

# 混合布谷鸟算法求解绿色流水车间调度问题

钟祯充<sup>1</sup> 钱 斌<sup>1</sup> 胡 蓉<sup>1</sup> 王 凌<sup>2</sup>

1.昆明理工大学信息工程与自动化学院,昆明,650500

2.清华大学自动化系,北京,100084

**摘要:**为协同考虑经济因素和环境因素,求解了优化目标为最小化最长完工时间和碳排放总量的多目标置换流水线车间调度问题(MOPFSP)。提出了一种混合布谷鸟算法(HCS)求解 2 台机器以上的 MOPFSP 问题。采用 LOV 规则将 HCS 算法中的个体从实数向量转换成工件排序,使其可在 MOPFSP 的解空间中进行搜索;设计了一种自适应步长控制因子,用于控制算法进化阶段的搜索范围;提出一种多邻域局部搜索,用于对 HCS 算法全局搜索发现的优质解区域进行细致搜索。由于融合了基于布谷鸟算法的全局搜索和多邻域局部搜索,故 HCS 算法可有效求解 MOPFSP。仿真实验和算法对比验证了 HCS 算法求解 MOPFSP 的有效性。

**关键词:**多目标置换流水线;混合布谷鸟算法;低碳;绿色调度

**中图分类号:**TP182; TH186

DOI:10.3969/j.issn.1004-132X.2018.22.004

开放科学(资源服务)标识码(OSID):



## HCS Algorithm for Multi-objective Flow Shop Scheduling Problems with Energy Consumption

ZHONG Lingchong<sup>1</sup> QIAN Bin<sup>1</sup> HU Rong<sup>1</sup> WANG Ling<sup>2</sup>

1.Faculty of Information Engineering and Automation,

Kunming University of Science and Technology, Kunming, 650500

2.Department of Automation, Tsinghua University, Beijing,100084

**Abstract:** To consider the economic and environmental factors at the same time, this paper dealt with the multi-objective permutation flow shop scheduling problems (MOPFSP) which minimized make-span and total carbon emissions. MOPFSP was proved to be a NP-hard problem for more than two machines. A HCS algorithm was proposed to solve the problems. Firstly, a largest-order-value rule was utilized to transform HCS's individuals from real vectors to job permutations so that HCS might be used to perform search in MOPFSP's solution spaces. Secondly, an adaptive factor of step size was designed to control the search scopes in the evolution phases. Thirdly, a multi-neighborhood local search was presented to exploit the excellent subregions obtained by HCS's global search. Due to the hybridization of CS-based global search and multi-neighborhood local search, MOPFSP may be solved efficiently. Simulations and comparisons verify the efficiency of HCS to solve MOPFSP.

**Key words:** multi-objective permutation flow shop; hybrid cuckoo search(HCS) algorithm; carbon efficiency; green dispatch

### 0 引言

全球变暖是由温室气体的过度排放引起的,特别是二氧化碳(CO<sub>2</sub>)的过度排放。其中,化石燃料的燃烧成为 CO<sub>2</sub> 排放的主要原因。合理减排能够减缓地球变暖。本文研究的问题模型包含环境指标,符合当前生态环境治理的需求。根据

FANG 等<sup>[1]</sup>的调查,约一半的世界能源消耗来自工业部门,制造企业已经成为全球变暖的主要因素之一。一方面,法律法规对温室气体的排放量控制使得制造企业不得不限制碳排放;另一方面,高碳排放量带来的高税收也使得制造企业寻求切实可行的方法来减少能源消耗。

在传统生产调度问题上,主要考虑的优化指标或目标均与时间、成本和质量相关。当制造业面对日益增强的环境和节能压力时,便需要在生产制造过程中协同考虑经济指标和绿色指标。本文所研究的带绿色指标的多目标置换流水线生产

收稿日期:2017-12-18

基金项目:国家自然科学基金资助项目(51665025);国家杰出青年科学基金资助项目(61525304);云南省自然科学基金资助项目(2015FB136)

调度问题(multi-objective permutation flow shop problem, MOPFSP)具有较强的工业背景。在计算复杂度上,2 台机器以上的置换流水车间调度问题(permutation flow shop scheduling problem, PFSP)被证明是 NP-hard 问题<sup>[2]</sup>,因此,更为复杂的 MOPFSP 也属于 NP-hard 问题。综上,开展带绿色指标的 MOPFSP 求解具有重要的工程和学术意义。

在过去几十年里,流水线车间调度问题已被广泛研究,然而,近十年来才有一些文献同时考虑经济和环境指标。LUO 等<sup>[3]</sup>针对带电力消耗的多目标混合流水车间调度问题,设计了蚁群优化算法进行求解。DING 等<sup>[4]</sup>针对带碳排放总量的多目标流水车间调度问题进行研究,并提出了一种改进的迭代贪心算法求解该问题。LIU 等<sup>[5]</sup>设计了一种自适应多目标遗传算法,可有效求解一类带碳排放和总加权延迟两个优化指标的流水线车间调度问题。DING 等<sup>[6]</sup>提出了基于非支配解结构特性的迭代贪心算法,用于求解带碳排放总量的双目标流水车间调度问题。TANG 等<sup>[7]</sup>提出了改进的粒子群优化算法,可求解基于能源消耗和最长完工时间两个目标的柔性流水车间调度问题。LU 等<sup>[8]</sup>针对带能源消耗的多目标流水车间调度问题,提出了一种基于机器设置时间和工件运输时间的混合多目标回溯搜索算法进行求解。综上所述,带绿色指标的多目标流水车间调度问题的研究仍较为有限,迫切需要加强对求解该类重要问题的有效方法的研究。

