一旦发现给出的邻域解有所改善, 则将该邻域解视为当前解, 并再次开始搜索更好的解; 否则继续搜索直至搜索完成所有的邻域结构.

为了进一步提高算法的性能, 基于逆序操作的邻域结构 NS5 在上述 4 个邻域完成后实施. NS5 的具体操作为: 从当前解序列中随机选择两个元素, 将两者之间的工件全部逆序, 以形成新的解. 该步骤重复 $n+m-1$ 次以寻找较好的解. 与前 4 个邻域结构一样, 该邻域结构只接受效果更好的解.

2.4 GA-VNS 算法框架

通过使用 2.3 节的 5 种邻域搜索结构, 再结合 2.2 节的遗传算法为 VNS 算法提供初始解, 即可对本文提出的问题实施 GA-VNS 算法. GA-VNS 的终止条

件为: (1) 达到最大迭代次数T_p; (2) 当前解连续nip次迭代没有得到改善. 若满足上述两个条件中的任意一个, 则立即终止算法, 并返回已求得的最好解. 关于T_p和 nip 将在第 3 节参数调试部分确定其具体取值. GA-VNS 算法的程序架构如图 4 所示.

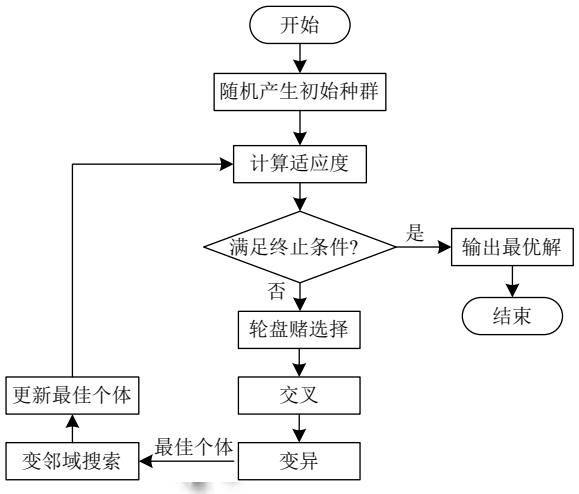

图 4 GA-VNS 程序框图

3 算例验证

本章采用两组算例验证算法的性能, 具体的参数取值见 3.1 节. 所提出的遗传变邻域混合搜索算法将在 3.2 节对算法关键参数进行参数调试. 本节对比了 Gurobi、GA、VNS[14] 和 GA-VNS 对小规模和大规模算例的计算效果, 以验证所提算法性能. 本节所涉及的所有算法均在 Visual Studio 2013 平台使用 C#语言编程实现, 在 CPU (Intel 3.6 G)/RAM (8 GB) 的个人计算机上运行. 为了方便分析算法的性能, 本节将求解得到的目标函数转换成百分偏差, 用 R 作为参数分析的响应变量, 其值由式 (18) 决定.

$$R=\frac{obj(Alg)-obj(Best)}{obj(Best)}\times 100\% \tag{18}$$

式 (18) 中, Alg是计算某算法 R 值时的当前算法, $obj(Alg)$ 则是当前算法计算 10 次的平均目标值; $Best$ 是指在计算某一具体算例时本文所比较的包括 Gurobi 在内的 4 种算法中计算 10 次平均目标值最小的算法, $obj(Best)$ 则是最好算法计算 10 次的平均目标值. 分析式 (18) 可知, R 值越小, 说明当前算法的效果越好.

3.1 算例设计

本节采用两组算例来验证算法性能. 其中小规模算例的工件数$n\in\{4,6,8,10,12\}$和$m\in\{2,3,4\}$, 大规模算例的工件数$n\in\{20,40,60,80,100\}$和$m\in\{4,6,8\}$, 共计 30 个算例. 同一组算例运算 10 次, 取平均值为最终运算结果. 对于各个工件, 设定工件 j 基本加工时间a_j, 惩罚加工时间b_j, 恶化工期h_j, 工件 j 开工时刻晚于恶化工期选择使用保温工作模式时单位时间能耗e_0, 加工时单位时间能耗q_j, 交货期d_j. 此处需要说明的是, 本文中所有工件在保温时单位时间能耗e_0都是相等的, 在此设为 5. 由于所研究的问题没有广泛可用的基准实例集, 因此本文测试实例的数据参考了文献 [14] 中产生数据的方法. 工件 j 的基本加工时间 a_j 为 [1, 100] 间均匀分布的随机数; 惩罚加工时间b_j为[1,50]之间均匀分布的随机数; 恶化工期h_j为$(0,D]$之间均匀分布的随机数, 其中$D=\sum_{j=1}^{n\times w}a_j/m$, 其中$w=0.5$; 工件在加工时单位时间能耗$q_j$为 [1, 20] 间均匀分布的随机数; 工件交货期d_j等于$a_j+(0,20]$, 即工件j的交货期等于其基本加工时间a_j与 (0,20] 之间均匀分布的随机数之和. 本文在计算时目标函数的加权系数α和β均取 1, 也就是本节所有算法计算的目标值都是总拖期与总能耗之和.

