Computer Science and Application 计算机科学与应用, 2017, 7(12), 1262-1269
Published Online December 2017 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2017.712141

Hans汉斯

A Novel Discrete Particle Swarm Optimization Algorithm Base on Crossover Operator and Neighborhood Search

Wenxue Zhang[1], Xiaoning Wei[2], Xiaowei Wan[3]

[1]School of Sciences, Ningxia Medical University, Yinchuan Ningxia
[2]Department of Information, Affiliated Hui Medicine & Chinese Medicine Hospital to Ningxia Medical University, Wuzhong Ningxia
[3]Department of Information, The Fifth People's Hospital of Ningxia, Shizuishan Ningxia
Email: wxzhang@163.com

Received: Dec. 4th, 2017; accepted: Dec. 17th, 2017; published: Dec. 28th, 2017

Abstract

Based on crossover operator, neighborhood search and the essential mechanism of information updating in particle swarm optimization, a novel discrete particle swarm optimization (NDPSO) algorithm is proposed in which some basic operations on particles velocity and location are redefined. The NDPSO is a general-purpose optimizing model for combinatorial optimization problem; It is evaluated with 23 benchmark instances of flowshop scheduling problem and found to be more efficient and effective than existing algorithms.

Keywords

Discrete Particle Swarm Optimization, Crossover Operator, Neighborhood Search, Combinatorial Optimization, Flowshop Scheduling

基于交叉算子和邻域搜索算子的离散粒子群优化算法

张文学[1]，韦晓宁[2]，万晓伟[3]

[1]宁夏医科大学 理学院，宁夏 银川
[2]宁夏医科大学附属回医中医医院 信息科，宁夏 吴忠
[3]宁夏第五人民医院 信息科，宁夏 石嘴山
Email: wxzhang@163.com

文章引用: 张文学, 韦晓宁, 万晓伟. 基于交叉算子和邻域搜索算子的离散粒子群优化算法[J]. 计算机科学与应用, 2017, 7(12): 1262-1269. DOI: 10.12677/csa.2017.712141

收稿日期：2017年12月4日；录用日期：2017年12月17日；发布日期：2017年12月28日

摘　要

基于交叉算子、邻域搜索算子和粒子群算法信息更新的本质机理，重新定义了粒子的基本运算，包含粒子位置的减法运算，粒子速度的数乘运算、加法运算以及粒子位置与速度的加法运算；提出了一种通用的新型离散粒子群优化算法；最后，以流水车间调度问题中23个标准算例为实验数据进行了仿真实验，实验结果表明了本文所提出算法的有效性。

关键词

离散粒子群，交叉算子，邻域搜索，组合优化，流水车间调度

1. 引言

粒子群优化(Particle Swarm Optimization, PSO)是 Eberhard 和 Kennedy [1] [2]于 1995 年提出的一种基于群体智能的新型优化算法，其思想来源于鸟群寻食行为，有依赖参数少、运算简单、易于实现，求解速度快等优点。PSO 算法提出后，引起了智能计算等领域的学者们的广泛关注，并在短短的几年时间里涌现出大量的研究成果，已经被广泛的应用到各个领域，如神经网络训练[3]、调度问题[4]和模糊控制系统[5]等，成为智能计算领域的一个研究热点。

