doi:10.3969/j.issn.1673-4785.2010.05.006

求解流水线调度问题的万有引力搜索算法

谷文祥,李向涛,朱 磊,周俊萍,胡艳梅
(东北师范大学 计算机学院,吉林 长春 130117)

摘 要:研究了以最大完工时间为目标的流水线调度问题,使用万有引力算法求解调度问题,提出了一种最大排序规则,利用物体间各个位置分量值存在的大小次序关系,并结合随机键编码的方法产生,将物体的连续位置转变成了一个可行的调度方案;提出了一种边界变异的策略使得越界的物体不再聚集在边界上,而是分布在边界附近的可行空间内,从而增加种群的多样性;结合交换算子和插入算子提出了一种新的局部搜索算法,有效地避免了算法陷入局部最优值,进一步提高了解的质量.最后证明了算法的收敛性,并且计算了算法的时间复杂度和空间复杂度,仿真实验说明了所得算法的有效性.

关键词:万有引力搜索算法;流水线调度;局部搜索算法;边界变异;最大排序规则;最大完工时间
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-4785(2010)05-0411-08

A gravitational search algorithm for flow shop scheduling

GU Wen-xiang, LI Xiang-tao, ZHU Lei, ZHOU Jun-ping, HU Yan-mei
(Department of Computer Science, Northeast Normal University, Changchun 130117, China)

Abstract: An improved gravitational search algorithm (IGSA) was proposed to solve the flow shop scheduling problem with the objective of minimizing production time. First, to make a GSA suitable for permutation of the flow shop scheduling problem (PFSSP), a new largest-rank-rule based on a random key was introduced to convert the continuous position of the GSA into the discrete job permutation so that the GSA could be used for solving PFSSP. Second, a new boundary mutation was proposed. This operation stopped the agents which have mutations as a result of using the above method from gathering at the border. They were distributed at a feasible distance from the boundary. This improvement also improved the population diversity. Third, by combining the communicating operator and inserting operator, the new local search was designed to help the algorithm escape from the local minimum. Finally, the convergence of the iterative algorithm and its complexities in time and space were proven. Additionally, simulations and comparisons based on PFSSP benchmarks were carried out, which show that the proposed algorithm is both effective and efficient.

Keywords: gravitational search algorithm; flow shop scheduling; local search; boundary mutation; largest rank rule; production time minimizing

流水线调度问题(flow shop scheduling problem, FSP)是许多实际生产调度过程的简化模型,它已经被证明是NP-hard[1-2].由于其重要的理论意义和工程价值,FSP问题已成为目前调度问题研究的热点之一.流水线调度是研究n个工件在m台机器上的流水加工过程,同时约定:1)每个工件在每台机器上只能加工1次;2)每个工件的加工不能被中断;3)每台机器每次最多只能加工1个工件;4)每台机器执行工件的顺序是相同的.目前求解该问题的方法通常包括3种:精确算法[3-4]、启发式算法[5-6]以及元启发式算法[7-8].精确算法一般仅仅适合于小规模问题;构造启发式算法和提高式启发式算法虽然能够解决部分大规模问题,但是解的质量往往不高;元启发式算法通过模仿自然界的某些现象和过程进行优化,一般是从若干个解出发,通过对搜索空间的不断搜索,直到达到一定的条件,最终获得该问题的最优解或者近似最优解.因此,元启发式算法成为了求解流水线调度问题的主要方法.

万有引力搜索算法(GSA)[9]是由E. Rashedi和H. Nezamabadi-pour等在2009年提出的一种源于对物理学中的万有引力进行模拟的新的优化搜索技术,与粒子群算法(PSO)相似,是一种元启发式算

收稿日期:2010-03-29.
基金项目:国家自然科学基金资助项目(60473042, 60573067, 60803102).
通信作者:李向涛. E-mail: lixt314@nenu.edu.cn.

法.它通过群体中各粒子之间的万有引力相互作用产生的群体智能指导优化搜索,但是目前对于GSA算法的研究还很少.实验证明,GSA的收敛性明显优于PSO、遗传算法(GA)等其他智能优化算法,但已提出的算法都存在早熟收敛,易陷入局部最优,缺少有效加速机制.

为了解决以上的不足,本文提出了一种基于万有引力搜索算法的IGSA算法用于求解流水线调度问题.考虑到连续的GSA算法不能处理离散的调度问题,提出了一种最大位置规则.为了提高GSA算法的效率,提出了一种边界变异策略,进而增加种群的多样性.并且为了进一步提高解的质量,提出了一种新的局部搜索算法.最后分析了IGSA算法的收敛性,并证明该算法的空间复杂度为$O((6n+2m)\times d)$和群体每次进化的最坏渐进时间复杂度为$O(mnd^2)$.仿真实验结果表明该算法是有效的.

