文章编号 1672-6634(2022)02-0014-13 **DOI** 10.19728/j.issn1672-6634.2021060015

帝国竞争算法求解资源约束混合流水车间调度问题

李俊青,李荣昊,陶昕瑞,曾清清,耿雅典

(聊城大学 计算机学院,山东 聊城 252059)

摘 要 资源约束的混合流水车间问题(RCHFS)近年来得到了广泛的研究。然而,对于同时考虑资源约束和能源消耗的实际案例却仍然研究甚少。对此本文提出离散帝国主义竞争算法(DICA)来解决最小化完工时间和能源消耗的RCHFS问题。首先,设计了适应该问题的两阶段编码机制。其次,设计了一种考虑资源分配的解码方法。最后,将DICA和模拟退火算法(SA)相结合用来提高算法的性能。除此之外,我们基于随机生成的真实车间调度系统实例进行仿真实验,并且与现有的启发式算法进行了比较。实验结果表明所提出的算法可以高效的解决RCHFS问题。

关键词 混合流水车间;帝国主义竞争算法;资源约束

中图分类号 TP393

文献标识码 A

开放科学(资源服务)标识码(OSID)

Research on Resource Constrained Hybrid Flowshop Scheduling Problem Based on Imperialist Competitive Algorithm

LI Junqing,LI Ronghao,TAO Xinrui,ZENG Qingqing,GENG Yadian

(School of Computer Science,Liaocheng University,Liaocheng 252059,China)

Abstract The resource-constrained hybrid flowshop problem (RCHFS) has been investigated thoroughly in recent years.However,the practical case that considers both resource-constrained and energy consumption still has rare research.To address this issue,a discrete imperialist competitive algorithm (DICA) was proposed to minimize the makespan and energy consumption.In the proposed algorithm,first,each solution was represented by a two-dimensional vector,where one vector represented the scheduling sequence and another one showed the machine assignment.Then,a decoding method considering the resource allocation was designed.Finally,we combined DICA and the simulated annealing algorithm (SA) to improve the performance of the proposed approach.Furthermore,we tested the proposed algorithm based on a randomly generated set of real shop scheduling system instances and compared with the existing heuristic algorithms.The results confirmed that the proposed algorithm can solve the RCHFS with high efficiency.

Key words hybrid flowshop;imperialist competitive algorithm;Resource-constrained

收稿日期:2021-06-30

基金项目:国家自然科学基金项目(61773192)资助

通讯作者:李俊青,男,汉族,博士,教授,研究方向:网络安全智能、柔性车间调度,E-mail:lijunqing@lcu-cs.com。

0　引言

混合流水车间调度问题（Hybrid Flowshop Scheduling Problem，HFSP）是传统流水车间调度问题的扩展。混合流水车间调度问题的多进程和多阶段的特点与其他类型的调度问题相比更加具有现实意义。在典型的生产制造业中，实际生产过程中可以会发生各种各样的动态事件，如资源受限、机器故障等。因此，研究资源约束混合流水车间调度问题（Resource-constrained Hybrid Flowshop Problem，RCHFS）具有一定的现实意义。此外，制造业的生产活动所产生的能源消耗作为全球变暖的主要来源，应该满足环境保护和能源消耗的相关规定。本文研究了完工时间和能源消耗的资源受限混合流水车间问题，并且这个问题具有重要的实践意义。

Rubén Ruiz 等人已经对于传统的 HFSP 进行了许多对于变体形式和解决方案的讨论[1]。一些研究人员采用了精确算法去解决 HFSP 问题，如拉格朗日松弛算法[2-4]和分支定界算法[5-7]。然而随着问题规模的增加，精确算法的效率已经受到了限制，因此，元启式算法和启发式算法得到了更广泛的应用，Behnamian 等人[8]和 Dugardin 等人[9]运用的遗传算法（Genetic Algorithm，GA）。Elmi 和 Topaloglu 采用模拟退火算法（Simulated Annealing，SA）求解多机器人调度问题[10,11]。Figelska 考虑了两阶段混合流水车间问题的禁忌搜索算法[12]。Marichelvam 提出了离散萤火虫算法求解双目标 HFSP[13]。Naden 和 Yazdani 考虑了帝国主义竞争算法来解决带有子模块和准备时间窗的 HFSP[14]。此外，混合人工蜂群算法[15]、混合果蝇优化算法[16]、混合松鼠搜索算法[17]等也都在 HFSP 研究领域得到了应用。

RCHFS 作为 HFSP 的一种变体形式，几年来一直受到许多研究者的关注和研究。Nishi 开发了拉格朗日分解和协调方法[18]，Sural 等人提出了分支定界算法来解决双资源约束的调度问题[19]，Pei 等人提出了蝙蝠算法和变邻域搜索相结合的方法（Bat Algorithm combined with Variable Neighborhood Search，BA-VNS）[20]。Li 等人研究和改进了人工蜂群算法（Artificial Bee Colony Algorithm，ABC）和两阶段编码机制[21-23]。Cheng 等人针对不同的情况提出了多项式算法[24]。Leu 和 Hwang 提出了基于遗传算法的搜索算法，并据此进行了灵敏度分析[25]。

近年来，绿色调度在制造业中已经成为研究的主要方向之一。Gao 等人[26]提供了一个完整的关于带有能源目标的智能制造生产调度系统的文献综述，提出了基于遗传规划的生产调度启发式自动设计的统一框架[27]。Dai 等人采用了一种 genetic-SA 的方法[28]并在此问题求解上取得了重大进展。同时，基于分解的多目标进化算法[29]也是被许多学者研究。Li 等人提出了一种基于 Pareto 的多目标优化算法[30]。Lei 等人提出了一种新颖的基于教与学优化算法（Teaching-Learning-Based Optimization，TLBO）[31]用来最小化总能耗和总延迟。Ding 等人提出了求解双目标问题的多目标迭代贪婪算法[32]。此外，混合迭代贪心算法[33]、高效多目标算法[34]和改进的 Jaya 算法[35]也用于多目标问题的求解。许多专家还研究了能量感知模型[36-38]、机器开关方案[39]和不同类型的机器约束[40]问题等等。

自混合流水车间问题提出以来，早期的研究一般侧重于生产效率的优化，随着生产力的提高，能源需求量越来越多，相应的能耗造成的环境问题也日益严重。无论从经济成本还是环境影响方面来考虑，对制造业中能耗目标加以优化已经是大势所趋。在 RCHFS 问题中考虑了能耗目标，将完工时间和能耗加权求和，所以设计了基于离散帝国主义竞争算法（Discrete Imperialist Competitive Algorithm，DICA）和模拟退火算法（SA）相结合的算法，对该问题进行求解。

