个体简化的遗传规划求解动态多柔性作业车间调度问题

邱佳龙[1], 龚文引[1], 张国辉[2], 卢　超[1†]

(1. 中国地质大学(武汉) 计算机学院, 湖北 武汉 430074; 2. 郑州航空工业管理学院, 河南　郑州 450046)

摘要: 研究动态柔性作业车间调度问题(DFJSP)对有效控制生产过程, 提升企业经济盈利能力具有重要意义. 现有研究大多仅考虑机器选择柔性及动态生产场景, 但实际生产中还存在更复杂的工序顺序柔性问题. 在定制化装备制造等场景下, 作业加工工序可按工艺要求划分为若干组, 同组内工序无强制加工顺序约束, 不同组间则需遵循严格加工顺序. 这种工序顺序柔性的引入显著增加问题搜索空间复杂度, 导致求解时间大幅延长. 因此, 本文结合定制化装备制造车间的实际生产需求, 综合考虑机器选择柔性, 工序顺序柔性及新作业到达的动态特性, 提出动态多柔性作业车间调度问题(DMFJSP), 其优化目标为最小化平均流程时间. 为求解该问题, 本文提出一种基于个体简化的遗传规划(GP)方法, 通过引入动态终端节点频率分析个体结构复杂度, 并在适应度评估阶段设置惩罚函数引导种群进化, 进而简化个体结构, 缩短GP训练时间. 为验证所提方法的有效性, 本文在3种不同生产场景下开展测试, 结果表明, 该方法在保证求解质量的同时, 可显著缩短计算时间. 面对更大规模和复杂问题时, 其收敛速度快, 能在合理时间内找到满意解.

关键词: 动态柔性作业车间调度; 工序顺序柔性; 遗传规划

引用格式: 邱佳龙, 龚文引, 张国辉, 等. 个体简化的遗传规划求解动态多柔性作业车间调度问题. 控制理论与应用, 2025, 42(12): 2587 – 2597

DOI: 10.7641/CTA.2025.40653

Genetic programming with individual simplification policy for dynamic multi-flexible job shop scheduling problem

QIU Jia-long[1], GONG Wen-yin[1], ZHANG Guo-hui[2], LU Chao[1†]

(1. School of Computer Science, China University of Geosciences, Wuhan Hubei 430074, China;
2. Zhengzhou University of Aeronautics, Zhengzhou Henan 450046, China)

Abstract: Studying the dynamic flexible job-shop scheduling problem (DFJSP) is of great significance for effectively controlling production processes and enhancing enterprises' economic profitability. Existing studies mostly only consider machine selection flexibility and dynamic production scenarios; however, a more complex issue of flexible process sequence exists in practical production. Specifically, in scenarios like customized equipment manufacturing, job processing operations can be divided into several groups according to process requirements. There are no mandatory processing sequence constraints apply to operations within the same group, whereas strict processing sequence constraints must be followed between different groups. The introduction of this flexible process sequence significantly increases the complexity of the problem's search space, leading to a substantial extension of solution time. Therefore, by integrating the practical production needs of customized equipment manufacturing workshops, this paper comprehensively considers machine selection flexibility, flexible process sequence, and the dynamic characteristic of new job arrivals, and proposes the dynamic multi-flexible job-shop scheduling problem (DMFJSP) with the optimization objective of minimizing the average flow time. To solve this problem, a genetic programming (GP) method based on individual simplification is proposed which analyzes the structural complexity of individuals by introducing dynamic terminal node frequency, and sets a penalty function during the fitness evaluation stage to guide population evolution-ultimately simplifying individual structures and reducing GP training time. To verify the effectiveness of the proposed method, tests are conducted under three different production scenarios. The results demonstrate that the method can significantly shorten computation time while ensuring solution quality; when addressing larger-scale and more complex problems, it exhibits fast convergence and can obtain satisfactory solutions within a reasonable time frame.

Key words: dynamic flexible job shop scheduling problem; flexible process sequence; genetic programming

Citation: QIU Jialong, GONG Wenyin, ZHANG Guohui, et al. Genetic programming with individual simplification policy for dynamic multi-flexible job shop scheduling problem . *Control Theory & Applications*, 2025, 42(12): 2587 – 2597

收稿日期: 2024−12−31; 录用日期: 2025−11−03.

[†]通信作者. E-mail: luchao@cug.edu.cn.

本文责任编委: 李德伟.

国家自然科学基金项目(52175490, 51805495)资助.

Supported by the National Natural Science Foundation of China (52175490, 51805495).

1 引言

随着信息通信技术发展, 制造企业融合云制造[1]、智能制造[2]及大数据[3]推进数字化转型. 柔性作业车间调度问题 (flexible job-shop scheduling problem, FJSP)作为作业车间调度问题(JSP)的扩展, 在车间调度领域应用广泛[4]. 但当前研究多聚焦静态调度, 并未考虑实际车间扰动, 难以满足生产需求. 进而动态柔性作业车间调度问题(dynamic FJSP, DFJSP)成为研究热点[5]. 新作业到达是实际生产中常见的动态事件[6], 初始调度作业并非全部可用, 新作业随时间陆续到达车间加工. 这种动态事件会显著增加调度不确定性与复杂性. 此外, 现代制造企业为适配动态生产环境, 依订单制定灵活工艺流程, 在满足工艺约束的情况下, 一些产品的加工工序具备工序顺序柔性, 如定制化特种装备装配车间中, 盾体、电气等部件装配可并行, 但需在总装前完成. 此类调度问题更贴合实际生产, 而DFJSP尚未考虑工序顺序柔性, 因此本文综合机器选择柔性与工序顺序柔性, 研究动态多柔性作业车间调度问题(dynamic multi-flexible JSP, DMFJSP).

针对新作业到达的DFJSP已有相关研究. Baykasoglu等[7]提出一种贪心随机自适应搜索策略, 求解依赖序列相关准备时间的DFJSP; Lei等[8]采用图神经网络与多层感知器代理的层次强化学习框架, 处理大规模DFJSP; Zhao等[9]结合注意力机制与策略强化学习算法求解DFJSP问题; Liu等[10]基于分布式分层双深度Q神经网络, 通过奖励驱动学习车间信息与调度决策的映射以求解DFJSP.

上述针对新作业到达DFJSP的研究, 均仅考虑工序的机器选择柔性, 当前已有研究关注工序顺序柔性. 如Vital-Soto等[11]提出改进非支配排序遗传算法, 求解考虑工序顺序柔性与工人分配的双资源约束FJSP; Zhang等[12]提出分布式蚁群算法, 解决含工序顺序柔性的柔性装配作业车间调度问题; 胡瑞淇等[13]设计基于表达式树的工序顺序染色体编码, 结合遗传算法求解工序顺序柔性作业车间调度问题; 魏光艳等[14]采用结合分布估计算法与邻域搜索算子的多维模因算法, 求解分布式多柔性装配作业车间调度问题. 但上述研究均针对静态车间场景, 未考虑新作业到达这类扰动因素.

现有研究尚未在动态作业车间中综合考虑机器选择柔性与工序顺序柔性, 而该问题在制造企业生产中十分常见. 本文调度问题源自高端装备制造企业装配车间, 车间内多工位可加工不同工序, 作业每道工序可选多个工位. 产品装配分为部装与总装阶段, 部装阶段无顺序约束可并行, 需全部完工后方能进入总装. 总装通常含1–2道工序(如调试、厂验), 需按序加工. 且产品装配作业随时间动态到达, 新作业到达前不可预测. 该场景类似动态多柔性作业车间调度问题. 因企业面临高度定制化, 复杂化生产任务, 订单需求与优先级各异. 为满足其平衡工作负荷与成本控制的需求, 本文以最小化平均流程时间目标, 构建DMFJSP问题.