1 流水线调度问题的数学模型

流水线调度问题可描述如下:n个待加工的作业$N=J_1,J_2,\cdots,J_n$需要在m台机器加工$M=M_1,M_2,\cdots,M_n$,每个作业由m道连贯的工序组成,每道工序要求不同的机器完成,这些作业通过机器的顺序是一样的,它们在每台机器上的顺序也是一样的,每个作业J_i在机器m_j上的处理时间为P_{ij}.对于一个可行调度$\pi=\{\pi_1,\pi_2,\cdots,\pi_n\}$,各作业的完工时间可按如下方法计算:

$$C(\pi,1)=p_{\pi_1,1},$$
$$C(\pi_j,1)=C(\pi_{j-1},1)+p_{\pi,1},j=2,3,\cdots,n,$$
$$C(\pi_1,i)=C(\pi_1,i-1)+p_{\pi,i},i=2,3,\cdots,m,$$
$$C(\pi_j,i)=\max(C(\pi_{j-1},i),C(\pi_j,i-1))+p_{\pi,i},$$
$$j=2,3,\cdots,n,\ i=2,3,\cdots,m.$$

最大完工时间可表示为

$$C_{\max}(\pi)=C(\pi_n,m).$$

流水线调度的目标是发现一个最好的方案π^*,使得

$$C_{\max}(\pi^*)\leqslant C(\pi_n,m),\forall\pi\in\prod.$$

2 万有引力算法

万有引力搜索算法[9]是基于引力法则,它们质量的性质对它们之间的行为表现有着一定的作用.物体之间都是相互吸引的,相互之间的作用力引起物体之间都朝着质量大的物体的方向移动,在GSA中,每个物体有4个特征:位置、惯性权重、施力物、受力物.物体的位置就是问题的解.每个物体被定义在一个n维的搜索空间内,由N个物体组成的种群$X=(x_1,x_2,\cdots,x_n)$,其中第i个物体的位置为$X_i=(x_i^1,x_i^2,\cdots,x_i^d,\cdots,x_i^n)$,$x_i^d$代表物体$i$在$d$维空间的位置.

在特定的时间t,定义作用在物体i和物体j之间的重力搜索表示为

$$F_{ij}^d(t)=G(t)\frac{M_{pi}(t)\times M_{aj}(t)}{R_{ij}(t)+\varepsilon}(x_j^d(t)-x_i^d(t)).$$

式中:M_{aj}表示施力物j的质量;M_{pi}表示受力物i的质量;ε是一个小的常量;$R_{ij}(t)$是物体i和物体j之间的欧几里德距离:

$$R_{ij}(t)=\|X_i(t),X_j(t)\|_2;$$

$G(t)$是在时间t下的引力常量:

$$G(t)=G_0e^{-\alpha\frac{t}{T}}.$$

则物体i在d维的合力可以表示为

$$F_i^d=\sum_{j=1,j\neq i}^{N}\text{rand}\times F_{ij}^d(t).$$

通过运动法则,在时间t时物体i在第d维的加速度:

$$a_i^d(t)=\frac{F_i^d(t)}{M_{ii}(t)}.$$

式中:$M_{ii}(t)$表示物体i的惯性质量.则物体的速度更新为

$$v_i^d(t+1)=\text{rand}\times v_i^d(t)+a_i^d(t),$$
$$x_i^d(t+1)=x_i^d(t)+v_i^d(t+1).$$

式中:rand是一个在[0,1]之间的随机数,用它可以让搜索变得随机化.

重力搜索和惯性质量通过适应度函数计算得到.质量高的物体吸引力强,移动地比较慢,假设引力质量和惯性质量相等,物体的质量可以使用1个隐射.则

$$M_{ai}=M_{pi}=M_{ii}=M,\ i=1,2,\cdots,N,$$
$$m_i(t)=\frac{m_{\text{fiti}}(t)-m_{\text{worst}}(t)}{m_{\text{best}}(t)-m_{\text{worst}}(t)},$$
$$M_i(t)=\frac{m_i(t)}{\sum_{j=1}^{N}m_j(t)}.$$

式中:$m_{\text{fiti}}(t)$表示在时刻t,物体i的最好适应度值,$m_{\text{best}}(t)$、$m_{\text{worst}}(t)$被定义为

$$m_{\text{best}}(t)=\min_{j\in\{1,2,\cdots,N\}}m_{\text{fiti}}(t),$$
$$m_{\text{worst}}(t)=\max_{j\in\{1,2,\cdots,N\}}m_{\text{fiti}}(t).$$