目前的多约束 HFSP 大多考虑的是机器或者时间的约束[41]，然而在实际的工业生产中往往需要人力或者其它类型的资源，针对这方面的研究相对较少。由于实际的生产工艺和加工环境的复杂性，多约束 HFSP 更加贴近实际生产调度，但多样化的约束、假设条件等也使得问题求解变得更加复杂，变量增加、建模难度增大、约束条件增多等导致了可行解很难寻找，全局最优解的搜索更加困难。此外，要在解决问题的基础上考虑算法的性能，使其更加充分的贴近实际工业生产的需要，加大了对于思考算法处理问题时的难度。

本文主要贡献如下：（1）考虑了最小化完工时间和总能耗的 RCHFS 问题，并提出了相应的数学规划模型。据我们对研究现状的分析可知，最小化完工时间和总能耗的 RCHFS 问题是第一次被提出。（2）采用了一种新颖的 DICA 方法来解决带能耗的 RCHFS 问题。此外，还设计了适应这一问题的编码和解码策略。

(3) 将 DICA 和 SA 相结合来提高算法的性能。

本文中首先描述了经典帝国主义竞争算法(Imperialist Competitive Algorithm,ICA)和改进算法以及编码和解码方法。然后,在随机生成的一组实例上进行的仿真实验,并将得到的结果与其他几种算法的结果进行比较分析。最后,对本文的工作进行总结,并展望了未来可行的研究方向。

1 问题描述

1.1 问题建模

所研究的问题可以描述如下:工件需要经过不同阶段的机器处理,在整个过程中,至少有一个阶段存在多于两台的并行机。第一阶段加工完成后,第二阶段才能继续加工。如果第一阶段的处理完成,工件可以暂存在无限容量的缓冲空间中,直到机器在第二阶段可用为止。工件可以在后续阶段上的任意一台机器上加工,开始后加工不能中断。

工厂里有 h 种资源,包括 $R_1,R_2,\cdots,R_h$ 。每种类型的资源数量都是预先给出的,每台机器的加工来自不同类型资源的合作。因此,为了开始加工,必须确保机器和资源同时可用。

在车间之中,至少有一个阶段存在两台或以上的并行机。其中每台机器具有两种状态,即处理状态和待机状态,不同状态消耗的能量是不同的。目标是使完工时间和能源消耗最小化。

1.2 变量描述

(1) 下标。i:工件下标,$i=1,2,\cdots,n$;k:机器下标,$k=1,2,\cdots,m$;j:加工阶段下标,$j=1,2,\cdots,g$;q:工件加工位置下标,$q=1,2,\cdots,n$;r:工序数,$r=1,2,\cdots O$;res:机床加工需要的资源,$res=1,2,\cdots h$。

(2) 参数。n:工件总数;m:机器总数;s:加工阶段总数;h:资源类型数量;L:一个很大的数;$p_{i,j}$:工件 i 在加工阶段 j 上的加工时间;O:工序总数;pe_k:机器 k 加工时单位能耗;se_k:机器 k 空闲时单位能耗。

(3) 决策变量。$B_{k,q}$:加工机床 k 在加工位置 q 的开工时间;$E_{k,q}$:加工机床 k 在加工位置 q 的完工时间;$b_{i,j}$:工件开工时间;$e_{i,j}$:工件完工时间;$rb_{r,res}$:资源开工时间;$re_{r,res}$:资源完工时间;$Y_{k,i,j,q}$:加工阶段,加工机床上工件和加工位置的对应关系;$R_{i,j,r,res}$:资源在某个位置对应的工序编号;$RM_{r,k,res}$:资源在某个位置加工的机床 k;TEC:总能耗。

1.3 数学模型

根据上述变量和下标,所建立的数学模型如下。

最小化目标

$$w * C_{\max} + (1-w) * TEC \ , \tag{1}$$

约束条件

$$B_{k,q+1} - E_{k,q} + L * (1 - Y_{k,i',j,q}) \geqslant 0 \ , \tag{2}$$

$$e_{i,j} = b_{i,j} + p_{i,k} \ , \tag{3}$$

$$e_{i,j} \leqslant b_{i,j+1} + L * (1 - \sum_{q=1}^{n} Y_{k',i,j+1,q}) \ , \tag{4}$$

$$B_{k,q} + \sum_{i=1}^{n} (p_{i,k} * Y_{k,i,j,q}) = E_{k,q} \ , \tag{5}$$

$$E_{k,q} \leqslant B_{k,q+1} \ , \tag{6}$$

$$B_{k,q} \leqslant b_{i,j} + (1 - Y_{k,i,j,q}) * L \ , \tag{7}$$

$$B_{k,q} + (1 - Y_{k,i,j,q}) * L \geqslant b_{i,j} \ , \tag{8}$$

$$\sum_{k=1}^{m} \sum_{q=1}^{n} Y_{k,i,j,q} = 1 \ , \tag{9}$$

$$\sum_{i=1}^{n} Y_{k,i,j,q} \leqslant 1 \ , \tag{10}$$

$$\sum_{i=1}^{n} Y_{k,i,j,q} \geqslant \sum_{i'=1}^{n} Y_{k,i',j,q+1} \ , \tag{11}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{s} R_{i,j,r,res} \geqslant \sum_{i'=1}^{n} \sum_{j'=1}^{s} R_{i',j',r+1,res} \ , \tag{12}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{s}R_{i,j,r,res}\leqslant 1\ , \tag{13}$$

$$\sum_{r=1}^{O}R_{i,j,r,res}=R_{i,res}\ , \tag{14}$$

$$\sum_{k=1}^{m}RM_{r,k,res}=1\ , \tag{15}$$

$$rb_{r,res}\geqslant re_{r-1,res}\ , \tag{16}$$

$$\begin{aligned} re_{r,re}&\leqslant e_{i,j}+L*(1-R_{i,j,r,res}),\\ e_{i,j}&\leqslant re_{r,res}+L*(1-R_{i,j,r,res}),\\ rb_{r,res}&\leqslant b_{i,j}+L*(1-R_{i,j,r,res}),\\ b_{i,j}&\leqslant rb_{r,res}+L*(1-R_{i,j,r,res}), \end{aligned} \tag{17}$$

$$RM_{r,k,res}\geqslant R_{i,j,r,res}-M*(1-\sum_{q=1}^{n}Y_{k,i,\ j,q})\ , \tag{18}$$

$$TEC=E_1+E_2\ , \tag{19}$$

$$E_1=\sum_{i=1}^{n}\sum_{k=1}^{m}\sum_{j=1}^{s}\sum_{q=1}^{n}p_{i,k}*Y_{k,i,j,q}*pe_k\ , \tag{20}$$

$$E_2=\sum_{j=1}^{s}\sum_{k=1}^{m}\sum_{q=1}^{n}(B_{k,q+1}-E_{k,q}-\sum_{i=1}^{n}Y_{k,i,j,q}*Y_{k,i',j,q+1})*se_k\ 。 \tag{21}$$