现有DFJSP求解方法主要分为启发式规则、元启发式方法与基于机器学习的方法. 启发式规则基于直观或经验逻辑制定, 易理解实现. Xiong等[15]针对考虑批量释放与扩展技术优先约束的动态车间调度问题, 基于松弛时间与动态比率提出4种调度规则, 优化延迟相关指标; Oukil等[16]提出基于数据包络分析的多视角方法, 解决多目标动态流水车间调度规则排名问题. 但启发式规则设计需丰富领域知识与大量试错成本, 且缺乏全局优化能力, 解质量难以保证[17]. 元启发式方法可通过参数调整适配不同DFJSP, 全局搜索能力强, 还能结合多种策略提升搜索效率与解质量. 王艳等[18]提出改进多目标差分进化算法, 求解以最小化能耗、完工时间与调度稳定性为目标的DFJSP; 张国辉等[19]结合事件驱动策略设计帝国竞争算法, 求解考虑机器故障的DFJSP. 但元启发式方法迭代与计算量大, 难以满足DFJSP的实时性需求[20]. 深度强化学习方法作为机器学习的一类, 能随环境变化自动调整策略, 通过试错反馈优化调度效果[21]. Shi等[22]提出基于深度强化学习的智能调度方法, 应用于离散自动化生产线; Song等[23]设计异构图神经网络, 将操作选择与机器分配整合为复合决策求解FJSP. 但深度强化学习需手动设计适配动态问题的重调度规则, 依赖复杂先验知识. 此外, 众多学者采用遗传规划(genetic programming, GP)求解DFJSP[24], 相较于启发式规则与深度强化学习, GP可模仿自然选择和遗传机制, 自动从候选规则中优化最佳规则, 更适配复杂动态环境. 相较于元启发式算法, GP生成的演化规则计算简单, 能快速响应动态事件. 因此, 本文选用GP作为算法框架求解DMFJSP.

GP求解DFJSP的研究较多. Zhang等[25]提出协同进化遗传规划(cooperative co-evolution genetic programming with two sub-populations, CCGP), 可同时优化机器选择与工序排序规则; Zhou等[26]采用双子树遗传规划(genetic programming with single population that a GP individuals contains two sub-trees, TTGP)方法, 将两类规则编码于单个GP个体求解; Xu等[27]设计带存档的多子树遗传规划方法(CCGP with archive, CCAGP)解决DFJSP, 通过存档存储进化潜力解以供后续迭代复用; Zhu等[28]提出改进交叉算子的CCGP方法(CCGP selects both two rules for crossover, and one rule and for swap, CCGP-C^{2}S), 在机器选择与工序排序规则上应用交叉, 交换策略提升种群多样性. 上述研究多聚焦改善GP种群, 却鲜少关注GP个体结构对算法性能的影响.

现有研究尚未关注DMFJSP, 本文针对该问题提出个体简化的GP方法, 该方法充分考虑机器选择、工序

排序与工序顺序柔性3个子问题, 通过适应度评估阶段的个体简化策略引导种群进化, 在缩短算法计算时间的同时保证求解质量. 本文主要贡献如下:

1) 研究一种动态多柔性作业车间调度问题, 该问题考虑新作业到达、机器选择柔性与工序顺序柔性3类现实约束, 以平均流程时间为优化目标, 建立该问题的数学模型;

2) 提出个体简化的GP方法求解该问题, 通过评估动态终端节点频率引导种群进化方向, 克服GP训练耗时久、收敛速度慢的缺点.

2　问题描述与建模

2.1　问题描述

DMFJSP除机器选择柔性外, 还考虑工序顺序柔性, 作业加工工序按工艺划分为若干组, 同组工序无顺序约束, 不同组需遵循严格加工顺序. 该问题可描述为: 生产车间内含m台机器$M=\{M_1,M_2,\cdots,M_m\}$, 初始有n个作业$J=\{J_1,J_2,\cdots,J_n\}$待加工, 后续新作业$J'=\{J'_1,J'_2,\cdots,J'_n\}$陆续到达, 作业$J_i$含$N_i$道加工工序$O=\{O_{i,1},O_{i,2},\cdots,O_{i,N_i}\}$, 其优先级层级为$R_i$, 优先级为$r$的工序共$K_{i,r}$道, 同优先级工序无顺序约束, 低优先级工序需待高优先级工序完工后进行, 且每个作业有对应到达时间, 优化目标为最小化平均流程时间, DMFJSP包含以下子问题: 1) 作业的机器选择问题; 2) 工序的排序问题; 3) 工序间的优先级约束问题.

此外, 为更好地表述问题, 作如下假设:

1) 所有作业的每道工序仅分配处理一次;

2) 同一作业工序受优先级约束, 不同作业工序间无优先级约束;

3) 作业工序加工过程不可中断;

4) 作业工序仅分配至适配机器加工;

5) 每台机器0时刻均可用;

6) 每台机器可加工多道工序, 但同一时刻至多加工一道.

2.2　符号说明

为了更好的描述DMFJSP的数学模型, 使用的符号和含义如表1所示.

2.3　数学模型

目标函数

$$\min \mathrm{Fmean}=\frac{1}{n}\sum_{i=1}^{n}(C_i-l_i), \tag{1}$$

其中式(1)表示最小化平均流程时间.

约束条件:

$$\sum_{k=1}^{m}x_{i,j,r,k}=1,\ \forall i,j,r, \tag{2}$$

$$\mathrm{ST}_{i,j,R_i}\geqslant l_i,\ \forall i,j, \tag{3}$$

$$\mathrm{CT}_{i,1,R_i}-P_{i,1,R_i}\geqslant l_i,\ \forall i, \tag{4}$$

$$\mathrm{ST}_{i,j,r}\geqslant\max(\mathrm{CT}_{i,j,r+1}),\ \forall i,j,r\in[1,R_i-1], \tag{5}$$

$$\mathrm{ST}_{i,j,r}\leqslant\mathrm{ST}_{i,j',r}+P_{i,j',r,k'},\ \forall i,r,j\neq j', \tag{6}$$

$$\mathrm{ST}_{i,j',r}\leqslant\mathrm{ST}_{i,j,r}+P_{i,j,r,k},\ \forall i,r,j\neq j', \tag{7}$$

$$x_{i,j,r,k}\leqslant W_{i,j,r,k},\ \forall i,j,r,k, \tag{8}$$

$$\mathrm{CT}_{i,j,r}=\mathrm{ST}_{i,j,r}+P_{i,j,r,k},\ \forall i,j,r,k, \tag{9}$$

$$(\mathrm{ST}_{i,j,r}+P_{i,j,r,k})\,y_{i,j,r,i',j',r',k}\leqslant\mathrm{ST}_{i',j',r'}, \tag{10}$$

$$\mathrm{ST}_{i,j,r}\geqslant 0,\ \mathrm{CT}_{i,j,r}\geqslant 0,\ P_{i,j,r,k}\geqslant 0, \tag{11}$$

其中: 式(2)表示所有作业的每道工序仅分配处理一次; 式(3)和式(4)表示作业最高优先级工序需在作业释放后进行; 式(5)表示同一作业中, 低优先级组任意工序需待高优先级组所有工序完工后进行; 式(6)与式(7)表示同一优先级的任意两道工序可并行加工; 式(8)表示每道工序均分配至适配机器; 式(9)表示工序加工过程不可中断; 式(10)表示每台机器同一时刻仅能处理一道工序; 式(11)表示工序的开始时间, 处理时间及完成时间均不小于0.