3 IGSA调度算法

3.1 最大排序规则

万有引力搜索算法(GSA)是一种连续的智能算法,标准的万有引力搜索算法所具有的连续编码不能被直接地用于求解流水线调度.因此构造从物体位置矢量到工件排序的恰当映射是应用GSA算法解决流水线调度问题的首要和关键问题.在本文中,提出一种基于随机键的新的排序规则——最大排序规则.使用这个规则,可以将连续的物体位置转化为离散的调度方案.在这个规则中,最大的实数将首先被挑选出作为新的调度方案的第1个位置,稍小的实数将被挑选出作为新的调度方案的第2个位置.

用这种方式,所有物体的位置将被转化为一个新的调度方案. 在表 1 中,使用一个简单的例子来阐述这个规则. 从表 1 中,可以很显然地看出 1.45 是最大的值,所以维数为 4 对应的调度排序为 1. 稍小的是 1.32,则维数为 6 对应的调度排序为 2. 用相同的方法,可以得到它的调度方案为 $\pi=(3,4,5,1,6,2)$. 虽然这个方法比较简单,但是它能使得 GSA 能够求解流水线调度问题.

表 1　X_i^t 的解表示

Table 1　Solution representation of Agent X_i^t

维数	1	2	3	4	5	6
x_{ij}^t	1.25	0.85	0.63	1.45	0.23	1.32
π_{ij}^t	3	4	5	1	6	2

3.2　边界变异

在 GSA 算法的实现过程中,当某个物体在搜索过程中由于牛顿第二定律(运动定律)的作用使得物体运动出可行域时,通常的处理方法是使该位置处于边界上,即为如下的方法:

If $(x_i > x_{\max})$ then

　　$x_i = x_{\max}$

End

If$(x_i < x_{\min})$then

　　$x_i = x_{\min}$

End

经过这种处理后,所有的越界的物体都聚集在边界处. 对于调度问题,这样的设置将导致排序不惟一,当同时有 2 个物体都越界,这样将会有 2 个相同的值,进行排序会产生问题. 为了避免这种问题的发生,将越界的物体作下面的改变,提出一种边界变异的方法:

If$(x_i > x_{\max})$ then

　　$x_i = x_{\max} - c * \mathrm{rand}(\cdot) * x_{\max}$

End

If $(x_i < x_{\min})$ then

　　$x_i = x_{\min} + c * \mathrm{rand}(\cdot) * (-x_{\min})$

End

其中:$c = 0.01$,rand 为[0,1]之间的一个随机数,从以上的过程可以看出,对越界的物体做了变异操作后,物体不再聚集在边界上,而是分布在离边界 $c \times \mathrm{rand}$ 附近的可行空间内. 通过这种操作,即使物体在不可行空间内,克服了使用最大排序规则时产生的排序不惟一问题,同时也增加了物种的多样性.

3.3　IGSA 算法的初始化

初始物体的位置是 GSA 搜索的起点. 好的初始群体有助于提高解的质量. 一个好的初始种群可以保证一定的分布性,这样可以分布覆盖整个解空间. 从另一方面应包含部分较高质量的个体,以指导算法能搜索到比较好的解. 所以,一个高质量的初始群体,可以提高优化算法性能. 采用如下方式生成:

$$x_{ij}^t = (x_{\max} - x_{\min}) \times r + x_{\min}.$$

式中:$x_{\min}=0$, $x_{\max}=4.0$, r 为(0,1)内产生的随机数.

3.4　局部搜索算法

IGSA 算法虽然收敛速度比较快,但是它的收敛精度比较低. 算法虽然具有比较强的全局搜索能力,但它的局部搜索能力较弱,为了提高解的质量,引入了 2 种局部搜索的算子:插入和交换. 这样可以有效地避免搜索过程陷入局部极小值,也有利于全局空间搜索和局部区域改良能力的平衡.

交换算子:从 1 个调度方案挑选出 2 个不同作业,并且交换它们的位置.

插入算子:从 1 个调度方案挑选出 2 个不同作业,将这 2 个作业之间的作业随着相同的方向移动.

表 1 显示的调度方案为 $\pi=(3,4,5,1,6,2)$,随机选出 2 个位置 2 和 4,表 2 显示出执行交换算子所得结果,表 3 显示出执行插入算子所得结果.