约束(1)表示目标为最小化完工时间。约束(2)确保同一阶段分配在同一机器上的排位靠后的工件必须等靠前的工件加工完后才可进行。约束(3)表示同一阶段上开始时间和完工时间的关系。约束(4)表示工件完成上一阶段的加工后才能进行下一个阶段加工。约束(5)表示一个阶段中工件一旦开始加工就不能中断。约束(6)表示每台机器开工时间和完工时间对应关系。约束(7)和约束(8)表示机器开始时间和工序时间对应关系。约束(9)表示同一阶段工件只能在一台机器上加工。约束(10)确保每台机床同一时刻只能加工一个工件。约束(11)和约束(12)保证同一阶段调度排列中优先级高的先加工。约束(13)-约束(16)表示同一时刻加工需要资源数量不超过单位总量。约束(17)表示资源的使用时间与对工序的时间关系。约束(18)表示机床与资源之间的对应关系。约束(19-21)表示的机器的加工能耗，其中 E_1 表示机器加工阶段的能耗，E_2 表示机器空闲阶段的能耗。

2　算法描述

2.1　问题编码

针对考虑能耗的资源约束混合流水车间调度问题(ERHFS)，使用基于双向量(TVB)解的表示和基于机器甘特图(MGB)解的表示的两阶段编码机制。在早期的进化阶段，基于双向量的表示可以识别具有广泛搜索能力的并且有前途的搜索空间。在后期的进化阶段，使用基于机器甘特图的解的表示来编码每台机器的详细调度，并通过足够大的搜索空间来进行搜索。两种编码策略的细节描述如下：

(1) TVB 解的表示。在编码表示中考虑了机器分配和操作排序，第一个向量为机器分配向量，长度为 $n\times g$ ，其中每个元素标识了每个工件所分配的机器号。如图 1 所示，机器分配向量为{1,1,1,1,1,2,3,2,3,2}，向量第一个元素为 1，表示加工的第一个阶段中，1 号工件被分配给了 1 号机器。

调度向量为{1,2,3,4,5}，定义了第一阶段的加工顺序。在第一阶段加工完成后，分配的机器和资源都可用时，立即将每个工件转移到下一个阶段进行加工。此外，由于机器和资源的占用，加工顺序可能在后期发生变化。

(2) MGB 解的表示。第二种向量 MGB 解的表示是根据当前的机器甘特图设计的，基于 MGB 解的表示中每个阶段的每台机器都有一个向量，每个向量表示了当前机器上的调度序列。对于图 2 中所举的示例，MGB 解的表示为{{1,2,3,4,5},{{1,3,5},{2,4}}}，其中第一个向量为第一个阶段中 1 号机器的编码，第二个向量为第二阶段中 2 号机器和 3 号机器上的编码。TVB 解的表示和 MGB 解的表示主要区别在于，MGB 解的编码可以识别每个阶段中每台机器上的工件排序，而 TVB 解的编码只识别第一阶段中的工件排序。

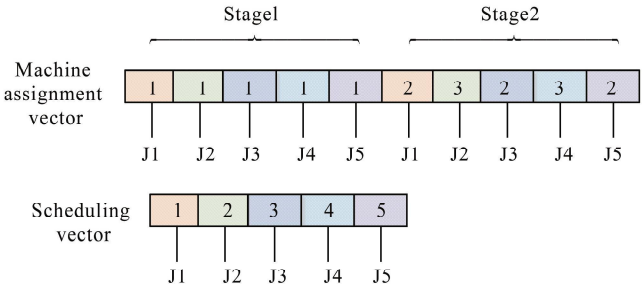


图 1 TVB 解的表示举例

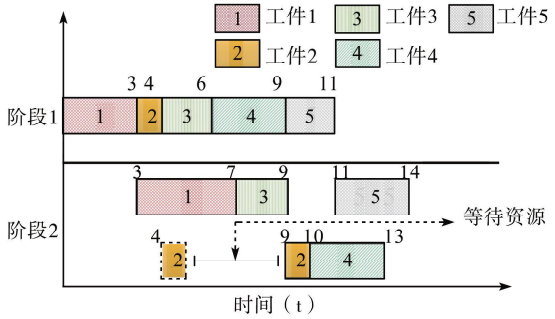


图 2 机器甘特图

2.2 问题解码

在解码过程中，我们使用 TVB 和 MGB 解决方案表示来确定每个操作的开始时间以及每台机器的合适资源。开始操作需要同时满足以下三个条件：工件到达指定机器；指定机器可用；指定机器所需的资源可用。

所有工件在当前加工阶段完工后，需要根据当前资源是否可用做出如下选择：如果当前资源可用，则安排到下一阶段加工；否则推迟该工件，直到资源可用再开始加工，机器间的运输时间忽略不计。

计算工件的开始加工时间时，需要考虑：(1) 上一阶段 $e_{i,j-1}$ 的完成时间；(2) 机器可用时间 I_k；(3) k 机器所需资源的最大可用时间。因此，$b_{i,j}$ 可以计算如

$$b_{i,j}=\max(b_{i,j-1}+p_{i,j-1},I_k,A_{R_k}), \tag{22}$$

$$A_{Rk}=\max_{r\in R_k}(A_r), \tag{23}$$

其中 A_r 是资源 $r\in R_k$ 的可用时间，A_{R_k} 是机器 k 所需的所有资源的最大可用时间。如果 $O_{i,j}$ 是工件 i 的第一个操作，则开始时间 $b_{i,j}$ 可以计算如

$$b_{i,j}=\max(I_kA_{R_k})。 \tag{24}$$

$O_{i,j}$ 完成后，R_k 资源将被消耗并立即释放以用于其他操作。因此，R_k 和 I_k 的可用时间将更新如

$$I_k=A_{R_k}=b_{i,j}+p_{i,j}。 \tag{25}$$

在所有的工件都安排好之后，总完工时间计算如

$$C_{\max}=\max_{i=1,\cdots,n}(b_{i,s}+p_{i,s})。 \tag{26}$$

2.3 解集初始化

初始化过程分为四步：(1) 将所有解存放在数组中作为初始的国家，并计算其标准化国家势力值。(2) 选出 N_{imp} 个国家势力值较大的国家作为帝国，剩下的 N_{col} 个国家作为殖民地。(3) 对选出来的 N_{imp} 个帝国，计算其标准化帝国势力值。(4) 根据标准化帝国势力计算帝国所拥有的殖民地数量，并将相应数量的殖民地分配给相应的帝国。初始帝国数量为解集大小的 20%。

采用贪心的策略，将 fitness 较小的几个解作为帝国，其余的作为殖民地，这样可以在一定程度上加快算法的收敛速度。整个初始化的过程就是将所有解划分为殖民地或者帝国，并将殖民地分配给相应的帝国。