布谷鸟搜索(cuckoo search, CS)算法是 YANG 和 DEB 于 2009 年提出的一种元启发式算法,该算法根据布谷鸟产卵时的飞行机制进行搜索,能够快速有效地求解连续优化问题。近年来,CS 算法也被扩展用于求解带经济指标的单目标生产调度问题。LI 等<sup>[9]</sup>利用 NEH 启发式规则产生部分初始种群,并设计了一种带局部搜索的混合 CS 算法对单目标 PFSP 问题进行求解。MAEICHELVAM 等<sup>[10]</sup>在 CS 算法的初始阶段采用 NEH 规则初始化部分种群,进而将其用于求解单目标多阶段混合流水线车间调度问题。ALAA 等<sup>[11]</sup>对 CS 算法中的莱维飞行公式进行改进,同时加强了对种群最优个体邻域的搜索,并将所提算法用于求解单目标柔性作业车间调度问题。WANG 等<sup>[12]</sup>采用 NEH 规则产生部分初始种群,继而提出了一种带局部搜索的混合布谷鸟算法,用于求解单目标流水线车间调度问题。由上述文献可知,尚无利用 CS 算法求解带绿色指

标的 MOPFSP 的相关研究。此外,上述文献均将步长控制因子设定为某个常数,并未对其进行动态调整。实际上,随着算法迭代次数的增加,若步长控制因子设置过大,布谷鸟算法的搜索易偏离优质解区域,从而导致性能变差;若步长控制因子设置过小,该算法在前期就容易陷入局部最优而早熟。因此,针对绿色流水车间调度问题,设计可动态调整步长控制因子的布谷鸟算法进行有效求解,具有重要意义。

本文提出了一种混合布谷鸟(HCS)算法用于求解优化目标为最长完工时间和总碳排放量的 MOPFSP。在 HCS 算法中,不仅提出了一种步长控制因子的自适应调整策略,使布谷鸟算法具有较快的收敛速度和较好的全局搜索性能;同时还设计了一种多邻域局部搜索机制,可对 HCS 算法全局搜索得到的优质解区域进行细致搜索,进而使算法在全局和局部搜索之间到达较好平衡。最后,通过仿真实验和算法对比来验证 HCS 算法的有效性。

## 1 低碳 MOPFSP 描述

设有  $n$  个工件需要在  $m$  台机器上加工,每个机器有  $s$  种不同的加工速度,其中工件集合为  $J = \{1, 2, \dots, n\}$ ,机器集合为  $M = \{1, 2, \dots, m\}$ ,速度集合为  $S = \{v_1, v_2, \dots, v_s\}$ 。 $P_{i,k}$  表示工件  $i$  在机器  $k$  上的标准加工时间; $V_{i,k}$  表示工件  $i$  在机器  $k$  上的加工速度; $P_{i,k}/V_{i,k}$  表示工件  $i$  在机器  $k$  上以速度  $V_{i,k}$  加工时的真实加工时间; $Q_{k,v}$  表示机器  $k$  以速度  $v$  工作时的单位能耗; $Q_k$  表示机器  $k$  待机状态时的单位能耗;当  $t$  时刻机器  $k$  以速度  $v$  工作时  $x_{kv}(t) = 1$ ,其他时刻  $x_{kv}(t) = 0$ ;  $t$  时刻机器  $k$  呈现待机状态  $y_k(t) = 1$ ,其他时刻  $y_k(t) = 0$ ;  $\pi = (j_1, j_2, \dots, j_n)$  为所有工件的某种排序; $\Pi$  是不同排序  $\pi$  的集合; $C(j_i, k) i = (1, 2, \dots, n)$  为工件  $j_i$  在机器  $k$  上的完成时间。

PFSP 的 2 个加工规则:①当工件  $i \in J$  在机器  $(k-1) \in M$  上加工完成之后,工件  $j \in J$  才能在机器  $k \in M$  上加工。②每台机器每次只能加工一个工件,不能同时加工多个工件。

假设工件  $j_1$  至  $j_n$  依次在机器 1 至机器  $m$  上顺序加工,最长完工时间 make-span 的数学模型如下:

$$C(j_1, 1) = P_{j_1,1}/V_{j_1,1} \quad (1)$$

$$C(j_i, 1) = C(j_{i-1}, 1) + P_{j_i,1}/V_{j_i,1} \quad (2)$$

$$C(j_1, k) = C(j_1, k-1) + P_{j_1,k}/V_{j_1,k} \quad (3)$$

$$C(j_i, k) = \max\{C(j_{i-1}, k), C(j_i, k-1)\} + P_{j_i,k}/V_{j_i,k} \quad (4)$$

$$C_{\max} = C(j_n, m) \quad (5)$$

其中,  $k=2,3,\cdots,m, i=2,3,\cdots,n$ ; 式(5)为最长完工时间。

假设  $\epsilon$  是每单位能耗的  $\text{CO}_2$  排放量,  $\text{CO}_2$  排放总量计算公式为

$$C_{\text{CO}_2} = \epsilon C_T = \epsilon \int_0^{C_{\text{max}}} \left( \sum_{v=1}^s \sum_{k=1}^m Q_{k,v} x_{kv}(t) + \sum_{k=1}^m Q_{k,y_k}(t) \right) dt \tag{6}$$

其中,  $C_T$  表示能源总消耗;  $C_{\text{CO}_2}$  表示  $\text{CO}_2$  排放总量, 以下简称碳排放总量 (total carbon emissions, TCE)。

为了计算碳排放总量, 需要构建机器的速率矩阵  $\mathbf{A}_{n \times m}$ , 其中  $A_{i,k} \in \{1,2,\cdots,s\}$ 。假设  $s=3, A_{i,k} \in \{1,2,3\}, n=3, m=2$ , 则矩阵  $\mathbf{A}_{3 \times 2}$  的其中一种情况表示如下:

$$\mathbf{A}_{3 \times 2} = \begin{bmatrix} 1 & 2 \\ 2 & 2 \\ 3 & 1 \end{bmatrix} \tag{7}$$