最初对 PSO 算法的研究主要集中在连续空间，即描述粒子状态及其运动规律的量都是连续的，并且在连续空间优化领域取得了巨大成功，因此许多学者试图用它去解决组合优化问题，但由于组合优化问题的离散特性，其求解效果不甚满意。因此对离散粒子群优化(Discrete Particle Swarm Optimization, DPSO)算法进行研究成为新的热点，对 PSO 算法的离散化研究可分为基于连续空间的 DPSO 和基于离散空间的 DPSO 两类[6]。1) 前者将实际离散问题映射到粒子连续运动空间后，在连续空间中计算和求解，算法生成的连续解与整数规划问题的目标函数评价值之间存在多对一的映射，导致大量冗余解空间与冗余搜索，从而影响算法的收敛速度。2) 后者则是将 PSO 算法映射到离散空间，在计算上以离散空间特有的对矢量的位操作取代传统向量计算，符合 PSO 的基本机理，不存在冗余搜索问题，且对离散问题表达自然。目前基于离散空间 DPSO 的研究主要有：Clerc 等[7]提出了离散化 PSO 算法的框架，针对 TSP 问题重新定义了 PSO 的基本运算；Wang 等[8]引入交换子与交换序列的概念，重新定义了 PSO 的基本运算；在文献[8]的基础上，Shi 等[9]针对 TSP 问题引入了置换子与置换序列的概念，重新定义了 PSO 的基本运算。可是，现有研究主要针对个别类型问题(如，Traveling Salesman Problem, knap sack problem)，没有通用的标准模型[6]。

本文在上述研究的基础上，基于交叉算子、领域搜索算子和 PSO 算法信息更新的本质机理，针对组合优化问题，在 PSO 算法框架下，重新定义粒子的基本运算，提出了一种通用的新型离散粒子群优化(Novel Discrete Particle Swarm Optimization, NDPSO)标准模型，并利用仿真实验验证本文提出算法的有效性。

2. 标准 PSO

Eberhard 和 Kennedy 于 1995 年提出 PSO 后，该算法得到各领域学者的广泛研究。为了更好的控制算法寻优能力，1998 年 Shi 等[10]进行了具有里程碑意义的研究，提出了惯性权重 ω，用 ω 来控制速度变化，较大的 ω 可以加强 PSO 的全局搜索能力，较小的 ω 则能加强局部搜索能力。Shi 等引入惯性权重的 PSO 被诸多学者称为标准 PSO 算法，可描述如下：

假设 $f(X)$ 是一个 d 维的最小化优化问题，对第 t 代的第 i 个粒子在第 d 维上的速度是 $v_{id}(t)\in[V_{\min},V_{\max}]$，其中 $V_{\min}$ 和 $V_{\max}$ 是依赖于问题的常数；位置为 $x_{id}(t)\in[X_{\min},X_{\max}]$，其中 $X_{\min}$ 和 $X_{\max}$ 为粒子搜索空间的边界；$p_{id}(t)$ 表示到目前为止第 i 个粒子所搜索的最优位置；$p_{gd}(t)$ 表示到目前为止所有粒子所搜索的最优位置。粒子的速度和位置的更新方程如下：

$$v_{id}(t+1)=\omega v_{id}(t)+c_1\xi\left(p_{id}(t)-x_{id}(t)\right)+c_2\eta\left(p_{gd}(t)-x_{id}(t)\right) \tag{1}$$

$$x_{id}(t+1)=v_{id}(t+1)+x_{id}(t),i=1,2,\cdots,m, \tag{2}$$

其中，m 为粒子个数；c_1 和 c_2 分别为粒子的自学习系数和社会学习系数，统称为加速系数；ξ 和 η 为 $[0,1]$ 之间是随机数，$\xi,\eta\in U(0,1)$。每一代最优位置的更新如下：

$$P_i(t+1)=\begin{cases}X_i(t+1), \text{if } f\left(X_i(t+1)\right)<f\left(P_i(t)\right)\\ P_i(t), \text{ otherwise}\end{cases} \tag{3}$$

***The procedure of PSO is given as follows*:**

Step 1: Initialization

Step 1.1. Initialize iterative counter be $t=0$, the m random velocities $v_i(0)$ and the m random positions $x_i(0)$ of the particles;

Step 1.2. Calculate $p_i(0)$ and $p_g(0)$;

Step 2: While termination criteria is satisfied do

Step 2.1. for ($i=1;i<=m;i++$) {

Update $v_i(t)$ using (1);

Update $x_i(t)$ using (2);

Update $p_i(t)$ using (3);

If ($f\left(p_i(t)\right)<f\left(p_g(t)\right)$) then $p_g(t)=p_i(t)$;

}//end for

Step 2.2. $t=t+1$;

Step 3: output.