表 1　符号说明

Table 1　Symbol notation

	符号	含义
索引	i	作业索引, $i=1,2,\cdots,n$
	j	工序索引, $j=1,2,\cdots,N_i$
	k	机器索引, $k=1,2,\cdots,m$
	r	优先级索引, $r=1,2,\cdots,R_i$
集合	J	作业集合, $J=\{J_1,J_2,\cdots,J_n\}$
	M	机器集合, $M=\{M_1,M_2,\cdots,M_m\}$
	O_i	作业J_i的工序集合, $O_i=\{O_{i,1,1},O_{i,2,2},\cdots,O_{i,N_i,R_i}\}$
参数	n	作业数量
	m	机器数量
	N_i	作业J_i的工序数量
	R_i	作业J_i的优先级层数
	$K_{i,r}$	作业J_i中优先级为r的工序数量
	l_i	作业J_i的释放时间
	C_i	作业J_i的完成时间
	$\mathrm{ST}_{i,j,r}$	作业J_i中优先级为r的第j道工序$O_{i,j,r}$的开始时间
	$\mathrm{CT}_{i,j,r}$	作业J_i中优先级为r的第j道工序$O_{i,j,r}$的完成时间
	$P_{i,j,r,k}$	作业J_i中优先级为r的第j道工序$O_{i,j,r}$在机器M_k上的运行时间
	$W_{i,j,r,k}$	如果工序$O_{i,j,r}$能够在机器M_k上加工, 则取值为1, 否则取值为0
决策变量	$x_{i,j,r,k}$	如果工序$O_{i,j,r}$在机器M_k上加工, 则取值为1, 否则取值为0
	$y_{i,j,r,i',j',r',k}$	如果工序$O_{i,j,r}$在工序$O_{i',j',r'}$之前在机器M_k上加工, 则取值为1, 否则取值为0

3 算法描述

3.1 个体设计

GP个体为树状结构, 含节点与子树. 节点分为操作符节点与终端节点: 操作符节点含算数运算符, 如Add, Sub和Max; 终端节点用于表示车间生产状态, 如PT, OWT和NIQ. 因本文需考虑工序优先级约束, 因此在终端节点中增加OP, OPR和NPIQ等与工序约束相关节点, 以促使调度决策更多考虑优先级约束, 间接影响决策演化结果. 表2列出了构建GP树所用终端节点与操作符节点功能.

表 2 GP个体节点及对应功能

Table 2 GP individual nodes and their corresponding functions

属性	符号	描述
终端节点	PT	工序的处理时间
	DD	到期日
	OWT	工序的等待时间
	NOR	作业中剩余的操作数
	TIS	系统时间
	NIQ	当前机器队列的工序数
	WIQ	当前工序所在机器的总处理时间
	OP	当前工序的优先级
	OPR	作业中具有相同优先级的剩余工序数
	NPIQ	作业在当前机器队列中具有相同优先级的剩余工序数
操作符节点	Add	相加
	Sub	相减
	Max	取大值
	Min	取小值
	Mul	相乘
	Div	相除

根据种群个体初始化方式不同, GP可划分为CCGP与TTGP. TTGP中, 单个种群个体为一棵树, 左、右子树分别对应机器选择与工序排序决策, 两者在同一GP个体中协同进化, 如图1(a)所示. CCGP则将种群个体拆分为两棵独立树, 分别对应机器选择与工序排序决策, 两类规则独立进化, 如图1(b)所示. 采用这两种方式创建新种群, 可充分衡量不同种群个体类型对算法性能的影响.

3.2 个体适应度评估

运行时间是评价算法性能的重要指标. GP运行时, 每次迭代需对种群所有个体进行适应度评估, 作为超启发式算法, GP个体并非问题的解, 而是求解问题的启发式规则, 故其适应度评估需模拟数千个陆续到达的作业. 个体解码的调度策略参与机器选择与工序排序决策: 新作业到达或当前工序完成时, 通过机器选择树计算候选机器排名以确定加工机器; 目标机器空闲时, 通过工序排序树计算等待工序排名以确定下一道处理工序. 该过程中所有树节点需执行数十万次运算, 导致计算耗时显著.

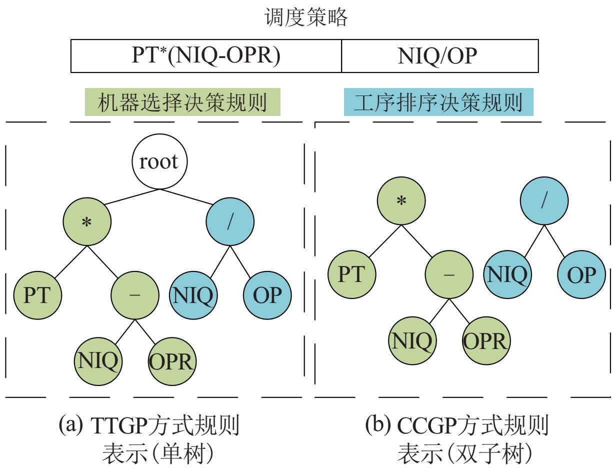


图 1 调度策略的编码表示

Fig. 1 Encoding representation of scheduling rules

为减少GP个体适应度评估耗时, 本文提出个体简化策略(individual simplification policy, ISP). ISP从两方面优化计算效率: 1) GP树节点含操作符节点与终端节点, 操作符节点为加减乘除等浮点数运算, 计算速度快, 终端节点按参数是否随程序变化分为静态与动态终端节点. 静态终端节点参数预定义, 如工序处理时间PT、到期日DD等, 耗时较短. 动态终端节点需实时计算车间状态, 耗时较长. 如工序等待时间OWT、同优先级剩余工序数OPR等; 2) 相同复杂度的GP个体, 动态终端节点越多, 高耗时动态节点占比越高, 运行时间越长, 故引入动态终端节点计算耗时占比, 量化个体的相对计算开销. 具体而言, ISP通过惩罚函数控制动态终端节点占比, 将其融入适应度计算. 惩罚函数设计需考虑了以下3个关键因素: 1) 动态终端节点占比: 计算动态节点数与总节点数比例; 2) 计算耗时权重: 为不同动态节点分配权重以反映耗时差异; 3) 惩罚力度: 结合前两者设置, 确保高动态节点在子代选择中处于劣势. ISP的优势在于从基因层面优化, 一方面, 动态终端节点中部分冗余节点的计算信息对GP个体核心决策无额外增益, 用非冗余节点替代后, 既能保留关键决策信息, 又能保障种群进化中解质量稳定; 另一方面, 通过量化其耗时占比, 搭配惩罚函数控制动态节点数量, 显著降低适应度评估耗时. 改进后的适应度计算公式如下所示:

$$\text{Fitness}^* = \text{Fitness} + \omega_{\text{F}}\Big(\sum_{i=1}^{n} b_i \times \frac{\text{num}_i}{\text{num}_{\text{dynamic}}}\Big)\text{Fitness}, \quad (12)$$

其中: ω_{F}为惩罚函数在原适应度中占比, b_i为第i种动态终端节点运行时间占所有动态节点运行时间的比例, $\text{num}_{\text{dynamic}}$为个体树中所有动态终端节点总数, num_i为个体树中第i种动态终端节点个数.

3.3　种群进化

种群个体经适应度评估后, 更新精英个体. 父代个体通过锦标赛选择从当前种群中筛选产生(参考文献[29], 比赛规模设置为3), 再通过交叉, 变异算子繁殖子代. 交叉算子采用单点交叉, 在两父代个体中各随机选取一个节点, 交换对应子树形成子代. 变异算子随机选取父代个体的一个节点作为突变点, 用随机生成的子树替换该节点子树. TTGP个体进化方式有特殊性, 其两种调度规则编码于同一棵树, 故交叉、变异算子需选择相同类型的子树节点. 交叉时从两父代中随机选取同类型子树节点, 交换子树生成子代; 变异时随机选择父代子树节点, 用随机子树替换该节点为根的子树.

图2给出 TTGP 交叉与变异算子的示例. 交叉时, 随机选取两个个体tree1(机器选择规则为PT*(NIQ-OPR))和tree2(机器选择规则为NOR+DD), 两者从同类型 GP 树中选取子树Subtree1与Subtree2, 交换后子代机器选择规则分别为PT*(NOR+DD), NIQ-OPR. 变异时, 随机选取个体tree1(机器选择规则为 PT*(NIQ-OPR)), 用随机生成的新子树替换其随机选定的子树Subtree1, 子代机器选择规则变为 PT*(OWT+TIS). 种群进化完成后, 将保留的精英个体混入新生子代, 再经锦标赛法筛选出与原父代种群数量一致的个体, 形成新种群.

3.4　算法流程

根据上述算法描述, 整个算法的流程图如图3所示.