如,  $A_{31}$  表示第 3 个工件在第 1 台机器上的加工速度为 3。

对于低碳多目标 PFSP, 不可能同时求得最长完工时间 (make-span) 和 TCE 的全局最优解, 但是可以求得这样一组解: 任何一个目标的继续优化都必须牺牲其他目标函数值的有效解<sup>[13-14]</sup>。这类有效解组成的集合称之为非劣解, 即 Pareto 解集。

2 多目标问题相关概念

本文中 MOPFSP 可以表示为

$$\min \mathbf{F}(\pi) = [f_1(\pi), f_2(\pi)], \pi \in \psi \tag{8}$$

$$f_1 = C_{\text{max}} \tag{9}$$

$$f_2 = C_{\text{CO}_2} \tag{10}$$

式中,  $\psi$  为可行域。

(1) 支配。对于目标向量  $\mathbf{F}(\pi_1) = (f_1(\pi_1), f_2(\pi_1))$  和  $\mathbf{F}(\pi_2) = (f_1(\pi_2), f_2(\pi_2))$ , 当且仅当  $(\forall i \in \{1,2\}: f_i(\pi_1) \leq f_i(\pi_2)) \cap (\exists i \in \{1,2\}: f_i(\pi_1) < f_i(\pi_2))$  时, 称  $\mathbf{F}(\pi_1)$  支配  $\mathbf{F}(\pi_2)$ , 记为  $\mathbf{F}(\pi_1) < \mathbf{F}(\pi_2)$ 。

(2) Pareto 优解。对于  $\pi \in \psi$ , 当且仅当不存在  $\pi' \in \psi$  使  $\mathbf{F}(\pi') < \mathbf{F}(\pi)$  时, 称  $\pi$  为 Pareto 优解。

(3) Pareto 解集合。对于 Pareto 解集合  $P$ :

$$P = \{\pi \in \psi \mid \nexists \pi' \in \psi: \mathbf{F}(\pi') < \mathbf{F}(\pi)\}$$

(4) Pareto 前沿。

对于 Pareto 前沿 PF, 有

$$P_F = \{\mathbf{F}(\pi) = (f_1(\pi), f_2(\pi)) \mid \pi \in P\}$$

3 混合布谷鸟算法

3.1 标准布谷鸟算法

YANG 等<sup>[15]</sup> 根据布谷鸟寻窝产蛋的行为和莱维飞行 (Lévy flight) 特征, 提出了标准的多目标布谷鸟算法。布谷鸟算法中个体更新方式有两种, 第一种是使用莱维飞行公式:

$$\mathbf{x}_{i,t+1} = \mathbf{x}_{i,t} + \alpha \oplus \text{Lévy}(\lambda) \tag{11}$$

其中,  $\mathbf{x}_{i,t+1}, \mathbf{x}_{i,t}$  分别表示第  $i$  个个体在第  $k$  代和第  $k+1$  代时的位置向量;  $\oplus$  表示点乘;  $\alpha$  为控制步长的步长因子, 大多数情况下,  $\alpha = O(1)$ ;  $\text{Lévy}(\lambda)$  为莱维飞行搜索路径。

从式(11)可以看出, 该分布使布谷鸟的连续位置形成了一种带重尾的概率分布, 能扩大搜索范围, 增加种群多样性, 且容易跳出局部最优。

第二种更新方式是根据一个固定的发现概率  $P_a$  与一个随机数  $\beta$  之间的关系确定是否产生新个体, 更新公式如下:

$$\mathbf{x}_{i,t+1} = \mathbf{x}_{i,t} + \gamma H(P_a - \beta) \oplus [\mathbf{x}_{o,t} - \mathbf{x}_{k,t}] \tag{12}$$

其中,  $\gamma$  和  $\beta$  二者均服从均匀分布,  $\gamma, \beta \in U[0,1]$ ;  $\mathbf{x}_{i,t}, \mathbf{x}_{o,t}, \mathbf{x}_{k,t}$  分别为第  $t$  代中 3 个不同的随机个体;  $H$  为赫威赛德函数, 其计算公式为

$$H(P_a - \beta) = \begin{cases} 0 & P_a > \beta \\ 0.5 & P_a = \beta \\ 1 & P_a < \beta \end{cases} \tag{13}$$

3.2 混合布谷鸟算法 (HCS)

3.2.1 自适应步长因子

CS 算法虽然在诸多领域得到应用研究, 但其本身存在固有不足: 莱维飞行是一种马尔科夫链, 只与当前情况有关, 随机性较大, 所以标准 CS 算法缺乏有效机制来加强搜索深度, 算法收敛精度不高。CS 算法在式(11)中定义了步长控制因子, 该因子在标准算法中一般设定为固定的常数 (譬如常取值为 0.01)。若步长控制因子取值过大, 易导致算法后期的搜索偏离优质解, 使其收敛速度变慢; 反之, 若步长控制因子取值过小, 则算法可能过早地陷入局部最优解, 从而导致算法性能较弱。因此, 对步长控制因子的改进有利于算法性能的提升。如果在算法搜索前期使用一个较大的步长控制因子, 有利于在全局范围内迅速发现优质解所在区域; 同时, 随着算法搜索的推进, 应逐渐减小步长控制因子, 加强对局部优质解区域的细致搜索, 这有利于提高算法的收敛速度和性能。

本文从步长控制因子方面对标准 CS 算法进行改进: 用动态步长控制因子替换原有固定的步长控制因子。寻优过程中, 随着个体质量逐步提高, 适当缩小搜索范围, 以加强搜索深度, 有利于