2.4 改进帝国同化行为

为了使 ICA 算法适用于求解离散的优化问题，提出了离散帝国同化过程。在同化过程中，随机选择以下两种交叉方式中的一种来实现解的变换。

第一种交叉方式，对每个帝国都进行交叉处理。其中一个父代为帝国，另一个父代为该帝国的殖民地，让帝国与其拥有的每个殖民地都进行交叉，计算获得子代的目标值。这里的交叉策略具体步骤如。

步骤 1 如图 3 所示，随机选择一个帝国和他的殖民地(父代)中几个工件编号的起止位置(两父代个体被选位置相同)。

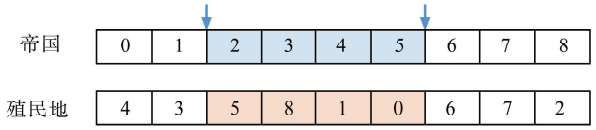


图 3 随机选取帝国和他的殖民地

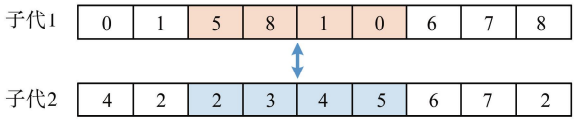


图 4 交换过程

步骤 2 如图 4，交换帝国和殖民地的位置。

步骤 3　看生成的子代工件编号是否重复，如果重复就把帝国和殖民地中选取的两组工件编号建立相应的映射关系，如图 5 所示，即{2,5,0},{8,3},{1,4}。第二步中子代 1 交换完成后出现了两个工件 1，可以通过映射关系把 1 转换为工件 4 并以此类推，直到最终形成的子代里工件编号只出现一次。

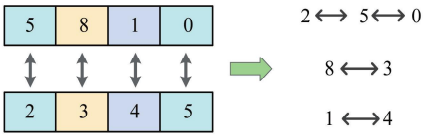


图 5　冲突检测

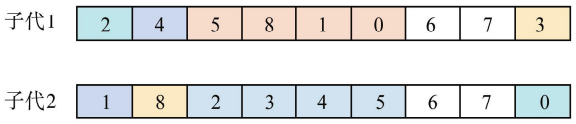


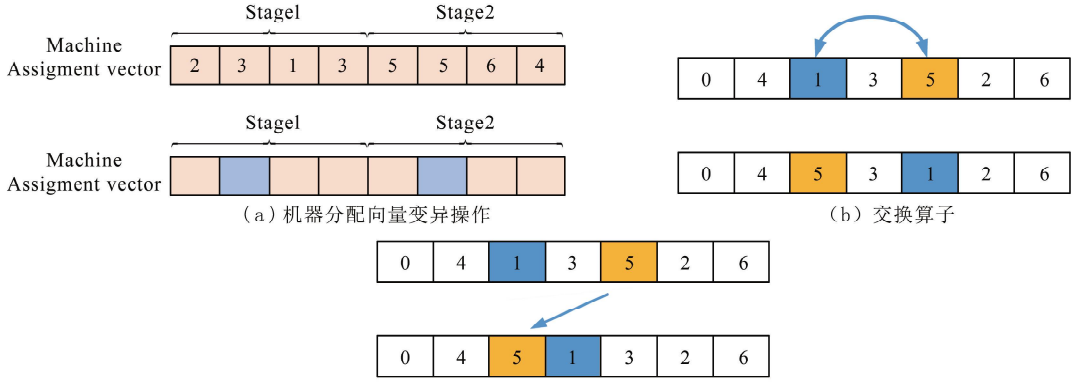
图 6　交叉完的帝国和殖民地

最终结果如图 6 所示。

第二种交叉方式，不区分帝国或殖民地，随机选取两个国家作为父代进行交叉，同样采用上述的交叉操作。另外，在帝国对内同化的过程中，有的殖民地不满帝国的统治想要进行革命，革命成功则会成为帝国，革命失败则继续是殖民地，革命的过程类似于变异的过程，这里提出了一种变异策略。

（1）对于机器分配向量随机选择几个位置，更换工件所对应的机器。

（2）对于调度向量，随机选择交换算子和插入算子其中的一个。对于交换操作，在调度向量中随机选择两个工件，然后交换两个工件以生成不同的子代。对于插入算子，在调度向量中随机选择两个工件，删除其中一个工件并将其插入到另一个选定工件的前一个位置。具体过程如图 7 所示。



（a）机器分配向量变异操作　　（b）交换算子

（c）插入算子

图 7　变异操作

2.5　改进帝国竞争行为

帝国想要发展除了对内的同化统治，还需要对外扩张，帝国竞争是算法的全局搜索过程。所有帝国都希望能够占有其他帝国的殖民地并控制它们，这种帝国之间的竞争行为会逐渐导致较弱的帝国势力下降而强大的帝国势力增加。通过挑选较弱帝国的殖民地，将它分配给其他较强的帝国来模拟帝国竞争的过程。如何挑选较弱的帝国，提出了两种策略。

算法 1　帝国竞争策略 1

输入：不可行的解
输出：可行的解

1. **For** 每一个帝国j **do**
2. 　　按殖民地数量排序
3. **End**
4. 殖民地数量少的作为弱小帝国，反之则为强大帝国
5. **For** 每一个帝国的殖民地的 i **do**
6. 　　按照弱小帝国的顺序破坏殖民地
7. 　　从弱小帝国中随机选择殖民地归入强大的帝国中
8. **End**

算法 2　帝国竞争策略 2

输入：不可行的解
输出：可行的解

1. 　　**For** 每一个帝国j **do**
2. 　　　　**For** 每个帝国的每个殖民地 **do**
3. 　　　　　　按目标值总和排序
4. 　　　　**End**
5. 　　　　根据帝国的总体适应性将帝国从小到大进行排序
6. 　　**End**
7. 　　目标值大的作为弱小帝国，反之则为强大帝国
8. 　　**For** 每一个帝国的殖民地 i **do**
9. 　　　　为了最弱小的帝国破坏殖民地
10. 　　　　从弱小帝国中随机选择殖民地归入强大的帝国中
11. 　　**End**

首先,殖民地较少的帝国可以作为弱小帝国。按照帝国殖民地的数量排序,找到一些弱小帝国打乱他们的殖民地顺序,随机选择一个殖民地归属到强大的帝国中。其次,还可以将帝国殖民地的目标值从小到大排序,选择较大目标值的国家作为弱小帝国,然后打乱弱小帝国的殖民地顺序,并随机选择一个殖民地归属到强大的帝国中。

2.6 帝国灭亡

如果一个帝国失去了所有的殖民地,就会灭亡。一段时间以后,所有的殖民地都被纳入最后的帝国之中。通常情况下,经典 ICA 是将灭亡的帝国直接删除,DICA 中把灭亡的帝国作为殖民地划分给强大的帝国,这样能够增加解的多样性。具体步骤如下。