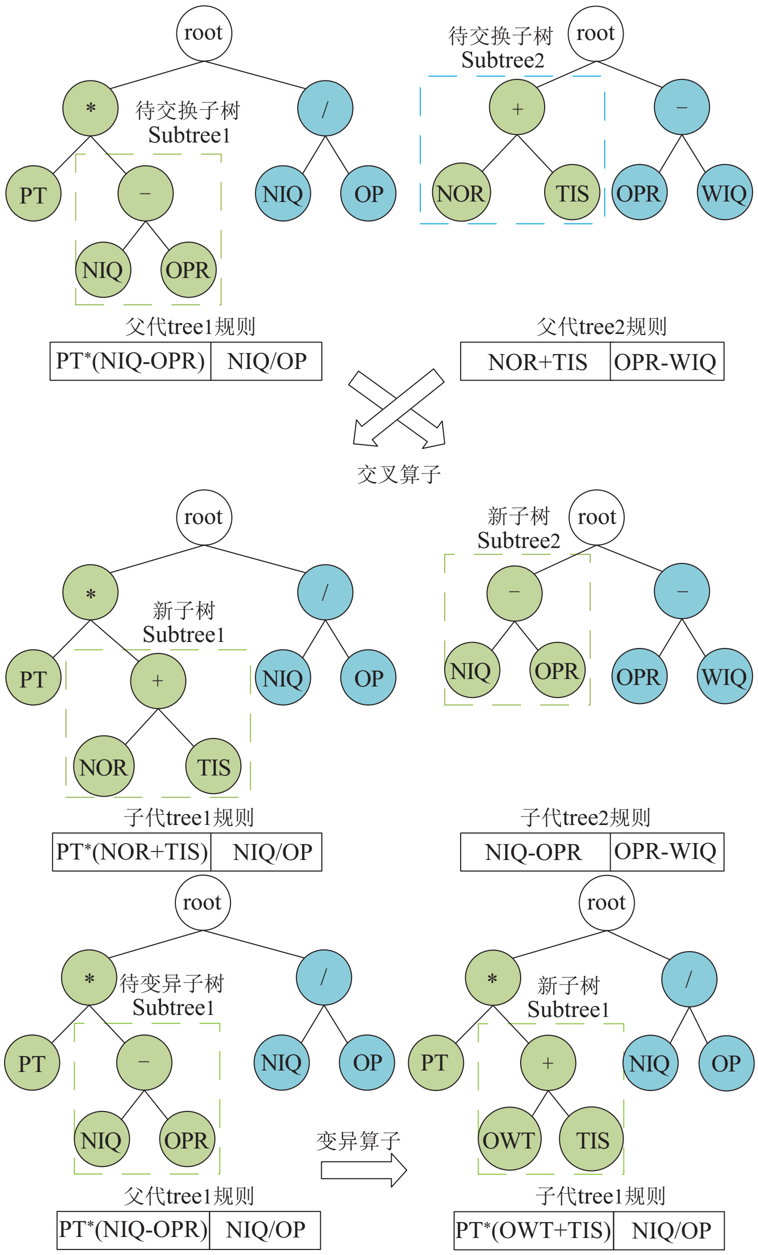


图 2　TTGP交叉和变异方式

Fig. 2　Crossover and mutation techniques in TTGP

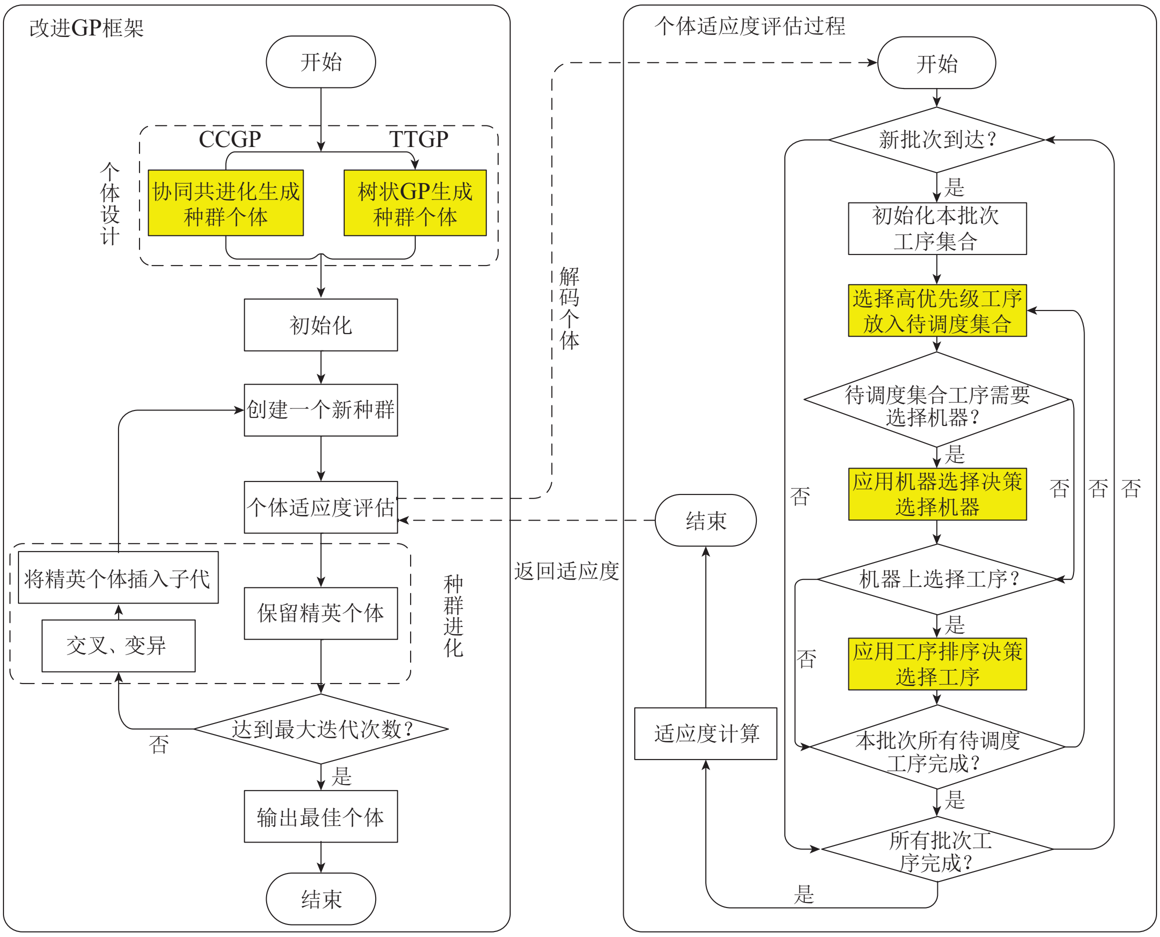


图 3 改进后的GP框架和个体适应度评估过程

Fig. 3 Improved GP framework and individual fitness evaluation process

算法的具体步骤如下:

步骤 1 初始化GP种群. 设置个体最大深度为6, 采用随机初始化方式分别生成CCGP和TTGP种群个体;

步骤 2 个体适应度评估(包含以下子步骤):

2.1 判断是否有新批次到达, 若有新批次到达, 初始化本批次工序集合, 否则跳转至步骤2.6;

2.2 选择高优先级工序放入待调度集合;

2.3 对于所有待调度集合工序, 判断是否需要选择机器: 若需要, 则应用机器选择决策计算候选机器排名并确定加工机器, 否则直接进入下一步;

2.4 判断当前机器是否需要选择工序: 若需要, 则应用工序排序决策计算等待工序排名并确定加工工序, 否则直接进入下一步;

2.5 判断本批次所有待调度工序是否完成: 若已完成, 则进入下一步, 否则跳转至步骤2.2;

2.6 判断所有批次工序是否完成: 若已完成, 则根据个体在该场景表现计算适应度, 否则跳转至步骤2.1;

步骤 3 更新精英个体: 判断种群中是否有个体适应度优于精英存档中个体, 若有则更新精英存档;

步骤 4 判断是否达到最大迭代次数, 若未达到, 通过锦标赛选择从当前种群筛选父代, 执行交叉与变异生成子代种群, 并将精英个体插入子代, 返回步骤2重新进行适应度评估;

步骤 5 若达到最大迭代次数, 输出适应度最优的个体, 该个体的调度规则作为最终用于动态场景的决策规则.