搜寻到更优的解。合理的步长控制因子应该是随着进化代数的增加而逐渐减小,使得算法在进化后期容易发现优质个体。

本文根据以下方面自适应调整步长控制因子 $\alpha$ :将 $\alpha$ 取值范围设置为 $[0.01, 0.2]$ ;另外,引入余弦函数使 $\alpha$ 随着进化代数的增加而减小。综上所述,提出 $\alpha$ 的改进公式:

$$\alpha^{(k+1)} = \begin{cases} \alpha_{\max} \cos(\frac{\pi}{2} \frac{k}{T_{\max}}) & R \leq 0.2 \\ \alpha^{(k)} & 0.2 < R < 0.5 \\ \alpha_{\min} & 0.5 \leq R \end{cases} \quad (14)$$

其中, $R$ 表示当前进化代数与总进化代数之比; $\alpha_{\min}$ 为步长控制因子的下限; $\alpha_{\max}$ 为步长控制因子的上限; $T_{\max}$ 为最大迭代次数; $k$ 为当前进化代数。算法初始阶段 $R \leq 0.2$ ,此时应有大步长去发现优质解所在区域,因此步长控制因子 $\alpha$ 随进化代数增加而逐渐减小;算法中期可能达到最佳更新状态,即 $0.5 > R > 0.2$ ,此时应在优质解所在区域进一步搜索,加强局部精细搜索, $\alpha$ 保持不变;算法后期 $R \geq 0.5$ ,此时个体逐渐接近 Pareto 前沿,无需大步长跳跃,因此保留 $\alpha$ 下界即可。

### 3.2.2 多邻域局部搜索

为进一步提高布谷鸟算法的局部搜索能力,本文引入多邻域局部搜索策略,对种群中的优质个体执行基于不同邻域的细致搜索。具体来说,就是对算法当前的非劣解集中的个体执行基于三种邻域的局部搜索。这三种邻域搜索分别为:Interchange local search, Insert local search<sup>[16]</sup>、2-opt local search,具体定义如下。

Interchange local search:对每个个体的工件排序,随机选择其中两个不同的位置,交换位置上的工件。例如 10 工件排序为 $[4, 2, 7, 1, 3, 5, 9, 8, 10, 6]$ ,随机产生了两个位置 $p_1=3, p_2=9$ ,则将位置 3 的工件 7 和位置 9 的工件 10 交换位置,得到一个新排序 $[4, 2, 10, 1, 3, 5, 9, 8, 7, 6]$ 。

Insert local search:该步骤可分为前插入和后插入。对每个个体的工件排序进行操作,随机选择其中 2 个不同的位置 $p_1$ 和 $p_2$ ,假设 $p_1 > p_2$ 。后插入是指将位置 $p_1$ 的工件插入位置 $p_2$ ,位置 $p_1+1 \sim p_2$ 的工件均往前挪一个位置;前插入是指将 $p_2$ 的工件插入位置 $p_1$ ,位置 $p_1 \sim p_2-1$ 的工件均往后挪一个位置。例如 10 工件排序为 $[4, 2, 7, 1, 3, 5, 9, 8, 10, 6]$ ,随机产生了两个位置 $p_1=3, p_2=9$ ,按照上文,后插入得到的新排序为 $[4, 2, 1, 3, 5, 9, 8, 10, 7, 6]$ ,前插入得到的新排序为 $[4, 2, 10, 7, 1, 3, 5, 9, 8, 6]$ 。

2-opt local search:对每个个体的工件排序,随机选择其中两个不同的位置 $p_1$ 和 $p_2$ ,将 $p_1 \sim p_2$ 的工件排序逆序排列,其他位置工件排序不变。例如 10 工件排序为 $[4, 2, 7, 1, 3, 5, 9, 8, 10, 6]$ ,随机产生了两个位置 $p_1=3, p_2=9$ ,按照上文,得到的新排序为 $[4, 2, 10, 8, 9, 5, 3, 1, 7, 6]$ 。

令 $\pi(X)$ 为个体 $X$ 基于 LOV 规则的工件排序, $\pi(X')$ 为个体 $X'$ 基于 LOV 规则的工件排序, $k$ 为扰动或探索次数。对个体 $X$ 执行多邻域局部搜索的具体步骤如下:

(1)扰动阶段。①设 $k=0$ ;②随机选择 2 个不同位置 $p_1$ 和 $p_2$ , $\pi(X) = \text{Insert}(\pi(X), p_1, p_2)$ , $k=k+1$ ;③如果 $k < 2$ ,则返回步骤②。

(2)探索阶段。①设 $k=0, t=0$ ;②随机选择两个不同位置 $p_1$ 和 $p_2$ ;如果 $t=0$ , $\pi(X') = \text{Insert}(\pi(X), p_1, p_2)$ ;如果 $t=1$ , $\pi(X') = \text{Interchange}(\pi(X), p_1, p_2)$ ;如果 $t=2$ , $\pi(X') = 2\text{-opt}(X, p_1, p_2)$ ;③如果 $\pi(X') < \pi(X)$ ,则 $\pi(X) = \pi(X')$ , $k=k+1$ ,否则 $t=t+1$ ;④如果 $k < 30$ ,则跳到步骤⑤,否则停止探索并输出 $\pi(X)$ 和 $X$ ;⑤如果 $t < 3$ ,返回步骤②,否则 $t=0$ ,返回步骤②。

### 3.3 HCS 算法求解 MOPFSP 的步骤

基于改进的 HCS 算法求解 MOPFSP 的主要步骤如下:

(1)参数初始化。设置种群规模 $N$ ;个体上下界,并在界内初始化种群 $W$ ;设置最大迭代次数 $g_{\text{enc}}$ 或算法运行时间 $T$ 。

(2)个体离散化。采用 LOV 规则将连续的个体转化为离散排序;计算每个个体的两个目标函数值。

(3)个体更新。随机挑选一个个体 $x_i$ ,根据式(11)采用莱维飞行对个体进行更新,产生一个新个体 $x_{i1}$ ;新老个体采用非支配原则择优,若互不支配,则随机保留一个,保留的个体存入 $x_{i1}$ 。