步骤 1 对于每一个帝国,判断其殖民地数量。若殖民地数量小于 1,则执行步骤 2;否则,执行帝国竞争过程。

步骤 2 将该帝国作为殖民地划分给其他帝国。

步骤 2.1 计算帝国总势力值。

步骤 2.2 该帝国作为殖民地划分给最强大的帝国。

步骤 2.3 将该帝国从帝国里面删除并添加到殖民地中。

步骤 3 检测是否还有其它帝国。

步骤 3.1 若还有其他帝国,回到步骤 1。

步骤 3.2 若没有其它帝国,记录当前最优解。

2.7 全局搜索过程

在经典的 ICA 算法中,帝国竞争阶段完成以后,帝国如果失去了所有的殖民地就会灭亡,相应的会在所有的解中将这个帝国删除,保留其它帝国。这样虽然提高了算法的收敛速度和局部搜索能力,但是解的多样性相对减少,算法容易陷入局部最优。因此将 DICA 算法与模拟退火算法(Simulated Annealing,SA)相结合,进一步提升算法的全局搜索能力。

算法迭代时的温度根据公式(27)定义

$$T_k = \alpha T_{k-1} = \alpha^k T_0 \ , \tag{27}$$

其中 $0 < \alpha < 1$ 是温度的降低速率。很显然,α 越小,温度下降的越慢。SA 的算法流程如算法 3 所示。

算法 3 模拟退火算法

输入:局部最优解

输出:全局最优解

1. 初始化温度
2. 创建一个随机解决方案 x 和评估函数 $f(x)$
3. 在 x_{best} 保存 x 的最优解
4. While 标准不满足时停止
5. 创建邻域解决方案 x^{new} 和评估函数 $f(x^{new})$
6. 接受 x^{new} 的概率为 $P(x, x^{new}, T)$
7. If x 比 x_{best} 更优,令 x_{best} 存储 x
8. 降低温度
9. End while

算法 4 邻域创建方案

输入:帝国主义

输出:改进的帝国主义

1. 概率是 1/2,执行步骤 2;否则执行步骤 3。
2. 在 vector(机器分配向量)中选择一个随机元素,并将其替换为区间[0,1]中的一个随机数。然后执行步骤 6。
3. 概率是 1/2,执行步骤 4;否则执行步骤 5。
4. 在(调度向量)中选择两个不同的元素并交换它们的位置。执行步骤 6。
5. 选择两个不同的元素,并颠倒它们之间的顺序(包括它们自己)。
6. End

通常情况下,SA 通过在邻域结构中创建新的解,迭代寻找更优的解。不同的编码方式有不同的邻域结构。因此,根据 RCHFS 的问题编码定义了一个随机过程来创建邻域,创建邻域的过程如算法 4 所示:

2.8 算法框架

针对 ERCHFS 问题的帝国竞争算法的具体框架如下。

步骤 1 系统设置阶段。

步骤 1.1　系统参数设置。

步骤 1.2　初始化解集。

步骤 2　帝国初始化阶段。

步骤 2.1　计算解集中解的目标值，将当前解集 N_{pop} 中最好的 N_{imp} 个解作为帝国，剩下的 N_{col} 个解作为殖民地。

步骤 2.2　计算帝国势力大小，根据帝国势力大小计算其拥有的殖民地数量。

步骤 2.3　将对应数量的殖民地随机分配给帝国。

步骤 3　如果满足停止条件，则保存至当前最好解；否则执行步骤 4 到 6。

步骤 4　帝国同化阶段。

步骤 4.1　执行 2.4 章节中的交叉和变异策略。

步骤 4.2　重新评价当前的帝国和殖民地。

步骤 5　帝国竞争阶段。

步骤 5.1　计算帝国总势力，根据帝国总势力计算每个帝国占有殖民地的概率。

步骤 5.2　从较弱帝国随机选择一个殖民地按一定概率加入其他帝国。

步骤 6　帝国灭亡阶段。

步骤 6.1　若帝国还有殖民地，重复执行步骤 4 到 6。

步骤 6.2　若帝国失去所有殖民地，则将帝国作为殖民地加入其他帝国。

步骤 7　检测是否只剩一个帝国。

步骤 7.1　若还有其他帝国，重复步骤 4 到 7。

步骤 7.2　若只剩一个帝国，记录至当前最优解。

步骤 8　全局搜索过程。

步骤 8.1　从 2.7 节中的邻域结构随机生成一个新解。

步骤 8.2　若新生成的解比当前最优解好，则将新解记录至最优解。

步骤 8.3　若新解比当前最优解差，则按概率 $P(x,x^{new},T)$ 保留新解。

步骤 9　回到步骤 8。

2.9　算法复杂度分析

根据算法的全局分析可知，在种群规模为 n，最大迭代 m 次时，算法的时间复杂度均为 $O(n\log n+n*m)$。

3　实验分析

3.1　实验设置

以 Visual Studio 2015 为开发环境，使用 Intel i5 2.40 GHz CPU、16 G 内存的电脑上进行实验。为了解决 RCHFS 问题并验证 DICA 的有效性，在经典的混合流水车间问题算例的基础上生成了 20 个 ERCHFS 问题的大规模测试用例。算例中工件的数量为(50、100、150、200)，阶段的数量为(2、4、6、8、10)。另外，在 RCHFS 算例的基础上加入了每台机器加工和空闲阶段的单位能耗。例如，算例 1 规定了车间中有 50 个工件、9 台加工机器、工件需要 2 个阶段的加工、总共有 3 种资源、每种资源总量都是 4 个单位，另外还规定了 9 台加工机器各自加工和空闲阶段的单位能耗。实验所得到的结果都是针对每个算例独立运行 30 次取得的平均值，并采用相对百分比增长(RPI)指标来作为算法性能分析比较的标准。

3.2　实验参数

实验中，DICA 主要参数有三个，分别是(1) 初始解集大小(P_s)；(2) 交叉概率(P_c)，它表示了算法中每个个体交叉的概率；(3) 变异概率(P_m)，它规定了算法中个体变异的概率。

采用实验设计 DOE Taguchi 方法构造了一套正交阵列 L_{16} 对参数进行组合，参数组合如表 1 所示。图 8给出了因子水平和交点。经过计算，DICA 算法表现最优的参数组合为 50，0.7，0.05。

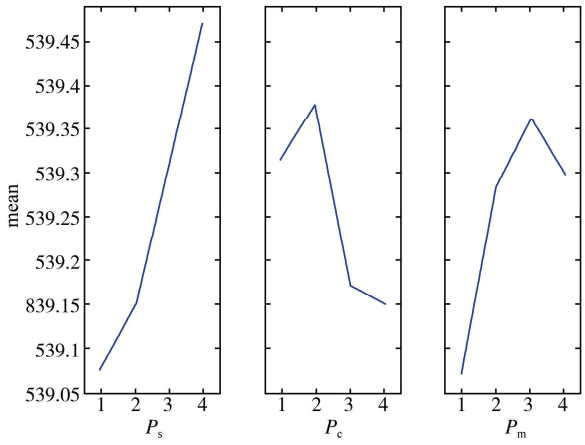


图 8 三个关键参数的因子水平趋势

表 1 参数值设置

参数	取值			
	1	2	3	4
P_s	50	100	150	200
P_c	0.10	0.30	0.50	0.70
C_n	0.05	0.10	0.20	0.30