4 实验及分析

4.1 实验设置

本实验中, 仿真模型参数在一定范围内随机生成. 具体范围如表3所示. 作业到达时间依据机器利用率确定. 假设新作业到达车间遵循泊松过程, 平均到达时间间隔计算如下:

$$t = \frac{\bar{p} \times n_{\mathrm{p}}}{u \times m}. \tag{13}$$

其中: t为作业平均到达时间间隔, $\bar{p}$为作业工序平均处理时间, n_{p}为作业平均工序数量, u为机器利用率, m为机器数量. 作业到达时间间隔由指数分布$E(1/t)$计算. 交货期设为作业加工时间与交货期宽裕度系数的乘积.

为确保测量起始即获稳态性能, 将初始1000个作

业设为预热任务, 性能评估从第1001个作业开始, 至第6000个作业完成, 其统计信息用于适应度计算. 此外, 作业工序优先级层级(operation priority size, Ops)设为不同等级, Ops越大, 加工难度越高, 作业工艺流程越复杂. GP算法种群进化的相关参数设置如表4所示. 其中, 种群大小与精英个体数量经多次测试确定为适配值, 种群初始化采用完全生成法与成长生成法结合的Ramped half-and-half方式[30], 50%个体为最大深度, 其余个体深度服从$U[2,6]$, 确保种群中兼具简单与复杂树结构. 其余参数设置参考文献[31].

表 3 仿真模型参数设置
Table 3 Setting parameters in simulation models

参数	设置
预热作业数量	1000
稳态作业数量	5000
机器数量	10
工序数量	$U[2,9]$
作业的加工时间及其分布	$U[1,99]$
机器利用率	0.85
工序可选机器数量	$U[1,10]$
交货期宽裕度系数	1.5

表 4 GP算法参数设置
Table 4 Setting parameters in the GP algorithm

GP算法参数	设置
种群大小	512
精英个体数量	50
迭代次数	50
初始化方式	Ramped half-and-ha1f
个体最大深度	6
选择方式	锦标赛选择
交叉概率	85%
变异概率	15%

实验分训练阶段与测试阶段, 训练阶段每次迭代通过评估相同动态场景进化调度规则, 测试阶段使用表现最优的调度规则实时响应动态事件. 新作业到达时, 该最优调度规则会根据车间信息进行决策, 从而保证重调度的实时性. 最终以30次测试的平均适应度值作为本次训练结果, 独立运行30次, 取测试阶段平均适应度值评估算法性能. 需注意, ISP修正适应度值仅用于种群进化, 评估算法性能仍用原适应度值, 以保证对比实验适应度值范围一致. 此外, ISP需计算各动态终端节点占总运行时间的比例, 表5列出各动态终端节点10次独立的时间(单位: ms)均值及占比. Ops增加会显著提升问题求解难度. 为探究ISP对GP算法的影响, 本文按Ops=3, 4, 5设置3个生产场景, 将CC-GP, TTGP分别结合ISP形成CCGP-ISP, TTGP-ISP, 验证ISP在不同GP个体构建场景下的表现.

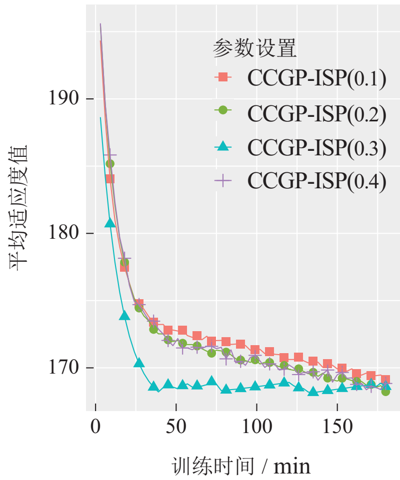
4.2 参数校验

ISP中惩罚函数占比ω_{F}控制种群个体简化程度. 占比过小则无法充分引导种群进化, 易出现更多高耗时动态终端节点; 占比过大则会过度简化个体, 导致其失去复杂性与表达能力. 本节探究不同ω_{F}对CCGP-ISP, TTGP-ISP性能的影响. 具体设置$\omega_{\mathrm{F}}=0.1, 0.2, 0.3, 0.4$共4种配置, 在3种生产场景中实验, 每次迭代取最优个体, 在同一场景下测试30次, 以平均适应度值表征该代种群性能.

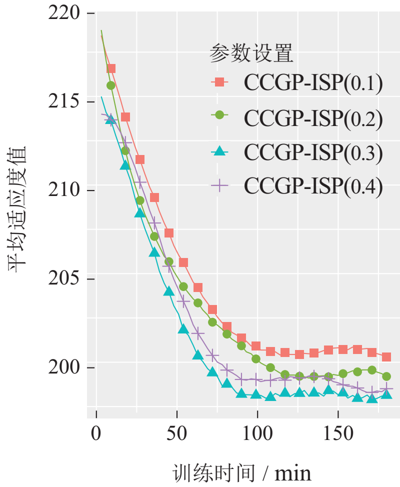
表 5 动态节点运行时间平均值及占比统计
Table 5 Statistics of average running time and proportion for dynamic nodes

动态节点	NIQ	WIQ	NOR	OWT	TIS	OPR	NPIQ
平均值	15.60	11.54	12.18	16.87	22.12	256.90	267.10
占比/%	2.6	1.9	2.0	2.8	3.7	42.7	44.3

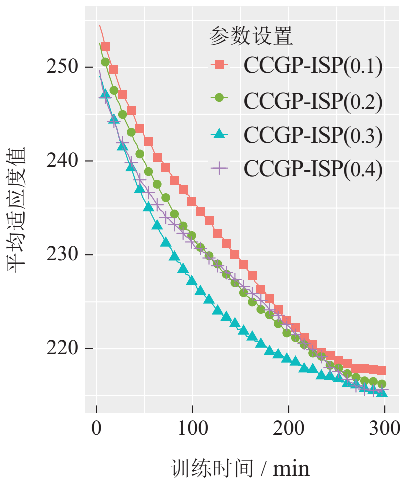
不同ω_{F}对CCGP-ISP的影响如图4所示(横轴为时间, 纵轴为平均适应度值). 3种场景中均呈现ω_{F}增大时收敛后平均适应度值先减再增的趋势, 其中$\omega_{\mathrm{F}}=0.3$时不仅最终适应度值最优, 且算法收敛更快, 如图5(TTGP-ISP平均适应度曲线)所示, $\omega_{\mathrm{F}}=0.3$在所有场景中同样表现最优, 故CCGP-ISP与TTGP-ISP均选取$\omega_{\mathrm{F}}=0.3$作为算法配置.