(4)抛弃概率的应用。根据抛弃概率判断是否对步骤(3)中保留的个体 $x_{i1}$ 进行操作,若要对其进行操作,则利用式(12)更新并得到一个新个体 $x_{i2}$ ,最后对新旧个体采用非支配原则择优;若互不支配,则随机保留一个,保留的个体存入 $x_{i1}$ 。

(5)保留 Pareto 前沿。利用非支配原则将本代的 Pareto 前沿找出,并将本代 Pareto 前沿的个体存入 Pareto 解集 $P^{(t)}$ 。

(6)多邻域搜索。将本代 $P^{(t)}$ 集合里的每个个体 $X^{P^{(t)}}$ 对应的排序依次进行多邻域搜索,得到

本代更新后的 Pareto 解集  $P^{(t)'}$ 。

(7) 记录当前 Pareto 解集。将更新后的 Pareto 解集  $P^{(t)'}$  与第  $t-1$  代保留下来的 Pareto 解集  $P$  融合,利用非支配原则求出当前第  $t$  代 Pareto 解集  $P$ ,并用此集合  $P$  代替本代非劣解集  $P^{(t)'}$ ,便于在下一代中使用。

(8) 根据式(14)更新步长控制因子  $\alpha$ , 每一代都要对步长控制因子做出判断,并更新。

(9) 终止条件若当前迭代次数小于最大迭代次数  $g_{mc}$  或进化时间小于算法运行时间  $T$ , 重复执行步骤(2)~步骤(8); 否则输出当前 Pareto 解集  $P$ , 结束算法。

基于以上步骤,利用 HCS 算法求解 MOPF-SP 的算法流程图见图 1。

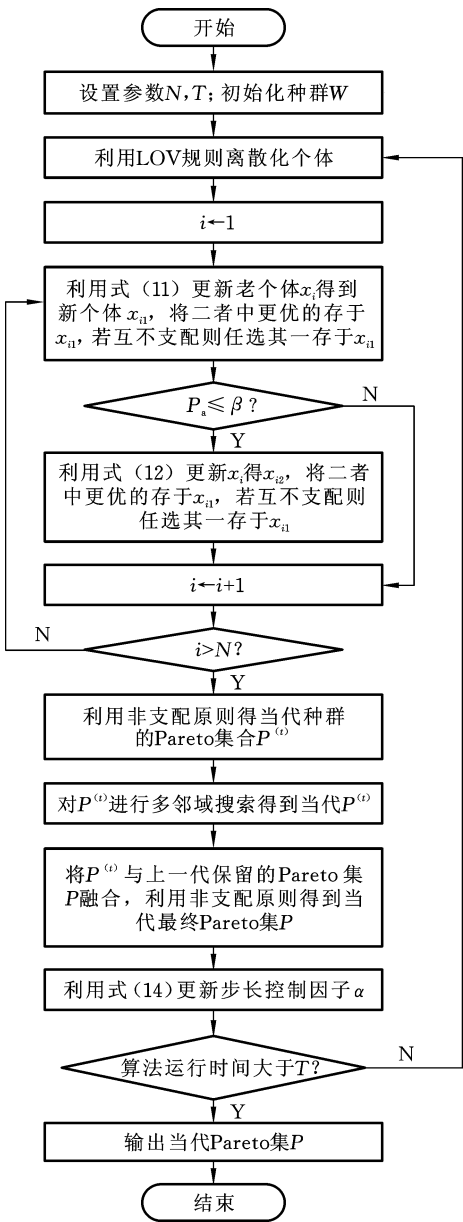


图 1 HCS 算法流程图  
Fig.1 Flow chart of HCS

4 算法测试结果及分析

为验证 HCS 算法求解 MOPFSP 的有效性, 本文选取了 10 种规模大小不同的算例, 采用标准 CS 和 INSGA-II 算法<sup>[17]</sup> 进行对比实验。INSGA-II 算法是基于经典多目标算法 NSGA-II 的改进算法, 文献[17]利用仿真实验验证了 INSGA-II 算法优于 NSGA-II 算法, 所以将 HCS 算法与 INSGA-II 算法比较是有意义的。

在测试算例中, 工件在每台机器上的加工时间采用 100 以内的随机正整数按照问题规模生成; 机器速度挡位设定为  $A_{i,k} \in \{1, 2, 3\}$ ; 所有算法的种群大小均为 30; HCS 算法与 CS 算法的抛弃概率均为 0.25; CS 算法步长控制因子为 0.01; HCS 算法初始步长控制因子为 0.2; INSGA-II 算法中变异概率为 0.3, 交叉概率为 0.9。本文以  $50n$  (单位: ms) 作为各算法运行的终止条件, 其中,  $n$  为每种问题规模的工件数。这使得测试同一个问题规模时, 所有算法的运行时间一致, 可确保比较的公平性。各算法对每一测试问题均独立运行 20 次。

所有算法和测试程序均用 Delphi 10.2 编程实现, 操作系统为 Win 10, 处理器为 Intel(R) Core(TM) i5-4210U 1.70 GHz, 内存为 4 GB。

本文采用的分析指标是文献[18]中提出的多目标分析指标, 分别为  $R\_NDS(S_r)$  和  $NDS\_NUM(S_r)$ , 计算公式如下:

$$R\_NDS(S_r) = |S_r - \{x \in S_r \mid \exists y \in S; y < x\}| / |S_r| \tag{15}$$

$$NDS\_NUM(S_r) = |S_r - \{x \in S_r \mid \exists y \in S; y < x\}| \tag{16}$$

其中,  $S_r$  是指算法  $r$  的 Pareto 解集;  $S$  是指  $K$  种算法的 Pareto 解集的并集, 可以表示为  $S = S_1 \cup \dots \cup S_r \cup \dots \cup S_K$ ;  $y < x$  是指个体  $y$  完全支配个体  $x$ ;  $|S_r|$  是指  $S_r$  集合中个体的数量;  $NDS\_NUM(S_r)$  是指算法  $r$  中未被支配的个体数量;  $R\_NDS(S_r)$  是指算法  $r$  中未被支配的个体数占算法  $r$  中总的 Pareto 解集个体数的比率。  $R\_NDS(S_r) = 1$  意味着  $S_r$  中所有的 Pareto 个体都不被支配;  $R\_NDS(S_r) = 0.9$  意味着  $S_r$  中 90% 的 Pareto 个体都不被支配。