3. 新型离散粒子群优化(NDPSO)

离散化粒子群优化算法的关键是如何定义 PSO 算法的基本运算规则[7]：位置的减法运算，即两个位置相减得到一个速度；速度的数乘运算，即一个数乘上一个速度后得到另一个速度；速度的加法运算，即两个速度相加得到另一个速度；粒子的移动，即位置与速度相加后得到一个新的位置。本节基于交叉算子、领域搜索算子和 PSO 算法信息更新的本质机理深入讨论该问题，提出一种新型的离散粒子群优化算法 NDPSO。

3.1. 粒子位置与位置的减法及速度数乘的复合运算

在标准PSO算法的公式(1)中 $c_1\left(p_{id}(t)-x_{id}(t)\right)$ 与 $c_2\left(p_{gd}(t)-x_{id}(t)\right)$ 分别表示粒子的历史最优位置和群体的最优位置对粒子新位置的影响。粒子位置与粒子位置的减法运算结果为粒子的速度，相当于遗传算法中的交叉算子；然后分别根据粒子的自学习系数 c_1 和社会学习系数 c_2 对速度 $\left(p_{id}(t)-x_{id}(t)\right)$ 和速度 $\left(p_{gd}(t)-x_{id}(t)\right)$ 进行调节，ξ 与 η 相当于遗传算法中的交叉概率。

Murata 等[11]提了置换流水车间调度问题最好的交叉算子(crossover for permutation flowshop scheduling, CPFS)：按交叉概率随机地从种群中选择两个个体作为父体，对于每个个体选取最先与最后回溯的工件位置作为交叉点，如果回溯工件个数小于(或等于) 1，则随机选择某两个(或一个)工件位置作为交叉点。首先将两个位置之前和之后的基因进行交叉复制，再按照父体中原来工件排列的顺序修补交叉部分中未包含的基因，该交叉操作既可以尽可能多地保留没有发生回溯的工件位置信息，又可以继承父染色体中工件之间的相对位置信息。如图 1 所示。

因此，定义粒子位置与位置的减法及速度数乘的复合运算 $V=c\times\left(X^a-X^b\right)$ 如下：

设速度 $V=\left(v_1,v_2,\cdots v_i,\cdots,v_n\right)$，位置 $X^a=\left(x_1^a,x_2^a,\cdots x_i^a,\cdots,x_n^a\right)$，位置 $X^b=\left(x_1^b,x_2^b,\cdots x_i^b,\cdots,x_n^b\right)$ 均是 n 维的行向量，常数 $c\in[0,1]$。V 的产生过程是生成 $[0,1]$ 之间的随机数 $rand(\,)$，若 $rand(\,)<c$，则以 X^a 和 X^b 为父体，随机选择某两个(或一个)工件位置作为交叉点，首先将两个位置之前和之后的部分进行交叉复制，然后按照父体中原来工件排列的顺序补齐交叉部分中没有包含的基因，再从两个子体中选取目标值较优的为 V；否则，$V=X^b$。

3.2. 常数与速度的乘法

在标准PSO算法中，常数与粒子速度的乘法运算结果为粒子的速度，相当于遗传算法中的变异算子。如，公式(1)中 $\omega v_{id}(t)$ 表示粒子的历史速度对新速度的影响，即惯性权重 ω 对速度 $\omega v_{id}(t)$ 的调节。公式(1)中 ω、$c_1\xi$ 和 $c_2\eta$ 相当于遗传算法中的变异概率。

在遗传算法中变异的目的是保持种群的多样性，邻域搜索是一种有效的求解大规模组合优化问题的优化方法，能够以初始解为基础遍历解空间，即其满足种群多样性的需求；特别地，通过邻域函数产生新解仍然在组合优化问题的解空间。邻域搜索算子主要包括移动 Swap(x, y)、插入 Insert(x, y)和 3-opt 等方式[12] [13]。如图 2 和图 3 所示。