(a) CCGP-ISP(Ops=3)不同惩罚函数占比的适应度曲线



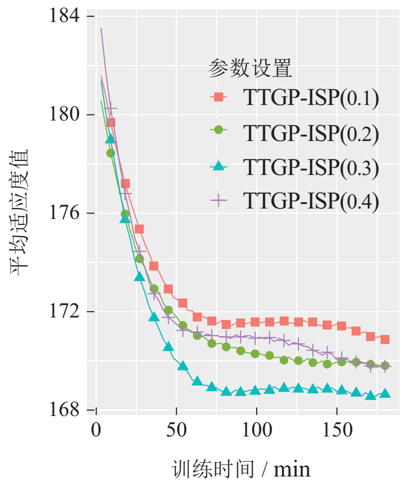
(b) CCGP-ISP(Ops=4)不同惩罚函数占比的适应度曲线



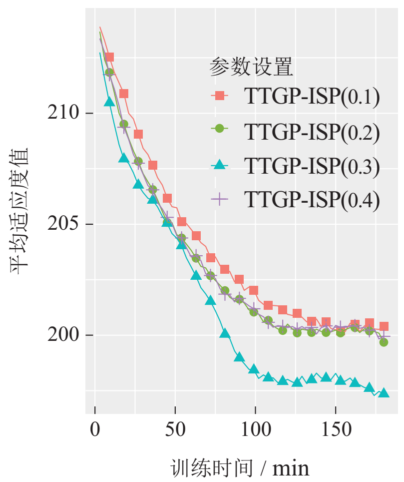
(c) CCGP-ISP(Ops=5)不同惩罚函数占比的适应度曲线

图 4　CCGP不同惩罚函数占比配置在3种生产场景中相同训练时间下独立运行30次得到的平均适应度曲线

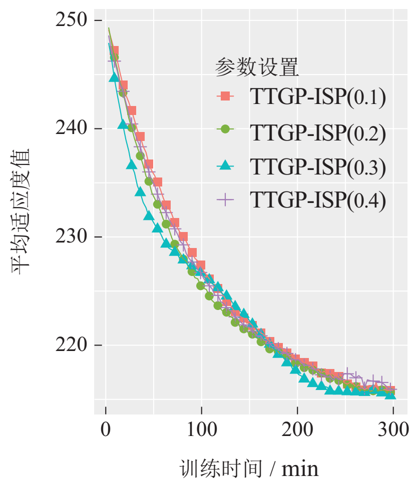
Fig. 4　Average fitness curves for CCGP using different proportions of penalty functions in three production scenarios, derived from 30 independent runs with identical training duration



(a) TTGP-ISP(Ops=3)不同惩罚函数占比的适应度曲线



(b) TTGP-ISP(Ops=4)不同惩罚函数占比的适应度曲线



(c) TTGP-ISP(Ops=5)不同惩罚函数占比的适应度曲线

图 5　TTGP不同惩罚函数占比配置在3种生产场景中相同训练时间下独立运行30次得到的平均适应度曲线

Fig. 5　Average fitness curves for TTGP using different proportions of penalty functions in three production scenarios, derived from 30 independent runs with identical training duration

4.3　实验结果及分析

为测试CCGP-ISP, TTGP-ISP性能, 本文将其与现有改进GP算法CCAGP[27], CCGP-C^{2}S[28]对比, 以验证ISP策略有效性. 采用显著性水平为0.05的Wilcoxon秩和检验, 基于3个生产场景下30次独立运行结果检验算法性能, 以“−”, “+” 和 “≈” 分别表示结果优于、差于和相近于对比算法.

表6给出不同算法在3种生产场景下30次独立运行的训练时间均值, 标准差及Wilcoxon秩和检验置信区间. 结果显示, CCGP-ISP在3个场景中训练时间均值均最优, TTGP-ISP在Ops=4, 5的场景中优于原始算法. 随问题规模从Ops=3扩大至Ops=5, ISP算法的时间优势更显著. 结果表明, 使用ISP可有效缩短GP训练时间. 此外, 工序优先级层级对训练时间影响显著, 层级越高, 工序越多, 工艺越复杂, 所需训练时间越长, 这一点在所有场景中均成立. 而简单场景下ISP算法与原始算法表现相近, 可能因简单场景中GP个体结构不复杂, 两类种群个体差异较小.

图6为不同算法在3种场景下30次独立运行的最优适应度小提琴图. Ops=3的场景中, CCAGP和CCGP-C^{2}S最优适应度值集中于较低区间, ISP算法与原始算法分布区间基本重合; Ops=4场景中, TTGP-ISP分布区间更低且中位数显著低于其他算法; Ops=5场景中, ISP算法与其余算法适应度分布基本持平. 结果表明, ISP通过减少动态终端节点计算冗余, 简化个体结构的同时保留机器选择与工序排序的核心决策信息, 不影响GP全局搜索能力. 且复杂场景中ISP可减少冗余节点对种群进化的干扰, 使算法聚焦于工序优先级约

束等关键因素, 进而获得更优解. 表7给出不同算法在3种场景下30次独立运行的最优适应度均值, 标准差及 Wilcoxon 秩和检验置信区间. 结果显示, 3 种场景中 ISP 算法与同类种群算法性能无统计学差异, 仅 Ops=4场景中ISP算法存在一定优势; 其余场景下各算法性能整体处于相近水平. 综上, ISP能在减少训练时间的同时, 实现与原始算法相似甚至更好的性能. 理论上, ISP仅降低个体计算复杂度, 不影响种群质量. 可能原因在于, 计算复杂动态终端节点在性能评估中的作用不优于静态节点, 减少其频率反而利于优化.

表 6　不同算法在3种生产场景中独立运行30次得到的训练时间平均值、标准差以及Wilcoxon秩和检验置信区间

Table 6　The mean of training time (in minutes), standard deviation, and confidence intervals from Wilcoxon rank - sum test for different algorithms over 30 independent runs in three production scenarios

生产场景	CCGP	CCGP-C^{2}S	CCAGP	CCGP-ISP	TTGP	TTGP-ISP
Fmean, Ops = 3	34(10)	46(10) (+)(5.78, 17.35)	32(4) (≈)(−3.25, 4.22)	28(7) (−)(−9.22, −1.43) (−)(−22.60, −14.47) (−)(−7.45, −1.63)	36(11)	34(10) (≈)(−0.08, 2.28)
Fmean, Ops = 4	168(23)	118(11) (−)(−80.62, −24.98)	113(17) (−)(−69.02, −18.62)	103(23) (−)(−74.05, −9.47) (−)(−23.08, −4.58) (−)(−21.13, −1.98)	140(13)	104(23) (−)(−142.81, −30.75)
Fmean, Ops = 5	291(17)	323(21) (+)(23.30, 40.01)	347(26) (+)(40.43, 60.33)	265(8) (−)(−28.88, −21.53) (−)(−66.07, −40.43) (−)(−87.63, −64.35)	305(11)	252(12) (−)(−59.51, −48.32)

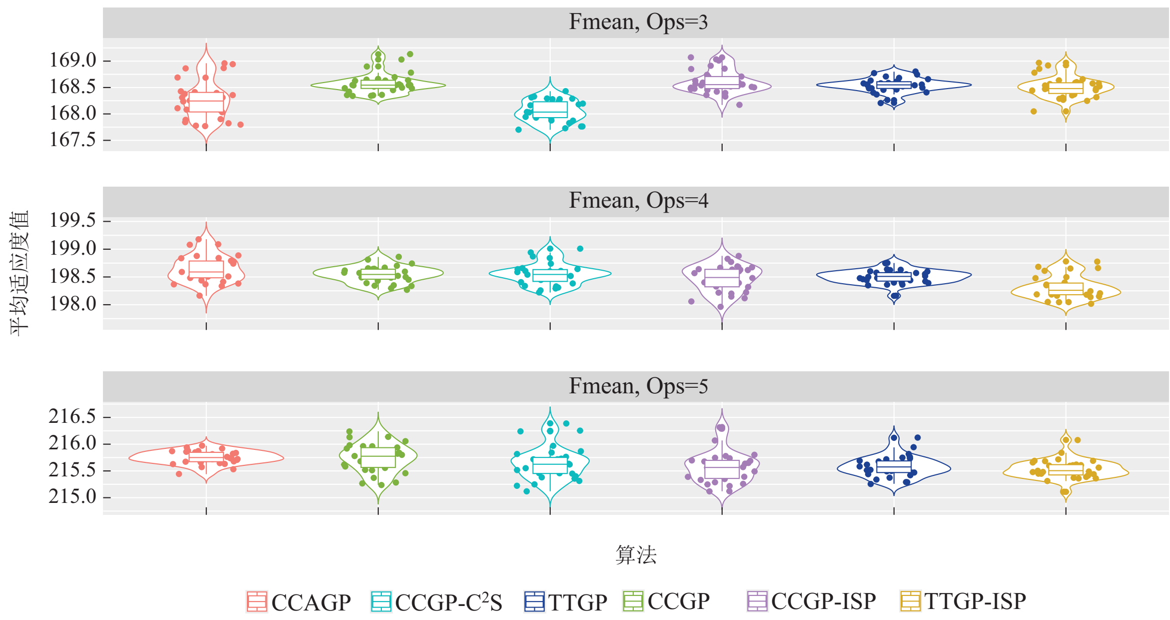


图 6　不同算法在3种生产场景中独立运行30次得到的最优适应度小提琴图

Fig. 6　Violin plots of optimal fitness for different algorithms over 30 independent runs in three production scenarios