需要注意的是, Gurobi、GA 和 VNS 实验数据也是来源于此. GA 的算法结构采用本文遗传算法部分的结构, VNS 的算法结构采用本文变邻域搜索部分的内容. 设定 Gurobi 运算时间为 3600 s, 如果在给定的时间内没有找到最优解, 则返回当前已发现的最好整数解. 关于 GA 和 VNS 算法的终止条件, 它们的最大循环次数T_p和当前解连续没有得到改善的迭代次数nip与 GA-VNS 的相同.

3.2 参数调试

算法求解的质量和算法参数的选择息息相关, 本文所提 GA-VNS 算法所涉及的主要参数为: 用作 GA-VNS 算法初始解的遗传算法的交叉概率P_c、变异概率P_m以及 GA-VNS 最大迭代次数T_p. 基于初步测试, 遗传算法阶段种群数量P_{num}和最好个体连续未改善迭代次数nip分别设为 50 和 80. 需重点调参的 3 个关键参数的取值范围如下:

P_c: 0.6、0.7、0.8、0.9、0.99.

P_m: 0.01、0.05、0.10、0.15、0.20.

T_p: 100、200、300、400、500.

在 GA-VNS 算法中, 使用工件数为 40 机器数为 6 的大规模算例对每个参数都进行 10 次求解运算, 并

对其求解结果实施单因素方差分析 (ANOVA). 为了方便分析不同取值水平下 GA-VNS 算法的效果, 图 5 至图 7 分别给出了 3 个参数不同取值的均值图和 95% 置信水平下的 Tukey 真实著性差异 (Honestly Significant Difference, HSD) 区间.