本文中每个问题规模的数据结果可以在表 1 和表 2 中找到; HCS 算法和 CS 算法的对比数据见表 1, HCS 算法和 INSGA-II 算法的对比数据见表 2。根据上文中  $S$  的定义, 表 1 中的  $S$  可以表示为  $S = S_{HCS} \cup S_{CS}$ , 表 2 中的  $S$  可以表示为  $S = S_{HCS} \cup S_{INSGA-II}$ 。  $R\_NDS\_HCS$  表示 20 个

$R\_NDS(S_{HCS})$ 数据的平均比率, $R\_NDS\_CS$ 表示 20 个  $R\_NDS(S_{CS})$  数据的平均比率,  $R\_NDS\_INSGA-II$ 表示 20 个  $R\_NDS(S_{INSGA-II})$ 数据的平均比率,  $NDS\_NUM\_HCS$ 表示 20 个  $NDS\_NUM(S_{HCS})$ 数据的平均数,  $NDS\_NUM\_CS$ 表示 20 个  $NDS\_NUM(S_{CS})$ 数据的平均数,  $NDS\_NUM\_INSGA-II$ 表示 20 个  $NDS\_NUM(S_{INSGA-II})$ 数据的平均数。

表 1 HCS 算法和 CS 算法对比数据

Tab.1 Statistical results of HCS and CS

问题 规模	HCS		CS	
	$R\_NDS\_HCS$	$NDS\_NUM\_HCS$	$R\_NDS\_CS$	$NDS\_NUM\_CS$
10_5	1	3	0.025	0.05
20_5	1	1	0	0
30_5	1	4	0	0
50_5	1	4.6	0	0
50_10	1	3.05	0	0
60_10	1	2.95	0	0
70_10	1	1.9	0	0
90_20	1	4.85	0	0
90_30	1	4.8	0	0
100_30	1	5.9	0	0

表 2 HCS 算法和 INSGA-II 算法对比数据

Tab. 2 Statistical results of HCS and INSGA-II

问题 规模	HCS		INSGA-II	
	$R\_NDS\_HCS$	$NDS\_NUM\_HCS$	$R\_NDS\_INSGA-II$	$NDS\_NUM\_INSGA-II$
10_5	1	3	0.96	2.85
20_5	0.8	2.6	0.35	0.25
30_5	0.987 5	3.95	0.175	0.6
50_5	0.91	3.95	0.3	1.35
50_10	0.925	2.55	0.3	0.85
60_10	0.66	1.9	0.5	1
70_10	0.8	1.6	0.575	1.1
90_20	0.82	3.5	0.379	1.15
90_30	0.837 5	4.4	0.297 5	0.85
100_30	0.762	3.55	0.59	1.7

由表 1 可看出, $R\_NDS\_HCS$  全部为 1,而  $R\_NDS\_CS$  几乎为零,说明在求解以上所有问题规模的 MOPFSP 时,HCS 算法完全支配标准 CS 算法,且  $NDS\_NUM\_HCS$  的个数在 3~6 之间,符合种群规模为 30 的情况;由表 2 可看出, $R\_NDS\_HCS$  均大于  $R\_NDS\_INSGA-II$ ,且有 8 个问题规模的  $R\_NDS\_HCS$  大于 0.8,说明 HCS 算法有 80% 以上的 Pareto 个体支配 INSGA-II 算法的 Pareto 个体,而在其他 2 个问题规模中,比率为 65% 以上。总体而言,HCS 算法得到的解集更优,证明了 HCS 算法的有效性。

10\_5 规模和 100\_30 规模某次运行情况见图

2、图 3。从图 2 中可看出,虽然 HCS 算法与 INSGA-II 算法寻得的 Pareto 解集重合,但都完全支配标准 CS 算法。从图 3 中可看出,HCS 算法的 Pareto 前沿在其他 2 个算法的左下方,说明 HCS 算法的 Pareto 解集把其他 2 个算法的 Pareto 解集完全支配。随着问题规模的增大,较标准 CS 算法和 INSGA-II 算法,HCS 算法的 Pareto 前沿与其他 2 个算法的 Pareto 前沿距离越拉越大,寻优的优越性越来越明显。

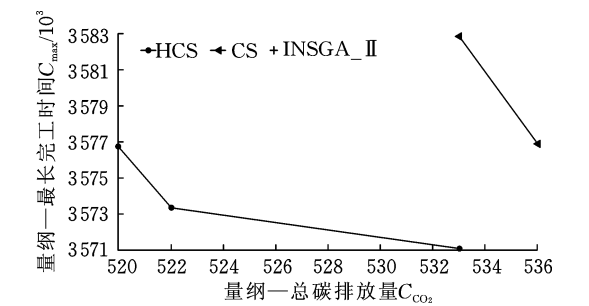


图 2 问题规模为 10\_5 时 3 种算法的某次 Pareto 解点图  
Fig. 2 Non-dominated solutions of HCS (3points)、CS (2points) and INSGA-II (3points) when the instance is 10\_5