为进一步分析算法在相同训练时间下的性能, 参考文献[32]的实验方法, 因不同算法每次迭代耗时不同, 无法通过迭代次数衡量算法相同时间内的性能. 每个场景设置大部分算法均能完成训练的基准时间, Ops = 3 为 51 min, Ops = 4 为 150 min, Ops = 5 为 300 min, 分别对应17个、50个和100个等间隔采样点 (间隔3 min), 以离采样点最近一代种群的适应度值作为该采样点的算法性能表现, 通过30次独立运行获取各采样点平均适应度值以衡量算法性能. 图7为不同算法在3种场景相同训练时间下的平均适应度曲线. 结果显示, 相同训练时间内, Ops=3, 4场景中ISP算法收敛速度略占优势; Ops=5场景中, TTGP-ISP表现更突出, 且3种场景下ISP算法在种群进化早期均能获得更优适应度值.

综上, 实验结果证明了在GP中使用ISP的有效性. 使用ISP可以通过简化个体结构缩短计算时间, 从而提升算法效率. 此外, 使用ISP不会影响种群生成个体的质量, 在所有情况下表现都不会更差, 在部分情况下可以实现更好的性能. 在相同的训练时间下收敛速度更快, 在较短的时间内可以找到更优的结果.

表 7 不同算法在3种生产场景中独立运行30次得到的最优适应度平均值、标准差以及Wilcoxon秩和检验置信区间

Table 7 The mean of optimal fitness, standard deviation, and confidence intervals from Wilcoxon rank-sum test for different algorithms over 30 independent runs in three production scenarios

生产场景	CCGP	CCGP-C^{2}S	CCAGP	CCGP-ISP	TTGP	TTGP-ISP
Fmean, Ops = 3	168.58(0.19)	168.05(0.20) (≈)(−0.03, 0.47)	168.26(0.36) (≈)(−0.17, 0.42)	168.61(0.22) (≈)(−0.11, 0.26) (≈)(−0.03, 0.40) (≈)(−0.08, 0.28)	168.55(0.13)	168.51(0.20) (≈)(−0.04, 0.13)
Fmean, Ops = 4	198.55(0.15)	198.55(0.20) (≈)(−0.08, 0.09)	198.64(0.24) (≈)(−0.20, 0.02)	198.47(0.24) (≈)(−0.05, 0.17) (≈)(−0.18, 0.06) (≈)(−0.28, 0.02)	198.50(0.12)	198.31(0.20) (≈)(−0.03, 0.31)
Fmean, Ops = 5	215.60(0.26)	215.58(0.30) (≈)(−0.03, 0.26)	215.59(0.20) (≈)(−0.09, 0.12)	215.57(0.30) (≈)(−0.05, 0.34) (≈)(−0.22, 0.06) (≈)(−0.30, 0.07)	215.56(0.19)	215.53(0.18) (≈)(−0.05, 0.13)

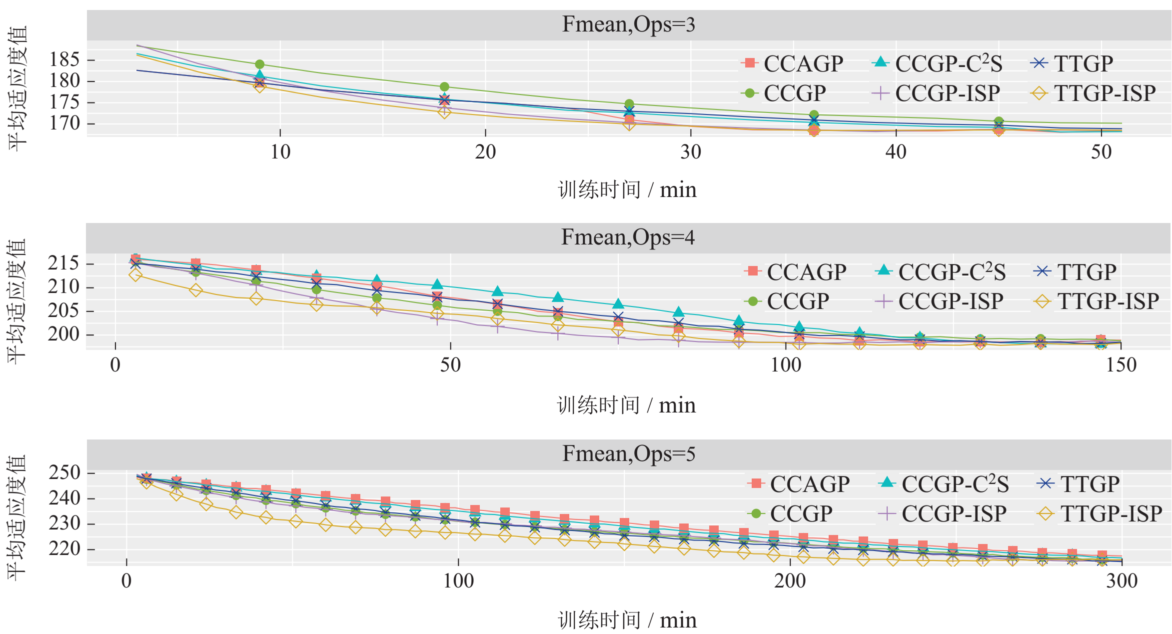


图 7 不同算法在3种生产场景中的相同训练时间下独立运行30次得到的平均适应度曲线

Fig. 7 Average fitness curves for different algorithms over 30 independent runs under identical training time in three production scenarios

5 结论

本文研究动态多柔性作业车间调度问题, 为有效解决该问题, 提出个体简化的遗传规划方法, 从动态终端节点计算耗时与数量占比两方面, 分析其对个体适应度评估耗时的影响, 通过惩罚函数削弱动态终端节点占比高的个体在子代选择中的优势, 进而缩短GP计算时间. 为验证该方法有效性, 本文在3个生产场景下对比6种算法, 结果显示, 该方法在不影响GP性能的同时, 显著缩短其计算时间. 在此基础上, 后续研究将从问题与算法两方面展开: 1) 本文基于ISP的GP方法仅验证了单目标优化问题, 未来可在GP中引入多目标优化框架以优化多个目标, 此外, 未来将研究其如何更好适应订单变更, 设备故障等动态生产环境; 2) 通过代理辅助模型替代部分适应度评估过程, 减轻计算负担, 提升GP在更复杂问题中的计算速度.

参考文献:

[1] YANG B, WANG S, LI S, et al. A robust service composition and optimal selection method for cloud manufacturing. *International Journal of Production Research*, 2022, 60(4): 1134 – 1152.

[2] LIU Qiang. Study on architecture of intelligent manufacturing theory. *China mechanical engineering*, 2020, 31(1): 24 – 36.
(刘强. 智能制造理论体系架构研究. 中国机械工程, 2020, 31(1): 24 – 36.)

[3] FARAYOLA O A, OLORUNFEMI O L, SHOETAN P O. Data privacy and security in it: A review of techniques and challenges. *Computer Science & IT Research Journal*, 2024, 5(3): 606 – 615.

[4] DAUZERE-PERES S, DING J, SHEN L, et al. The flexible job shop scheduling problem: A review. *European Journal of Operational Research*, 2024, 314(2): 409 – 432.

[5] REN W, YAN Y, HU Y, et al. Joint optimisation for dynamic flexible job-shop scheduling problem with transportation time and resource constraints. *International Journal of Production Research*, 2022, 60(18): 5675 – 5696.