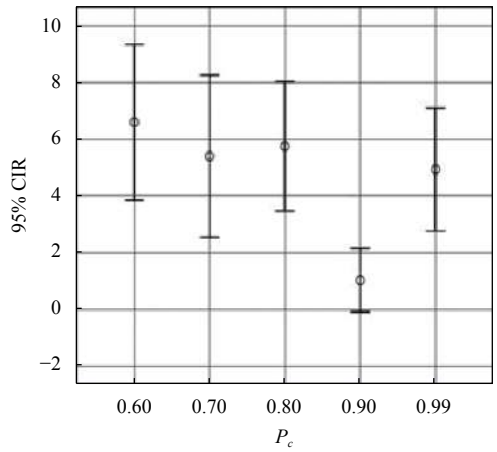

图 5 不同P_c取值均值图和 TukeyHSD 区间

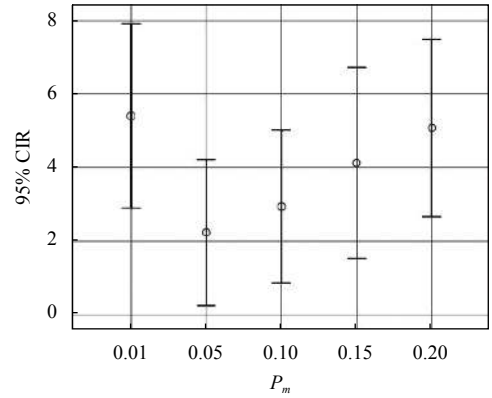

图 6 不同P_m取值均值图和 TukeyHSD 区间

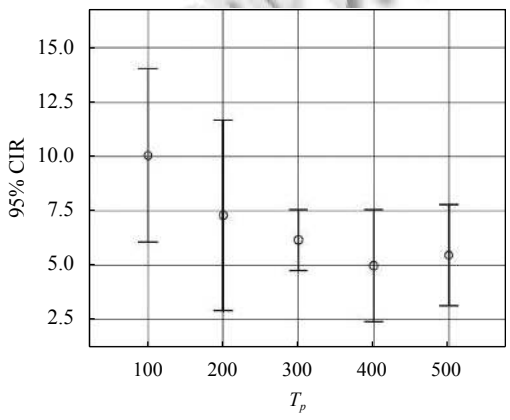

图 7 不同T_p取值的均值图和 TukeyHSD 区间

从图 5 可以看出, 交叉概率P_c太小, 会降低求解质量, 而当P_c值太大时, 求解质量也不是太理想. 明显得出当P_c取值为 0.9 时, R 值最小, 算法效果最好; 从图 6 可以看出变异概率P_m过小对算法效果的影响并不大, 当P_m的值为 0.05 时, R 有最小值, 且随着P_m越来越大, R 值也越来越大. 综合来看, P_m设为 0.05 最合适; 图 7 显示当参数T_p取值水平为 400、500 时, 并未产生明显差异, 且二者拥有相近的均值, 为保证更高的求解效率, 400 是更好的选择. 故基于上述分析, 交叉概率P_c=0.9, 变异概率P_m=0.05, 最大迭代次数T_p=400.

3.3 算例分析

在本节的计算中, 同一组算例运算 10 次, 取平均值的整数部分, 保留 R 值的小数后两位与平均运行时间的小数后 3 位为最终运算结果. 对比了 Gurobi、GA、GA-VNS 和 VNS 对小规模算例和大规模算例的运算结果.

表 2 列出了各个算法求解小规模算例的计算结果. 从表 2 中可以看出, GA-VNS 和 Gurobi 都能给出所有小规模算例的最好结果, 且其解效果十分稳定. 从计算时间上来看, GA-VNS 的平均计算时间为 0.232 s, 且只有工件数为 12, 机器数为 4 的算例计算时间超过 1 s, 其余均低于 1 s. 而 Gurobi 的平均计算时间为 1.166 s, 这说明在求解质量相同的情况下, GA-VNS 的求解效率要好于 Gurobi. GA 和 VNS 的计算时间比起 GA-VNS 来说相差无几, 但是两者的求解质量却不及 GA-VNS. 由表 2 可以看到, 在全部 15 个小规模算例中, VNS 只给出了 8 个算例的最优解, 平均 R 值为 0.804. 与 VNS 相比, GA 给出了 13 个算例的最优解, 平均 R 值为 0.107, 优于 VNS, 但还是不如 GA-VNS 给出的解的质量好. 从基于小规模算例的计算结果可以看出, GA-VNS 表现出了最好的求解性能.

使用 Gurobi、GA、VNS 和 GA-VNS 求解大规模算例, 计算结果见表 3. 由于问题的难求解性, Gurobi 仅能给出机器数为 6 工件数为 20、机器数为 8 工件数为 20 以及机器数为 8 工件数为 40 的近似最优解. 并且在求解所有大规模算例时, 均耗光了计算时间都未找到最优解, 且还有 12 个算例未找到可行解. 总体来讲, Gurobi 求解大规模算例的效果并不理想. GA-VNS 给出的平均 R 值为 0.007, 明显优于 Gurobi 和另外两个启发式算法. 除了机器数为 6, 工件数为 100 这一算例外, 其余算例的最优值 *Best* 均由 GA-VNS 给出.

在 3 种启发式算法中, GA-VNS 由于引入 GA 输入初始解和变邻域搜索结构, 使得其平均计算时间最长, 平均计算时间为 3.142 s. 但考虑到 GA-VNS 优良的求解性能, 付出的计算时间还是可以接受的. 相比之下, GA 的运算时间是最短的, 其平均计算时间为 0.689 s. 但从表 3 可明显看出, 虽然 GA 的求解时间是最短的, 但其求解效果在 3 个启发式算法中却是最差的, 平均 R 值达到了 38.233, 说明其求解精度还有很大的提高空间. VNS 的平均计算时间和 GA-VNS 相差不大, 为 2.994 s, 但在 15 个大规模算例中, VNS 仅给出了一个最好的 *Best* 值, 平均 R 值为 5.24, 且随着工件数的增加, 其 R 值也有逐渐增大的趋势. 总的来说, GA-VNS 在较短的时间内给出了不错的近似最优解, 能够有效解决同时考虑工时阶梯恶化与能耗的并行机调度问题.