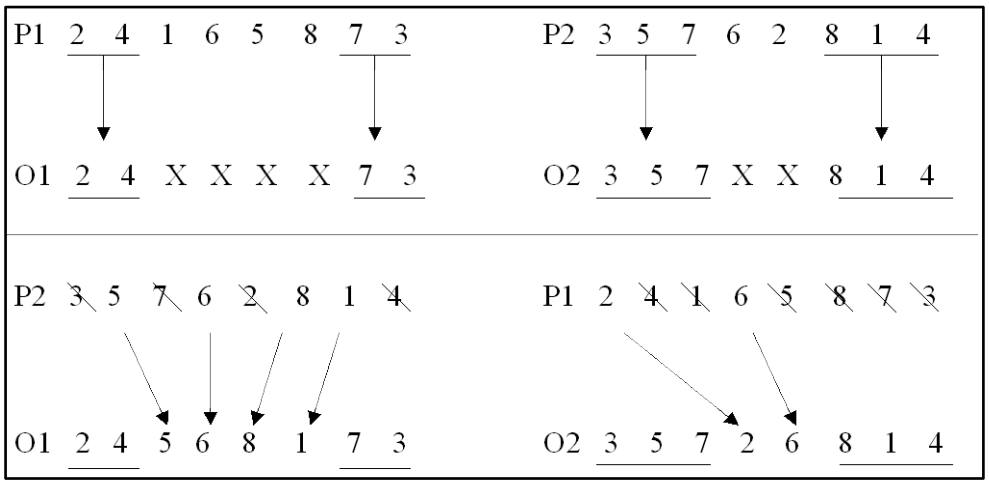

Figure 1. CPFS process diagram
图 1. CPFS 过程示意图

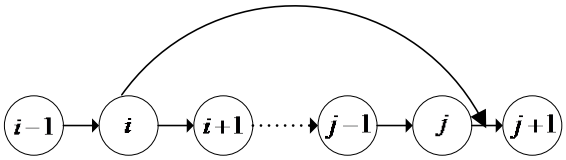

Figure 2. Insert(x, y) process diagram
图 2. Insert(x, y)过程示意图

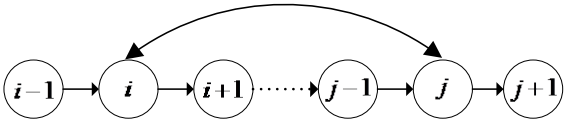

Figure 3. Swap(x, y) process diagram
图 3. Swap(x, y)过程示意图

对于 3-opt，任选节点 $i, j, k\left(i<j<k\right)$，删除边 $\left(i,i+1\right)$，$\left(j,j+1\right)$，$\left(k,k+1\right)$，为了避免将某部分路径反转(方向性变化使这部分路径的长度不可预计)，只能连接新边 $\left(i,j+1\right)$，$\left(j,k+1\right)$ $\left(k,i+1\right)$ 形成唯一的新路径，交换过程如图 4 所示，称这种 3-opt 为正方向 3-opt (positive direction 3-opt, PD3opt)。

因此，定义常数与速度的乘法 $V'=c\cdot V$ 如下：

设速度 $V'=\left(v_1',v_2',\cdots,v_i',\cdots,v_n'\right)$，速度 $V=\left(v_1,v_2,\cdots,v_i,\cdots,v_n\right)$ 均是 n 维的行向量，常数 $c\in\left[0,1\right]$，$1\le i\le n$。V' 的产生过程如下：

Step 1：生成 $\left[0,1\right]$ 之间的随机数 $rand\left(\right)$，$\left[1,n\right]$ 之间的随机整数 $randInt\left(\right)$，令 $t=0$，$V'=V$；

Step 2：$t++$; if $t\le randInt\left(\right)$, then go to Step 3; otherwise, go to Step 4；

Step 3：if $rand\left(\right)\le c$, then 以 V 为初始解，随机选取邻域搜索算子 Swap(x, y)、Insert(x, y)和 PD3opt 进行邻域搜索，并根据目标函数值更新 V'；otherwise, go to Step 4；

Step 4：stop, output.