[6] LÜ Yan, XU Zhengjun, LI Congbo, et al. Comprehensive energy saving optimization of processing parameters and job shop dynamic scheduling considering disturbance events. *Journal of Mechanical Engineering*, 2022, 58(19): 242 – 255.
(吕岩, 徐正军, 李聪波, 等. 考虑扰动事件的机械加工工艺参数与车间动态调度综合节能优化. 机械工程学报, 2022, 58(19): 242 – 255.)

[7] BAYKASOGLU A, MADENOGLU F S, HAMZADAY A. Greedy randomized adaptive search for dynamic flexible job-shop scheduling. *Journal of Manufacturing Systems*, 2020, 56(4): 425 – 451.

[8] LEI K, GUO P, WANG Y, et al. Large-scale dynamic scheduling for flexible job-shop with random arrivals of new jobs by hierarchical reinforcement learning. *IEEE Transactions on Industrial Informatics*, 2023, 20(1): 1007 – 1018.

[9] ZHAO L, FAN J, ZHANG C, et al. A DRL-based reactive scheduling policy for flexible job shops with random job arrivals. *IEEE Transactions on Automation Science and Engineering*, 2024, 21(3): 2912 – 2923.

[10] LIU R, PIPLANI R, TORO C. Deep reinforcement learning for dynamic scheduling of a flexible job shop. *International Journal of Production Research*, 2022, 60(13): 4049 – 4069.

[11] VITAL-SOTO A, BAKI M F, AZAB A. A multi-objective mathematical model and evolutionary algorithm for the dual-resource flexible job-shop scheduling problem with sequencing flexibility. *Flexible Services and Manufacturing Journal*, 2023, 35(3): 626 – 668.

[12] ZHANG S, LI X, ZHANG B, et al. Multi-objective optimisation in flexible assembly job shop scheduling using a distributed ant colony system. *European Journal of Operational Research*, 2020, 283(2): 441 – 460.

[13] HU Ruiqi, CHENG Hui, ZHANG Zhinan. Solving job shop scheduling problem with flexible process sequence based on expression tree model. *Computer Integrated Manufacturing System*, 2024, 30(6): 2036 – 2043.
(胡瑞淇, 程辉, 张执南. 基于表达式树的顺序柔性车间调度问题求解. 计算机集成制造系统, 2024, 30(6): 2036 – 2043.)

[14] WEI Guangyan, YE Chunming. Research on distributed and multi-flexible assembly job-shop scheduling problems. *China Mechanical Engineering*, 2023, 34(20): 2442 – 2455.
(魏光艳, 叶春明. 分布式多柔性装配作业车间调度问题研究. 中国机械工程, 2023, 34(20): 2442 – 2455.)

[15] XIONG H, FAN H, JIANG G, et al. A simulation-based study of dispatching rules in a dynamic job shop scheduling problem with batch release and extended technical precedence constraints. *European Journal of Operational Research*, 2017, 257(1): 13 – 24.

[16] OUKIL A, EL-BOURI A. Ranking dispatching rules in multi-objective dynamic flow shop scheduling: A multi-faceted perspective. *International Journal of Production Research*, 2021, 59(2): 388 – 411.

[17] MURAD S A, MUZAHID A J M, AZMI Z R M, et al. A review on job scheduling technique in cloud computing and priority rule based intelligent framework. *Journal of King Saud University-Computer and Information Sciences*, 2022, 34(6): 2309 – 2331.

[18] WANG Yan, DING Yu. Optimal scheduling and decision making method for dynamic flexible job shop. *Journal of System Simulation*, 2020, 32(11): 2073 – 2083.
(王艳, 丁宇. 动态柔性作业车间优化调度与决策方法. 系统仿真学报, 2020, 32(11): 2073 – 2083.)

[19] ZHANG Guohui, LU Xixi, HU Yifan, et al. Machine breakdown rescheduling of flexible job shop based on improved imperialist competitive algorithm. *Journal of Computer Applications*, 2021, 41(8): 2242 – 2248.
(张国辉, 陆熙熙, 胡一凡, 等. 基于改进帝国竞争算法的柔性作业车间机器故障重调度. 计算机应用, 2021, 41(8): 2242 – 2248.)

[20] SHANG Zhengyang, GU Jinan, TANG Shixi, et al. Comparative study on application performance of several metaheuristic algorithms. *Machinery Design & Manufacture*, 2021, 4: 34 – 38.
(尚正阳, 顾寄南, 唐仕喜, 等. 针对几种元启发式算法的应用性能对比研究. 机械设计与制造, 2021, 4: 34 – 38.)

[21] LUO S, ZHANG L, FAN Y. Real-time scheduling for dynamic partial-no-wait multiobjective flexible job shop by deep reinforcement learning. *IEEE Transactions on Automation Science and Engineering*, 2021, 19(4): 3020 – 3038.

[22] SHI D, FAN W, XIAO Y, et al. Intelligent scheduling of discrete automated production line via deep reinforcement learning. *International Journal of Production Research*, 2020, 58(11): 3362 – 3380.

[23] SONG W, CHEN X, LI Q, et al. Flexible job-shop scheduling via graph neural network and deep reinforcement learning. *IEEE Transactions on Industrial Informatics*, 2022, 19(2): 1600 – 1610.

[24] ZHANG F, MEI Y, NGUYEN S, et al. Survey on genetic programming and machine learning techniques for heuristic design in job shop scheduling. *IEEE Transactions on Evolutionary Computation*, 2023, 28(1): 147 – 167.

[25] ZHANG F, MEI Y, NGUYEN S, et al. Evolving scheduling heuristics via genetic programming with feature selection in dynamic flexible job-shop scheduling. *IEEE Transactions on Cybernetics*, 2020, 51(4): 1797 – 1811.

[26] ZHOU Y, YANG J, HUANG Z. Automatic design of scheduling policies for dynamic flexible job shop scheduling via surrogate-assisted cooperative co-evolution genetic programming. *International Journal of Production Research*, 2020, 58(9): 2561 – 2580.

[27] XU M, ZHANG F, MEI Y, et al. Genetic programming with archive for dynamic flexible job shop scheduling. *The IEEE Congress on Evolutionary Computation (CEC)*. Kraków, Poland: IEEE, 2021: 2117 – 2124.

[28] ZHU L, ZHANG F, FENG M, et al. Crossover operators between multiple scheduling heuristics with genetic programming for dynamic flexible job shop scheduling. *The IEEE Congress on Evolutionary Computation (CEC)*. Yokohama, Japan: IEEE, 2024: 1 – 8.

[29] CHEN X, LI J, WANG Z, et al. Optimizing dynamic flexible job shop scheduling using an evolutionary multi-task optimization framework and genetic programming. *IEEE Transactions on Evolutionary Computation*, 2025, 29(5): 1502 – 1516.

[30] SCHWEIM D, WITTENBERG D, ROTHLAUF F. On sampling error in genetic programming. *Natural Computing*, 2022, 21(2): 173 – 186.

[31] ZHANG F, MEI Y, NGUYEN S, et al. Correlation coefficient-based recombinative guidance for genetic programming hyper heuristics in dynamic flexible job shop scheduling. *IEEE Transactions on Evolutionary Computation*, 2021, 25(3): 552 – 566.

[32] ZHANG F, MEI Y, NGUYEN S, et al. Collaborative multifidelity-based surrogate models for genetic programming in dynamic flexible job shop scheduling. *IEEE Transactions on Cybernetics*, 2021, 52(8): 8142 – 8156.

作者简介:

邱佳龙　硕士研究生, 目前研究方向为智能算法与优化调度, E-mail: 1307541283@qq.com;

龚文引　教授, 博士生导师, 目前研究方向为智能优化与应用, E-mail: wygong@cug.edu.cn;

张国辉　教授, 硕士生导师, 目前研究方向为工业工程、能优化算法与车间调度, E-mail: zgh09@zua.edu.cn;

卢　超　副教授, 博士生导师, 目前研究方向为智能绿色调度、分布式车间调度、智能优化算法与路径规划, E-mail: luchao@cug.edu.cn.