表 2　交换算子

Table 2　Swap

维数	1	2	3	4	5	6
x_{ij}^t	1.25	1.45	0.63	0.85	0.23	1.32
π_{ij}^t	3	1	5	4	6	2

表 3　插入算子

Table 3　Insert

维数	1	2	3	4	5	6
x_{ij}^t	1.25	0.63	1.45	0.85	0.23	1.32
π_{ij}^t	3	5	1	4	6	2

在此基础上,提出一种新的局部搜索算法,可以有效地提高解的质量. 算法的伪代码工作流程如下描述.

```
Procedure local search (π)
Begin
Let s_g=π permutation of the global solution
  for i=1 to genmax
     s_1=s_g
     f(s_1)=f(s_g)
  Find r_1 and r_2 randomly and r_1≠r_2
  if (rand<0.5)
     s'=swap(s_1, r_1, r_2)        //Swap
  else
     s'=insert(s_1, r_1, r_2)      //Insert
  end
  calculate f(s')  //Calculate the makespan
                     of the new permutaion
  if (f(s')-f(s_g)<=0)
     s_g=x'
     f(s_g)=f(s')
  end                          //end if
  end                          //end for
End
```

3.5 IGSA 算法求解流水线调度问题

图 1 所示为 IGSA 求解流水线调度问题的流程图. 从图 1 中可以看出,一种新的最大排序规则被提出,这种规则可以使连续本质的万有引力算法能直接用于调度问题. 其次,还提出一种边界变异的策略使得越界的物体不再聚集在边界上,而是分布在离边界 $c \times \text{rand}$ 附近的可行空间内,从而增加种群的多样性. 最后,使用一种新的局部搜索算法,局部搜索算法可以有效地避免搜索过程陷入局部极小值,提高解的质量.

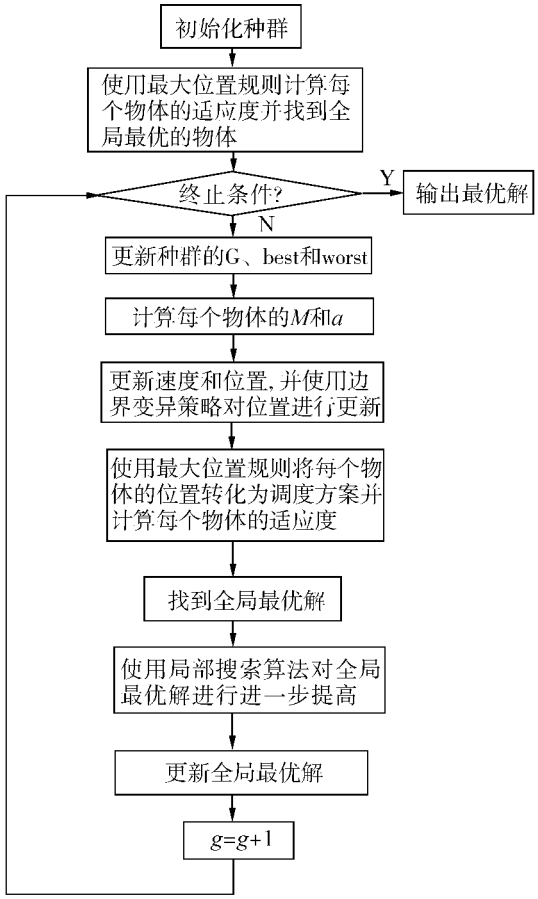


图 1 IGSA 求解流水线调度问题的流程图

Fig. 1 The framework of the proposed algorithm IGSA

4 算法分析

在基于群体的优化模型中,最优解往往是可以预先知道的,本节将讨论利用 IGSA 算法解决流水线调度问题的收敛性问题. 收敛性通常是指一个系统或过程达到一个稳定状态,对基于智能的优化算法来说,算法的收敛可以根据群体的行为来定义.

定义 1 若 IGSA 算法在 t 时刻或者在第 t 次迭代种群对应的调度 $\pi(t)$ 满足

$$\lim_{t\to\infty} \pi(t) = \pi(0),\ \pi(0) \in \pi,$$

那么就称 IGSA 算法具有渐进收敛性.

定义 2 设 IGSA 算法的 π_t 是指在 t 时刻或者在第 t 次迭代种群对应的调度方案的最大完工时间的最小值,π^* 为调度问题在所有的可能调度方案组成的集合 π 中所取的最小值,若 π_t 满足:

$$\lim_{t\to\infty} P\{\pi_t = \pi^*\} = 1,$$

则称 IGSA 算法收敛到最优解.

定理 1 IGSA 算法求得的解是有效的.

证明 结合文献[10],每种操作都满足"all different"约束,IGSA 算法的搜索空间都是由所有有效的调度方案构成的,所以 IGSA 求得的解是有效的.