表 2 小规模算例计算结果

m	*n*	*Best*	Gurobi		GA		VNS		GA-VNS	
			Time(s)	*R*	Time(s)	*R*	Time(s)	*R*	Time(s)	*R*
2	4	84	0.031	0.00	0.023	0.00	0.056	0.00	0.230	0.00
	6	261	0.418	0.00	0.103	0.00	0.046	0.00	0.098	0.00
	8	320	0.546	0.00	0.130	0.00	0.100	0.00	0.126	0.00
	10	561	1.031	0.00	0.445	0.00	0.212	2.10	0.377	0.00
	12	1043	10.392	0.00	0.416	0.00	0.326	0.00	0.503	0.00
3	4	82	0.031	0.00	0.024	0.00	0.030	0.00	0.030	0.00
	6	234	0.031	0.00	0.023	0.00	0.029	0.60	0.030	0.00
	8	243	0.850	0.00	0.114	0.00	0.089	1.15	0.125	0.00
	10	519	0.348	0.00	0.115	0.00	0.103	1.17	0.203	0.00
	12	869	2.346	0.00	0.877	1.26	0.856	1.81	1.365	0.00
4	4	82	0.002	0.00	0.012	0.00	0.009	0.00	0.010	0.00
	6	232	0.002	0.00	0.051	0.00	0.035	0.00	0.046	0.00
	8	206	0.010	0.00	0.053	0.00	0.056	4.95	0.050	0.00
	10	513	0.006	0.00	0.057	0.00	0.068	0.00	0.060	0.00
	12	818	1.441	0.00	0.215	0.35	0.120	0.28	0.268	0.00
平均值	—	—	1.166	0.00	0.177	0.107	0.142	0.804	0.232	0.00

表 3 大规模算例计算结果

m	*n*	*Best*	Gurobi		GA		VNS		GA-VNS	
			Time(s)	*R*	Time(s)	*R*	Time(s)	*R*	Time(s)	*R*
4	20	10826	3600	—	0.387	17.70	0.367	3.10	0.449	0
	40	29316	3600	—	0.493	31.54	0.820	8.66	1.000	0
	60	47407	3600	—	0.572	43.67	2.965	9.41	2.109	0
	80	72442	3600	—	0.665	55.11	4.269	7.37	4.563	0
	100	109485	3600	—	0.745	41.51	8.022	10.70	8.039	0
6	20	12519	3600	1.50	0.388	20.63	0.413	3.12	0.475	0
	40	26190	3600	—	0.450	26.56	0.913	4.74	1.082	0
	60	42323	3600	—	0.543	44.32	2.965	7.71	2.292	0
	80	73601	3600	—	0.605	42.74	4.285	4.20	4.742	0
	100	97935	3600	—	2.622	51.42	7.740	0	8.314	0.11
8	20	9031	3600	0.00	0.323	17.57	0.439	1.71	0.537	0
	40	40788	3600	2.67	0.493	16.81	0.940	2.97	1.149	0
	60	43245	3600	—	0.574	37.48	2.230	4.51	2.531	0
	80	69253	3600	—	0.680	39.04	4.514	5.38	4.916	0
	100	68357	3600	—	0.796	87.39	4.035	5.02	4.939	0
平均值	—	—	3600	1.39	0.689	38.233	2.994	5.24	3.142	0.007

4 结论与展望

本文针对尿素造粒塔生产过程, 提出了一类考虑工时恶化和能耗的并行机调度问题, 建立了优化目标为加权总拖期与总能耗的混合整数线性规划模型. 由于问题的难求解性以及工作模式选择的特殊性, 本文提出了基于遗传算法框架的改进的遗传变邻域混合搜索算法. 采用了单因素方差分析确定关键算法参数, 并基于随机产生的两组算例, 分析了算法的性能. 与数学

求解器 Gurobi、基本的遗传算法与基本的变邻域搜索算法的计算结果做了对比, 结果验证了本文所提算法的可行性与高效性. 后续研究可引入动态规划对两类模式进行选择, 考虑结合更切合尿素造粒塔的实际生产情况的真实数据优化问题, 同时设计多目标的求解算法, 把两个优化目标单独研究.

参考文献

1 吴国忠, 白浩然, 杨显志, 等. 尿素造粒塔内热环境仿真研究. 当代化工, 2014, 43(6): 975–977, 993. [doi: 10.3969/j.issn.1671-0460.2014.06.035]

2 付亚平, 黄敏, 王洪峰, 等. 混合并行机调度问题的多目标优化模型及算法. 控制理论与应用, 2014, 31(11): 1510–1516. [doi: 10.7641/CTA.2014.31233]

3 Li K, Yang SL. Non-identical parallel-machine scheduling research with minimizing total weighted completion times: Models, relaxations and algorithms. Applied Mathematical Modelling, 2009, 33(4): 2145–2158. [doi: 10.1016/j.apm.2008.05.019]

4 孟磊磊, 张超勇, 詹欣隆, 等. 不相关并行机节能调度问题建模. 中国机械工程, 2018, 29(23): 2850–2858. [doi: 10.3969/j.issn.1004-132X.2018.23.012]