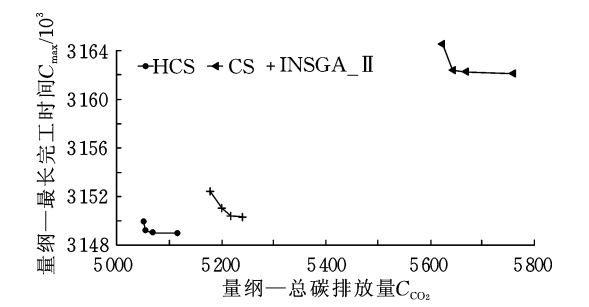


图 3 问题规模为 100\_30 时 3 种算法的某次 Pareto 解点图  
Fig. 3 Non-dominated solutions of HCS (4points)、CS (4points) and INSGA-II (4points) when the instance is 100\_30

综上所述,在以上所有问题规模中,HCS 算法在求解 MOPFSP 时比 INSGA-II 算法和 CS 算法有效。

5 实例分析

为进一步验证所提算法的有效性,将 HCS 算法用于求解江西瑞金某电线电缆厂的电线电缆生产调度问题。该工厂初成型电缆生产过程依次为单丝拉制、单丝退火、导体绞制、绝缘挤出、成缆共 5 个环节。5 个环节分别在 5 台特定的机器上加工,各环节的加工机器可通过调整挡位来设定加工速度。近年来,该公司积极响应绿色生产节能减排,在其生产过程中同时考虑经济指标 (make-span) 和环境指标 (TCE)。显然,此初成型电缆的生产调度问题是典型的 MOPFSP。目前,该电缆厂生产调度是由调度员基于经验对工

件编号后进行人工排序调度。

本文采用该工厂生产 30 类电缆的实际生产数据作为测试实例,用 HCS 算法运行 1.5 s 求解,同时请调度员在 5 min 内给出人工调度方案。

HCS 算法获得了调度方案 S1~S4,见表 3 和图 4。调度员通过经验得到的调度方案 T1,见表 3。由表 3 可知,S1~S4 均明显优于 T1。这一结果表明 HCS 算法可快速有效地求解实际问题。

表 3 调度方案  
Tab.3 Scheduling plan

调度方案	工件调度排序															C <sub>max</sub>	TCE
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		
	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30		
T1	27	26	15	8	2	25	10	4	11	7	18	14	21	28	24	1 108	9 764 903
	3	6	9	20	30	5	29	19	23	16	13	12	1	17	22		
S1	18	12	28	24	10	5	15	2	27	9	26	4	11	8	19	1 035	9 750 170
	29	16	23	25	3	30	7	21	20	1	13	6	14	17	22		
S2	18	12	27	2	24	10	5	15	9	26	21	29	23	28	11	1 039	9 749 536
	16	25	30	7	4	22	8	1	20	19	13	14	3	6	17		
S3	18	27	2	12	24	10	5	15	26	21	29	20	8	22	4	1 056	9 744 074
	11	16	25	30	9	28	23	19	1	6	14	3	17	13	7		
S4	18	28	9	26	15	5	10	24	27	12	2	4	8	19	29	1 042	9 748 265
	16	25	30	7	21	20	11	1	22	23	3	6	14	17	13		

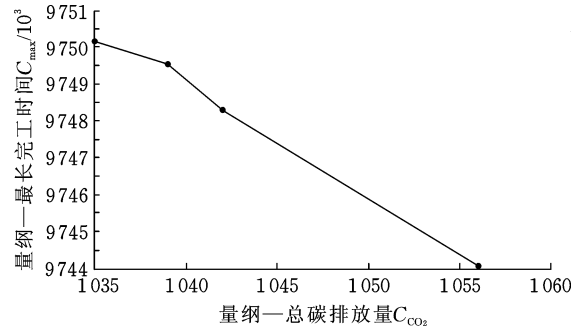


图 4 HCS 目标值对比  
Fig.4 HCS target value comparison

6 结论

本文提出了一种混合布谷鸟算法,用于求解绿色多目标流水车间调度问题(MOPFSP)。HCS 算法通过采用所提的自适应步长控制因子和多邻域局部搜索,较好地平衡了算法的全局和局部搜索,提高了算法的性能和收敛速度。仿真实验和算法比较结果表明,HCS 算法能够较快求解 MOPFSP,且其性能优于 CS 算法和 INSGA-II 算法。验证了 HCS 算法在求解 MOPFSP 上的有效性。

关于布谷鸟算法在复杂生产调度上的未来研究,可以考虑将其拓展应用于比 MOPFSP 更加复杂的调度问题上,特别是不确定绿色调度问题。

参考文献:

[1] FANG Kan, UHAN N, ZHAO Fu, et al. A New Approach to Scheduling in Manufacturing for Power Con-

sumption and Carbon Footprint Reduction[J]. Journal of Manufacturing Systems, 2011, 30: 234-240.

[2] GAREY M R, JOHNSON D S, SETHI R. The Complexity of Flowshop and Jobshop Scheduling [J]. Mathematics of Operations Research, 1976, 1 (2): 117-29.

[3] LUO Hao, DU Bing, HUANG G Q, et al. Hybrid Flow Shop Scheduling Considering Machine Electricity Consumption Cost[J]. International Journal of Production Economics, 2013, 146(2): 423-439.

[4] DING Jianya, SONG Shiji, WU Cheng, et al. Carbon-efficient Scheduling of Flow Shops by Multi-objective Optimization[J]. European Journal of Operational Research, 2016, 248(3): 758-771.

[5] LIU Cheng-Hsiang, HUANG Ding-Hsiang. Reduction of Power Consumption and Carbon Footprints by Applying Multi-objective Optimisation via Genetic Algorithms[J]. International Journal of Production Research, 2014, 52(2): 337-352.

[6] DING Jianya, SONG Shiji, WU Cheng. Carbon-efficient Scheduling of Flow Shops by Multi-objective Optimization[J]. European Journal of Operational Research, 2015, 248(3): 1-14.