DICA 表现最优的组合中三个参数的取值分别为 50、0.7、0.05。其它算法所用的参数设置均得自相应的参考文献,所有算法均使用第 3 章中的编码解码策略。

表 2 DICA 与 CPLEX 求解器比较

Ins	Scale	Best	Fitness		RPI	
			DICA	CPLEX	DICA	CPLEX
Inst 1	4-3-2	332.94	**332.94**	351.56	**0.00**	**5.59**
Inst 2	4-4-2	340.72	**340.72**	342.16	**0.00**	0.42
Inst 3	5-4-2	355.92	**355.92**	377.24	**0.00**	5.99
Inst 4	5-5-3	494.06	**494.06**	500.00	**0.00**	**1.20**
Inst 5	6-4-2	509.38	**509.38**	525.46	**0.00**	3.16
Inst 6	7-4-2	481.51	**481.51**	494.04	**0.00**	2.60
Inst 7	7-5-2	495.85	**495.85**	511.40	**0.00**	3.14
Inst 8	7-5-3	319.53	**319.53**	-	**0.00**	-
Inst 9	8-4-2	400.07	**400.07**	416.62	**0.00**	4.14
Inst 10	8-5-3	339.16	**339.16**	-	**0.00**	-
	Mean		**406.91**	439.81	**0.00**	3.28

3.3 小规模算例验证

为了验证所提出的优化模型的有效性,通过精确求解器 IBM ILOG CPLEX 12.7 对 10 个小规模算例进行了计算。小规模算例是根据 3.1 节所用的算例随机生成的。在精确求解器中,最大线程数为 3,每次运行的 CPU 时间限制设置为 3 h。

表 2 显示了 DICA 算法和 CPLEX 求解器的比较结果。表格第一列表示算例编号,第二列表示问题规模(其中 4-4-2 表示实例包括 4 个工件、4 台机器和 2 个阶段)。对每个实例运行 30 次 DICA 算法并取平均值。最后两列显示了这两个方法的 RPI 值。可以看到:(1) 提出的 DICA 算法获得了较高质量的解;(2) 对于较大规模的计算实例,CPLEX 的求解性能的表现不如 DICA。("-"表示 CPLEX 解算器在 3 h 内找不到可行解,粗体表示比较算法中的最优值),性能指标为 RPI。所得结果如下。

3.4 与现有算法进行比较

最后,为了评估提出的 DICA 算法的效率,将其与离散人工蜂群算法(Discrete Artificial Bee Colony Algorithm,DABC)[42] 和带遗传算法的混合帝国主义竞争算法(Hybrid Imperialist Competitive Algorithm with Genetic Algorithm,ICA-G)[43] 进行了比较。之所以选择这三种方法是因为(1) DABC 算法在资源受限的混合流水车间问题里已经得到了有效证明;(2) ICA-G 虽然没有在 RCHFS 问题里得以应用,但是在混合流水车间问题里也已经有了一定的研究,只需稍作修改就可以用于求解提出的 ERCHFS 问题。上述对

比算法参数取自于各自文献。

为了让完工时间和总能耗在相同大小的范围内，对两个目标进行了归一化处理，对完工时间和总能耗分别使用公式(28)和(29)计算。加权和的最终目标函数如式(30)所示。

$$F_{\text{value}} = F / F_{\max} \ , \tag{28}$$

$$E_{\text{value}} = E / E_{\max} \ , \tag{29}$$

$$\min(f) = w * F_{\text{value}} + (1 - w) * E_{\text{value}} \ , \tag{30}$$

其中 $F_{\max}$ 和 $E_{\max}$ 分别是当前种群中最大的完工时间和最大的总能耗，w 是反映两者相对重要性的权重，取值为 $0 \leqslant w \leqslant 1$。通常情况下，把 w 的值设置为 0.8。

每个算法在同一台计算机上对每个算例分别运行 30 次，把得到的数据取平均值。比较实验结果如表 3。第一列写明了算例名称，第二列列出了每个算例求解得到的最优值，后面三列提供了每个算法求解得到的最优值。为了直观地比较三种算法得到的解的质量，计算了相对百分比增幅，并在最后三列给出了相应的结果。表 3 中列出的结果可以总结如下：(1) DICA 算法在给定的示例中获得了 14、13 和 12 个最优解，这远远优于其他对比算法的结果；(2) 如表中最后一行所示，DICA 平均目标值和平均百分比偏差远远低于其他算法。实验结果表明，与其他最近提出的算法相比，所提出的 DICA 算法能够更有效的解决 RCHFS 问题的方法。

表 3　DICA 算法与其他算法最好解比较

Ins	Best	fitness			dev		
		DICA	DABC	ICA-G	DICA	DABC	ICA-G
Inst 1	1899.4	**1899.4**	2093.68	2208.57	**0.00**	10.23	16.28
Inst 2	6860.35	7107.37	7046.76	**6860.35**	3.60	2.72	**0.00**
Inst 3	7051.28	**7051.28**	7226.92	7416.97	**0.00**	2.49	5.19
Inst 4	13933	**13933**	14375	14384	**0.00**	3.17	3.24
Inst 5	13337	**13337**	13915.4	13854.7	**0.00**	4.34	3.88
Inst 6	3684.24	**3684.24**	3704.23	3816.05	**0.00**	0.54	3.58
Inst 7	15248.2	15391	**15248.2**	15706.2	0.94	**0.00**	3.00
Inst 8	15986.7	**15986.7**	16645.4	17440	**0.00**	4.12	9.09
Inst 9	17010.9	**17010.9**	17533.5	17668.3	**0.00**	3.07	3.86
Inst 10	18494.1	**18494.1**	19063	19353.9	**0.00**	3.08	4.65
Inst 11	**8094.01**	8094.01	8459.09	8292.61	**0.00**	4.51	2.45
Inst 12	19872.6	20055.8	19981	**19872.6**	0.92	0.55	**0.00**
Inst 13	30527.1	**30527.1**	31421.8	31938.8	**0.00**	2.93	4.62
Inst 14	38523.5	38840.9	**38523.5**	38767	0.82	**0.00**	0.63
Inst 15	42384.1	**42384.1**	43727.3	44299.1	**0.00**	3.17	4.52
Inst 16	17814.1	17824.6	**17814.1**	18198.6	0.06	**0.00**	2.16
Inst 17	18624.5	**18624.5**	19176.6	19648.7	**0.00**	2.96	5.50
Inst 18	22079.6	**22079.6**	22622.7	23076.5	**0.00**	2.46	4.52
Inst 19	40177.3	40580.1	**40177.3**	41909.6	1.00	**0.00**	4.31
Inst 20	28379.4	**28379.4**	28981	29350.7	**0.00**	2.12	3.42
Mean		19064.26	19386.82	19703.16	0.37	2.62	4.25