5 周炳海, 顾佳颖. 考虑多资源约束的非等效并行机节能调度算法. 东北大学学报 (自然科学版), 2019, 40(3): 403–408. [doi: 10.12068/j.issn.1005-3026.2019.03.019]

6 雷德明, 潘子肖, 张清勇. 多目标低碳并行机调度研究. 华中科技大学学报 (自然科学版), 2018, 46(8): 104–109. [doi: 10.13245/j.hust.180820]

7 Li ZT, Yang HD, Zhang SQ, *et al*. Unrelated parallel machine scheduling problem with energy and tardiness cost. The International Journal of Advanced Manufacturing Technology, 2016, 84(1–4): 213–226. [doi: 10.1007/s00170-015-7657-2]

8 王永琦, 吴飞, 江潇潇, 等. 求解并行机拖期与能耗成本优化调度的混合教—学算法. 计算机应用研究, 2019, 36(3): 673–676. [doi: 10.19734/j.issn.1001-3695.2017.09.0929]

9 郭鹏, 程文明, 张则强. 求解具有恶化工件单机调度问题的改进遗传算法. 西南交通大学学报, 2011, 46(3): 506–511. [doi: 10.3969/j.issn.0258-2724.2011.03.025]

10 Ji M, Cheng TCE. Parallel-machine scheduling of simple linear deteriorating jobs. Theoretical Computer Science, 2009, 410(38–40): 3761–3768. [doi: 10.1016/j.tcs.2009.04.018]

11 Wang XY, Zhou ZL, Ji P, *et al*. Parallel machines scheduling with simple linear job deterioration and non-simultaneous machine available times. Computers & Industrial Engineering, 2014, 74: 88–91. [doi: 10.1016/j.cie.2014.05.003]

12 Guo P, Cheng WM, Wang Y. Parallel machine scheduling with step-deteriorating jobs and setup times by a hybrid discrete cuckoo search algorithm. Engineering Optimization, 2015, 47(11): 1564–1585. [doi: 10.1080/0305215x.2014.982634]

13 Guo P, Cheng WM, Zeng M, *et al*. Scheduling step-deteriorating jobs on parallel machines by mixed integer programming. Journal of Donghua University (English Edition), 2015, 32(5): 709–714, 719.

14 Cheng WM, Guo P, Zhang ZQ, *et al*. Variable neighborhood search for parallel machines scheduling problem with step deteriorating jobs. Mathematical Problems in Engineering, 2012, 2012: 928312. [doi: 10.1155/2012/928312]

15 Lalla-Ruiz E, VOß S. Modeling the parallel machine scheduling problem with step deteriorating jobs. European Journal of Operational Research, 2016, 255(1): 21–33. [doi: 10.1016/j.ejor.2016.04.010]

16 Pei J, Wang XM, Fan WJ, *et al*. Scheduling step-deteriorating jobs on bounded parallel-batching machines to maximise the total net revenue. Journal of the Operational Research Society, 2019, 70(10): 1830–1847. [doi: 10.1080/01605682.2018.1464428]

17 轩华, 秦莹莹, 王薛苑, 等. 带恶化工件的不相关并行机调度优化. 系统仿真学报, 2019, 31(5): 919–924. [doi: 10.16182/j.issn1004731x.joss.17-0150]

18 Yin YQ, Wang Y, Cheng TCE, *et al*. Parallel-machine scheduling of deteriorating jobs with potential machine disruptions. Omega, 2017, 69: 17–28. [doi: 10.1016/j.omega.2016.07.006]

19 Arık OA, Toksarı MD. Multi-objective fuzzy parallel machine scheduling problems under fuzzy job deterioration and learning effects. International Journal of Production Research, 2018, 56(7): 2488–2505. [doi: 10.1080/00207543.2017.1388932]

20 Lu SJ, Liu XB, Pei J, *et al*. A hybrid ABC-TS algorithm for the unrelated parallel-batching machines scheduling problem with deteriorating jobs and maintenance activity. Applied Soft Computing, 2018, 66: 168–182. [doi: 10.1016/j.asoc.2018.02.018]

21 郭鹏. 具有分段恶化效应生产过程的智能优化调度研究 [博士学位论文]. 成都: 西南交通大学, 2014.

22 Guo P, Cheng WM, Wang Y. Scheduling step-deteriorating jobs to minimise the total weighted tardiness on a single machine. International Journal of Systems Science: Operations & Logistics, 2017, 4(2): 92–107. [doi: 10.1080/23302674.2015.1084393]