3.3. 速度与速度加法的复合运算

在标准 PSO 算法中，粒子速度与粒子速度加法的运算结果为粒子的速度。并依据进化算法中“优胜劣汰”的基本规律。因此，定义 $V'=V+V^p+V^g$ 如下：

设某粒子的当前速度 $V=\left(v_1,v_2,\cdots,v_i,\cdots,v_n\right)$，粒子与自己的历史最优位置运算所得速度 $V^p=\left(v_1^p,v_2^p,\cdots,v_i^p,\cdots,v_n^p\right)$，粒子与全局最优位置运算所得速度 $V^g=\left(v_1^g,v_2^g,\cdots,v_i^g,\cdots,v_n^g\right)$，粒子的新速度 $V'=\left(v_1',v_2',\cdots,v_i',\cdots,v_n'\right)$，则以 V、V^p 和 V^g 两两之间形成父体，选取 CPFS 算子以概率 1 进行优化，选择较优的子体为 V'。

3.4. 速度与位置的加法运算

在标准 PSO 算法中，粒子速度与粒子位置加法的运算结果为粒子的位置。并依据进化算法中“优胜劣汰”的基本规律。因此，定义 $X'=X\oplus V$ 如下：

设某粒子的当前速度 $V=\left(v_1,v_2,\cdots,v_i,\cdots,v_n\right)$，粒子的新位置 $X'=\left(x_1',x_2',\cdots,x_i',\cdots,x_n'\right)$，则以 V 与 X 形成父体，选取 CPFS 算子以概率 1 进行优化，选择较优的子体为 X'。

3.5. NDPSO 中粒子速度和位置的更新公式

重新定义粒子的基本运算后，粒子速度和位置的更新公式如下：

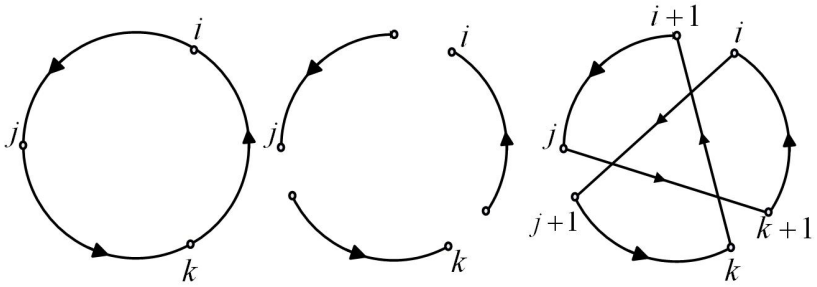

Figure 4. PD3opt process diagram
图 4. PD3opt 过程示意图

$$V_t^p = c_1 \times \left(X^p - X_t \right) \tag{4}$$

$$V_t^g = c_2 \times \left(X^g - X_t \right) \tag{5}$$

$$V_{t+1} = \left(\omega \cdot V_t \right) + V_t^p + V_t^g \tag{6}$$

$$X_{t+1} = X_t \oplus V_{t+1} \tag{7}$$

其中，式(4)表示由粒子的第 t 代位置向量 X_t，粒子自己的历史最优位置向量 X^p 和粒子对局部最优解的信任概率 $c_1 \in [0,1]$，计算第 t 代的局部速度向量 V_t^p；

式(5)表示由粒子的第 t 代位置向量 X_t，全局最优位置向量 X^g 和粒子对全局最优解的信任概率 $c_2 \in [0,1]$，计算第 t 代的全局速度向量 V_t^g；

式(6)表示由粒子的第 t 代速度向量 V_t，惯性权重 $\omega \in [0,1]$，局部速度向量 V_t^p，全局速度向量 V_t^g，计算第 $t+1$ 代的速度向量 V_{t+1}；