4 总结与展望

本文提出了 DICA&SA 方法来解决 ERCHFS 问题。主要内容如下:首先针对经典的 HFS 问题,考虑了资源受限约束和能耗目标,提炼出了 ERCHFS 问题,建立了相应的数学规划模型。其次是算法求解,实现了离散的帝国竞争算法对问题进行求解,结合了 SA 增强了算法的全局搜索能力。最后是算法的实验分析,根据经典的 HFS 问题 Benchmark 算例,生成了与问题相符合的 RCHFS 问题算例。通过算法的实验比较与分析,验证了所提出的 DICA&SA 算法的有效性。

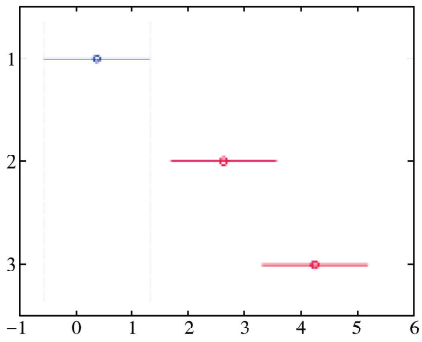


图 9 算法最优解的比较

4.1 论文主要工作和创新点

随着经济全球化以及现代工业生产技术的发展,迫使制造企业不得不寻找高质量的调度方案来减少生产成本。另外,国家对环境以及绿色制造的重视,也使得降低制造能耗成为智能制造中的一个关键问题。因此,针对 RCHFS 调度问题,建立了相应的数学规划模型。设计了离散帝国竞争算法求解该问题,通过实际工业生产仿真实验,验证了 DICA 求解 RCHFS 问题的性能。主要创新点包括:

(1) 根据文献[41]Li 等人所提出的 HFS 考虑时间约束上对 HFS 问题进行扩展,提出了考虑带资源约束和能源消耗的 HFS 问题。扩展了经典 HFS 算例,验证了算法策略以及求解的有效性。基于实际生产车间生产的数据,对算法求解性能进行了验证。

(2) 根据文献[44]Qin 等人在研究考虑一种约束的情形时只以最小化最大完工时间为优 化目标。针对 RCHFS 问题,考虑了围绕带完工时间和总能耗为性能目标的 ERCHFS 问题,构建了数学规划模型,使 ERCHSF 问题结果更加贴近于实际生产需要。

(3) 根据文献[45] Li 等人在考虑资源约束类柔性作业车间调度问题(Flexible Job Shop Scheduling Problem,FJSP)时采用人工蜂群算法,相比针对 RCHFS 问题的求解,采用帝国竞争算法求解,设计了算法的离散化策略,改进了算法流程,提出了与编码相适应的局部搜索策略,结合了 SA 算法,提高了算法的搜索能力。同时,相比较于资源约束类 FJSP,RCHFS 问题增加了机床选择的灵活性,可以显著优化系统目标。

4.2 后续研究工作展望

目前针对优化调度问题的研究已经取得了较多的成果,但随着工业技术以及制造产业的发展,理论研究与实际的工业生产还有一定的差距。针对资源受限的混合流水车间建立了数学模型,并且设计了一种有效的解决算法,取得了较好的结果。但目前研究还处在初级阶段,未来的工作还需要更深入的研究,可以从以下几个方面入手:一方面,在问题层面本文只考虑了混合流水车间中的资源约束,但实际的生产车间生产情况必定更为复杂,肯定还会存在其它各类工艺约束。如何在 HFS 问题中融合新的工艺约束,考虑多约束的混合流水车间问题,是未来研究的方向之一;此外,分布式生产是近年来生产的一种新型模式,围绕 HFS 问题的分布式生产调度及其求解方法策略等也亟待更多关注。另一方面,在算法层面混合算法在求解方面表现出了较为优越的性能,研究算法的混合策略、参数调整、平衡算法的局部和全局搜索能力等理论成果,还需要进一步的深入研究;另一方面,深度学习算法和大数据分析技术已经成为研究的热门方向,如何将帝国竞争算法与深度学习算法相结合,也将是我们未来研究的方向之一。

参 考 文 献

[1] RUIZ R,VAZQUEZ JA.The hybrid flow shop scheduling problem[J].Eur J Oper Res,2010,205(1):1-18.

[2] TANG LX,XUAN H.Lagrangian relaxation algorithms for realtime hybrid flowshop scheduling with finite intermediate buffers[J].J Oper Res Soc,2006,57(3):316-324.

[3] TANG L,XUAN H,LIU J.A new Lagrangian relaxation algorithm for hybrid flowshop scheduling to minimize total weighted completion

time[J].Comput Oper Res,2006,33(11):3344-3359.

[4] PORTMANN MC,VIGNIER A,DARDILHAC D,et al.Branch and bound crossed with GA to solve hybrid flowshops[J].Eur J Oper Res,1998,107(2):389-400.

[5] FATTAHI P,HOSSEINI MH,JOLAI F,et al.A branch and bound algorithm for hybrid flow shop scheduling problem with setup time and assembly operations[J].Appl Math Model,2014,38(1):119-134.

[6] WANG S,LIU M,CHU C.A branch-and-bound algorithm for two-stage no-wait hybrid flow-shop scheduling[J].Int J Prod Res,2015,53(4):1143-1167.

[7] ZHANG W,YIN C,LIU J,et al.Multi-job lot streaming to minimize the mean completion time in m-1 hybrid flowshops[J].Int J Prod Econ,2005,96(2):189-200.

[8] BEHNAMIAN J,GHOMI SF,ZANDIEH M.A multi-phase covering Pareto-optimal front method to multi-objective scheduling in a realistic hybrid flowshop using a hybrid metaheuristic[J].Expert Syst Appl,2009,36(8):11057-11069.

[9] DUGARDIN F,YALAOUI F,AMODEO L.New multi-objective method to solve reentrant hybrid flow shop scheduling problem[J].Eur J Oper Res,2010,203(1):22-31.

[10] ELMI A,TOPALOGLU S.Scheduling multiple parts in hybrid flow shop robotic cells served by a single robot[J].Int J Comput Integr Manuf,2014,27(12):1144-1159.

[11] ELMI A,TOPALOGLU S.A scheduling problem in blocking hybrid flow shop robotic cells with multiple robots[J].Comput Oper Res,2013,40(10):2543-2555.