[7] TANG D. Energy-efficient Dynamic Scheduling for a Flexible Flow Shop Using an Improved Particle Swarm Optimization[J]. Computers in Industry, 2015, 81(C): 82-95.

[8] LU C. Energy-efficient Permutation Flow Shop Scheduling Problem Using a Hybrid Multi-objective Backtracking Search Algorithm [J]. Journal of



Cleaner Production,2017,144:228-238.

[9] LI Xiangtao, YIN Minghao. A Hybrid Cuckoo Search via Lévy Flights for the Permutation Flow Shop Scheduling Problem[J]. International Journal of Production Research,2013,51 (16): 4732-4754.

[10] MARICHELVAMMK, PRABAHARAN T, YANG Xinshe. Improved Cuckoo Search Algorithm for Hybrid Flow Shop Scheduling[J]. Applied Soft Computing Journal,2014,19 (1): 93-101.

[11] ALAA S, ALOBAIDI A. Two Improved Cuckoo Search Algorithms for Solving the Flexible Job-shop Scheduling Problem[J]. International Journal on Perceptive and Cognitive Computing, 2016, 2 (2): 25-31.

[12] WANG H, WANG W, SUN H, et al. A New Cuckoo Search Algorithm with Hybrid Strategies for Flow Shop Scheduling Problems[J]. Soft Computing,2017,21 (15): 4297-4307.

[13] 张建军,彭亚丽,张利,等. 基于改进粒子群优化算法的多目标铜卷加工生产调度研究[J]. 中国机械工程,2011,22(17): 2061-2066.

ZHANG Jianjun, PENG Yali, ZHANG Li, et al. Improved PSO for Multi-objective Copper Roll Production Scheduling[J]. China Mechanical Engineering,2011,22(17): 2061-2066.

[14] 陈可嘉,周晓敏. 多目标置换流水车间调度的改进食物链算法[J]. 中国机械工程,2015,26(3): 348-353.

CHEN Kejia, ZHOU Xiaomin. Improved Food Chain Algorithm for Multi-objective Permutation Flow Shop Scheduling[J]. China Mechanical Engineering,2015,26(3): 348-353.

[15] YANG Xinshe, DEB S. Multiobjective Cuckoo Search for Design Optimization[J]. Computers & Operations Research,2013,40(6): 1616-1624.

[16] LEI Deming, GUO Xiuping. A Shuffled Frog-leaping Algorithm for Hybrid Flow Shop Scheduling with Two Agents [J]. Expert Systems with Applications,2015,42(23): 9333-9339.

[17] WANG Ruihua. An Improved Non-dominated Sorting Genetic Algorithm for Multi Objective Problem[J]. Mathematical Problems in Engineering, 2016,2006(103): 1-7.

[18] QIAN Bin, LI Zihui, HU Rong, et al. A Hybrid Differential Evolution Algorithm for the Multi objective Reentrant Job-shop Scheduling Problem[C]//IEEE International Conference on Control and Automation (ICCA). Hangzhou,2013:485-489.

(编辑 陈 勇)

---

作者简介:钟祯充,女,1994年生,硕士研究生。研究方向为智能算法与优化调度。钱 斌(通信作者),男,1976年生,教授、博士研究生导师。研究方向为优化调度理论与方法、智能优化方法。E-mail:bin.qian@vip.163.com

---

(上接第 2673 页)

[12] 高亮,高海兵,周驰.基于粒子群优化的开放式车间调度[J].机械工程学报,2006,42(2):129-134.

GAO Liang, GAO Haibin, ZHOU Chi. Open Shop Scheduling Based on Particle Swarm Optimization [J]. Chinese Journal of Mechanical Engineering, 2006,42(2): 129-134.

[13] SHA D Y, HSU C Y. A New Particle Swarm Optimization for the Open Shop Scheduling Problem [J]. Computers & Operations Research, 2008, 35 (10):3243-3261.

[14] NASIRI M M, YAZDANPARAST R, JOLAI F. Simulation Optimization Approach for Optimizing Jobs Dispatching Rule in an Open Shop Scheduling Problem[J]. International Journal of Computer Integrated Manufacturing,2017,30(46):1239-1252.

[15] BAI D, ZHANG Z, ZHANG Q. Flexible Open Shop Scheduling Problem to Minimize Makespan[J]. Computers & Operations Research,2015,67(1): 207-215.

[16] WANG W L, WANG H Y, ZHAO Y W, et al. Parallel Machine Scheduling with Splitting Jobs by a Hybrid Differential Evolution Algorithm[J]. Computers & Operations Research,2013,40(5):1196-1206.

[17] MATTA M E. A Genetic Algorithm for the Proportionate Multiprocessor Open Shop [J]. Computers & Operations Research,2009,36(9): 2601-2618.

[18] CHEN Y, ZHANG A, CHEN G, et al. Approximation Algorithms for Parallel Open Shop Scheduling[J]. Information Processing Letters,2013,113(7):220-224.

[19] 展勇,邱长华,祝海涛.基于网络流的多并行机可中断开放车间调度算法[J]. 计算机集成制造系统, 2011,17(5): 990-996.

ZHAN Yong, QIU Changhua, ZHU Haitao. An Interruptible Multiprocessor Open Shop Scheduling Algorithm Based on Network Flow [J]. Computer Integrated Manufacturing System,2011,17(5):990-996.

[20] BEAN J. Genetic Algorithms and Random Keys for Sequencing and Optimization[J]. Orsa Journal on Computing, 1994, 6:154-160.

(编辑 张 洋)

---

作者简介:陈正斌,女,1977年生,副教授。研究方向为生产计划与调度、制造系统建模与仿真。发表论文 11 篇。E-mail: yarongchen@126.com. 黄沈权(通信作者),男,1988年生,讲师。研究方向为服务型制造、智能制造系统。发表论文 15 篇。E-mail: hshenquan@163.com。