式(7)表示由粒子的第 t 代位置向量 X_t，速度向量 V_t，局部速度向量 V_t^p，全局速度向量 V_t^g，计算第 $t+1$ 代的位置向量 X_{t+1}。

4. 仿真实验

4.1. 实验设计和参照算法

为了测试本文提出算法的性能，以 23 个无等待流水车间调度问题[14]标准算例为测试数据。采用 VC++6.0 作为编程语言实现 NDPSO 算法，实验在配置为 Pentium3/CPU3.00GHz/RAM512M 的台式机上进行。

NDPSO 算法的基本流程与标准 PSO 算法相同，只是根据公式(4)、(5)、(6)和(7)更新种群。种群大小为 100，最大迭代次数为 500。每组实验独立运行 30 次，记录 makespan 和 CPU 的平均值。

基于下降算法和禁忌搜索的 DS + M 算法和 TS + M 算法是文献[11]提出的五种算法中更有优势的两种算法；变邻域搜索算法 VNS 和模拟退火与遗传的混合算法 GASA 是文献[15]提出的算法；离散粒子群 DPSO 是文献[16]提出的算法。

4.2. 实验结果与分析

实验结果如表 1 所示，其中 CPU 表示算法结束所需的时间。PRD 为所用算法与参考算法值相比的偏差，$\mathrm{PRD}(\mathrm{A}) = 100\left(C^{\mathrm{A}} - C^{\mathrm{Ref}}\right) / C^{\mathrm{Ref}}$，为了使结果具有可比性，本文借鉴上几种算法，以文献[17]中的 RAJ 作为参考值，C^{Ref} 的取值为 C^{RAJ}，C^{A} 代表算法 A 的 makespan 值，$\mathrm{A} \in \{\mathrm{VNS}, \mathrm{GASA}, \mathrm{DS}+\mathrm{M}, \mathrm{TS}+\mathrm{M}, \mathrm{IDPSO}\}$。

从表 1 可知：

Table 1. Comparison of the experiment results obtained by various algorithms
表 1. 各种算法实验结果比较

VNS		GASA		DS + M		TS + M		DPSO		NDPSO	
PRD	CPU	PRD	CPU	PRD	CPU	PRD	CPU	PRD	CPU	PRD	CPU
−5.61	23.78	−1.18	200.13	−4.53	0	−6.59	0.87	−4.85	0.043	−6.71	2.95

1) 在求解质量方面，本文提出的 NDPSO 算法明显优于其它算法。这表明本文基于粒子群基本优化机制的 NDPSO 算法具有良好的求解性能，可用于组合优化问题的求解。

2) 在计算时间方面，本文提出的 NDPSO 算法略差于 DS+M 算法、TS + M 算法和 DPSO 算法，但明显优于 VNS 和 GASA 算法。

5. 结束语

粒子群算法是近年来发展起来的智能优化算法，因其在连续空间优化领域取得的巨大成功，使许多学者试图用它去解决组合优化问题。本文基于交叉算子、邻域搜索算子和粒子群算法信息更新的本质机理，重新定义了粒子的基本运算，包含粒子位置的减法运算、粒子速度的数乘运算、粒子速度的加法运算和粒子位置与速度的加法运算；给出了一种针对组合优化问题，通用的离散粒子群优化算法。最后，以 23 个标准算例为实验数据进行了仿真实验，实验结果表明了本文所提出算法的有效性。为了更好地求解组合优化问题，未来可进一步研究 NDPSO 算法的进化行为，如根据特定问题在基本流程中引入具有针对性的邻域搜索机制。

基金项目

国家社会科学基金西部项目(17XGL016)，宁夏自然科学基金(NZ17083)，宁夏高等学校科学研究项目(NGY2016085)，2017 年宁夏医科大学优秀青年后备骨干培育对象(宁医校发【2017】119 号)。

参考文献 (References)

[1] Eberhard, R. and Kennedy, J. (1995) A New Optimizer Using Particle Swarm Theory. *Proceedings of Sixth International Symposium on Micro Machine and Human Science*, Nagoya, 4-6 October 1995, 39-43. https://doi.org/10.1109/MHS.1995.494215