[12] FIGIELSKA E.A heuristic for scheduling in a two-stage hybrid flowshop with renewable resources shared among the stages[J].Eur J Oper Res,2014,236(2):433-444.

[13] MARICHELVAM MK,PRABAHARAN T,YANG XS.A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems[J].IEEE Trans Evol Comput,2013,18(2):301-305.

[14] NADERI B,YAZDANI M.A model and imperialist competitive algorithm for hybrid flow shops with sublots and setup times[J].J Manuf Syst,2014,33(4):647-653.

[15] SONG MX,Li JQ,HAN YQ,et al.Metaheuristics for solving the vehicle routing problem with the time windows and energy consumption in cold chain logistics[J].Appl Soft Comput,https://doi.org/10.1016/j.asoc.2020.106561.

[16] Li JQ,PAN QK,MAO K.A hybrid fruit fly optimization algorithm for the realistic hybrid flowshop rescheduling problem in steelmaking systems[J].IEEE Trans Autom Sci Eng,2015,13(2):932-949.

[17] KHARE A,AGRAWAL S.Scheduling hybrid flowshop with sequence-dependent setup times and due windows to minimize total weighted earliness and tardiness[J].Comput Ind Eng,2019,135:780-792.

[18] NISHI T,KONISHI M,HASEBE S.A decentralized scheduling method for flowshop problems with resource constraints[J].Electr Eng Jpn,2004,149(1):44-51.

[19] SURAL H,KONDAKCIS,ERKIP N.Scheduling unit-time tasks in renewable resource constrained flowshops[J].Zeitschrift für Oper Res,1992,36(6):497-516.

[20] PEI J,LIU X,FAN W,et al.A hybrid BA-VNS algorithm for coordinated serial-batching scheduling with deteriorating jobs,financial budget, and resource constraint in multiple manufacturers[J].Omega,2019,82:55-69.

[21] LI J,PAN Q,DUAN P.An improved artificial bee colony algorithm for solving hybrid flexible flowshop with dynamic operation skipping [J].IEEE Trans Cybernet, 2016,46(6):1311-1324.

[22] LI J,HAN Y,DUAN PY,et al.Meta-heuristic algorithm for solving vehicle routing problems with time windows and synchronized visit constraints in prefabricated systems[J].J Cleaner Prod,2020,https://doi.org/10.1016/j.jclepro.2019.11. 9464.

[23] LI J,LIU Z,LI C,et al.Improved artificial immune system algorithm for Type-2 fuzzy flexible job shop scheduling problem[J].IEEE Trans Fuzzy Syst,2020,https://doi.org/10.1109/TFUZZ. 3016225.

[24] CHENG T,LIN B,HUANG H.Resource-constrained flowshop scheduling with separate resource recycling operations[J].Comput Oper Res,2012,39(6):1206-1212.

[25] LEU S, HWANG S.GA-based resource-constrained flow-shop scheduling model for mixed precast production[J].Autom Constr,2002,11(4):439-452.

[26] GAO K,HUANG Y,SADOLLAH A,et al.A review of energyefficient scheduling in intelligent production systems. Complex Intell Syst,2020,6:237-249.

[27] NGUYEN S,MEI Y,ZHANG M.Genetic programming for production scheduling: a survey with a unified framework[J].Complex Intell Syst,2017,3:41-66.

[28] DAI M,TANG D,GIRET A,et al.Energy-efficient scheduling for a flexible flow shop using an improved geneticsimulated annealing algorithm[J].Robot Comput Integr Manuf,2013,29(5):418-429.

[29] JIANG ED,WANG L.An improved multi-objective evolutionary algorithm based on decomposition for energy-efficient permutation flow shop scheduling problem with sequence-dependent setup time[J].Int J Prod Res,2019,57(6):1756-1771.

[30] LI JQ,PAN QK,DUAN PY,et al.Solving multi-area environmental/economic dispatch by a Pareto-based chemical-reaction optimization algorithm[J].IEEE/CAA J Autom Sin,2019,6(5):1240-1250.

[31] LEI D,GAO L,ZHENG Y.A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop[J].IEEE Trans Eng Manage,2017,65(2):330-340.

[32] DING JY,SONG S,WU C.Carbon-efficient scheduling of flow shops by multi-objective optimization[J].Eur J Oper Res ,2016,248(3):758-771.

[33] LI JQ,DU Y,GAO KZ,et al.A hybrid iterated greedy algorithm for a crane transportation flexible job shop problem[J].IEEE Trans Autom Sci Eng,2020,(In press).

[34] LI JQ,TAO XR,JIA BX,et al.Efficient multi-objective algorithm for the lot-streaming hybrid flowshop with variable sub-lots[J].Swarm Evolution Comput,2019, https://doi.org/10.1016/j.swevo.2019.100600.

[35] LI JQ,DENG JW,LI CY,et al.An improved jaya algorithm for solving the flexible job shop scheduling problem with transportation and setup times[J].KnowlBased Syst,2020, https://doi.org/10.1016/j.knosys.2020.106032.

[36] BRUZZONE AA,ANGHINILFI D,PAOLUCCI M,et al.Energyaware scheduling for improving manufacturing process sustainability:a mathematical model for flexible flow shops[J].CIRP Ann,2012, 61(1):459-462.

[37] ZHANG H,ZHAO F,FANG K,et al.Energy-conscious flow shop scheduling under time-of-use electricity tariffs[J].CIRP Ann ,2014,63(1):37-40.

[38] FAZLI K A,WANG Y.Energy-cost-aware flow shop scheduling considering intermittent renewables,energy storage,and real-time electricity pricing[J].Int J Energy Res,2018,42(12):3928-3942.

[39] LU C,GAO L,LI X,et al.Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm[J].J Clean Prod,2017,144:228-238.

[40] HUANG RH,YU SC,CHEN PH. Energy-saving scheduling in a flexible flow shop using a hybrid genetic algorithm[J].Journal of Environmental Protection,2017,8(10):1037.

[41] 李俊青,李文涵,陶昕瑞等.时间约束混合流水车间调度问题综述[J].控制理论与应用,2020,37(11):2273-2290.

[42] LI J,DUAN P,SANG H,et al.An efficient optimization algorithm for resource-constrained steelmaking scheduling problems[J].IEEE Access,2018,99:1-1.

[43] GOLDANSAZ SM,JOLAI F,ZAHEDI A AH.A hybrid imperialist competitive algorithm for minimizing makespan in a multi-processor open shop[J].Applied Mathematical Modelling,2013,37(23):9603-9616.

[44] 秦浩翔,韩玉艳,陈庆达,等.求解阻塞混合流水车间调度的双层变异迭代贪婪算法[J/OL].控制与决策,[2021-08-25].

[45] 李俊青,杜宇,田杰,等.带运输资源约束柔性作业车间调度问题的人工蜂群算法[J].电子学报,2021,49(2):324-330.