[2] Kennedy, J. and Eberhard, R. (1995) Particle Swarm Optimization. *Proceedings of IEEE Conference on Neural Networks*, Piscataway, IEEE Press, 1942-1948. https://doi.org/10.1109/ICNN.1995.488968

[3] Lin, C.J. and Hsieh, M.H. (2008) Classification of Mental Task from EEG Data Using Neural Networks Based on Particle Swarm Optimization. *Neurocomputing*, **72**, 1121-1130. https://doi.org/10.1016/j.neucom.2008.02.017

[4] 谈超, 李小平. 双目标无等待流水线调度的加权混合算法[J]. 计算机科学, 2008(11): 199-213.

[5] Karakuzu, C. (2008) Fuzzy Controller Training Using Particle Swarm Optimization for Nonlinear System Control. *ISA Transactions*, **47**, 229-239. https://doi.org/10.1016/j.isatra.2007.09.003

[6] 沈林成, 霍霄华, 牛轶峰. 离散粒子群优化算法研究现状综述[J]. 系统工程与电子技术, 2008(10): 1986-1994.

[7] Clerc, M. (2004) Discrete Particle Swarm Optimization Illustrated by the Traveling Salesman Problem. *Springer Berlin Heidelberg*, **47**, 219-239. https://doi.org/10.1007/978-3-540-39930-8_8

[8] Wang, K.P., Huang, L., Zhou, C.G., *et al*. (2003) Particle Swarm Optimization for Traveling Salesman Problem. *International Conference on Machine Learning and Cybernetics*, **3**, 1583-1585.

[9] Shi, X.H., Liang, Y.C., Lee, H.P., *et al*. (2007) Particle Swarm Optimization-Based Algorithms for TSP and Generalized TSP. *Information Processing Letters*, **103**, 169-176. https://doi.org/10.1016/j.ipl.2007.03.010

[10] Shi, Y. and Eberhard, R. (1998) A Modified Particle Swarm Optimizer. *IEEE World Congress on Computational Intelligence*, Anchorage, 69-73. https://doi.org/10.1109/ICEC.1998.699146

[11] Murata, T., Ishibuchi, H. and Tanaka, H. (1996) Genetic Algorithms for Flowshop Scheduling Problems. *Computers*

and Industrial Engineering, **30**, 1061-1071. https://doi.org/10.1016/0360-8352(96)00053-8

[12] Eksioglu, B., Eksioglu, S.D. and Jain, P. (2008) A Tabu Search Algorithm for the Flowshop Scheduling Problem with Changing Neighborhoods. *Computers & Industrial Engineering*, **54**, 1-11. https://doi.org/10.1016/j.cie.2007.04.004

[13] Lawler, E.L., Lenstra, J.K., Rinnooy, K.A., *et al*. (1997) The Traveling Salesman Problem—A Guided Tour of Combinatorial Optimization. John Wiley & Sons, Hoboken.

[14] Grabowski, J. and Pempera, J. (2005) Some Local Search Algorithm Search for No-Wait Flow-Shop Problem with Makespan Criterion. *Computers & Operations Research*, **32**, 2197-2212. https://doi.org/10.1016/j.cor.2004.02.009

[15] Schuster, C.J. and Framinan, J.M. (2003) Approximative Procedures for No-Wait Job Shop Scheduling. *Operations Research Letters*, **31**, 308-318. https://doi.org/10.1016/S0167-6377(03)00005-1

[16] 潘全科, 谢圣献, 张亚卿, 等. 解决无等待流水线调度问题的新算法[J]. 机械科学与技术, 2006(12): 1487-1490.

[17] Rajendran, C. (1994) A No-Wait Flowshop Scheduling Heuristic to Minimize Makespan. *Journal of the Operational Research Society*, **45**, 472-478. https://doi.org/10.1057/jors.1994.65