定理 2 设种群规模为 m,问题规模为 l,种群对应的调度状态记为$(c_1, c_2, \cdots, c_m)$,经过若干次迭代后,种群对应的调度状态可以收敛到$(c^*, c^*, \cdots, c^*)$.

证明 结合文献[10],对于改进的万有引力算法,可以把每个物体看作一个状态,很显然,所有物体的当前位置可以看作一种状态的分布,由于万有引力算法是一种随进算法,所以这种状态是会随着算法的运行而发生改变的,算法只与当前的状态有关,而与初始状态无关.

种群对应的调度的状态记为$(c_1, c_2, \cdots, c_m)$,最佳调度状态为 c^*,所有的种群状态构成了一个 $1 \times N$ 的行向量,其中 $N = (m \times l)!$,这个行向量的每个元素由排在一起的实数构成,可表示为

$$\{(c_1^1, c_2^1, \cdots, c_m^1), (c_1^2, c_2^2, \cdots, c_m^2), \cdots, (c_1^N, c_2^N, \cdots, c_m^N)\}.$$

经过若干次进化后,已达到最优的种群的状态$(c^*, c^*, \cdots, c^*)$,不妨设其排在状态行向量的第 1 分量处,即

$$\{(c_1^*, c_2^*, \cdots, c_m^*), (c_1^1, c_2^1, \cdots, c_m^1), \cdots, (c_1^{N-1}, c_2^{N-1}, \cdots, c_m^{N-1})\}.$$

由物体的速度和位置更新公式可知,算法要将当前种群中的最优解与前一代保留下来的最优解进行比较,这一操作用矩阵 $\boldsymbol{M}_U$ 来表示,那么从种群第 t 次迭代 C^t 到 C^{t+1} 的转移概率为

$$P(C^{t+1} = s_j \mid C^t = s_i) = \begin{cases} 1, & \min\{f(C_i^t), i = 1,2,\cdots,m\} = f(C_i^{t+1}), \\ & C^t = C^{t+1}; \\ 0, & \text{其他.} \end{cases}$$

从而矩阵 $\boldsymbol{M}_U$ 每一行有且仅有一个 1. 个体要么被替代,要么不变. 因此,更新矩阵可写为下三角的形式:

$$\boldsymbol{M}_U = \begin{bmatrix} \boldsymbol{M}_{U_1,1} & & & \\ \boldsymbol{M}_{U_2,1} & \boldsymbol{M}_{U_2,2} & & \\ \vdots & \vdots & \ddots & \\ \boldsymbol{M}_{U_k,1} & \boldsymbol{M}_{U_k,2} & \cdots & \boldsymbol{M}_{U_k,k} \end{bmatrix}. \quad (1)$$

式中:$\boldsymbol{M}_{U_i,j}$ 为 $k \times k$ 维,而 $\boldsymbol{M}_{U_i,1}$ 是一个单位阵. 将式(1)转化为

$$\boldsymbol{M} = \begin{bmatrix} 1 & \boldsymbol{\Phi} \\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22} \end{bmatrix}.$$

式中:$\boldsymbol{M}$ 为随机矩阵,其每一行至少 1 个正数,$\boldsymbol{\Phi}$ 为 $N-1$ 维行向量,显然 $\boldsymbol{M}$ 是一个可约随机矩阵,且 $\boldsymbol{M}_{21}$、$\boldsymbol{M}_{22}$不是零矩阵. 根据可约随机矩阵的性质有 $\boldsymbol{M}^{\infty} = (M_1^{\infty}, 0, \cdots, 0)$,其中 $M_1^{\infty} > 0$. 因为 $\boldsymbol{M}^{\infty}$ 可以看作一个状态分布,所以 $M_1^{\infty} = 1$,于是 $\boldsymbol{M}^{\infty} = (1, 0, \cdots, 0)$,即算法收敛到 c^*.

所以,经过若干次迭代后,种群对应的调度状态可以收敛到$(c^*, c^*, \cdots, c^*)$.

定理 3　IGSA 算法的空间复杂度为 $O((6n + 2m) \times d)$,群体每次进化的最坏渐进时间复杂度为 $O(mnd^2)$,其中 n 为群体规模,d 表示作业数量,m 表示处理机个数.

证明　在 IGSA 算法运行中,物体的当前位置、当前速度、物体的质量、物体的加速度、物体之间的距离以及物体所受的合力都要被存储,群体规模为 n,那么它所消耗的存储空间为 $O(6nd)$,在求解物体的适应度时还要存储调度问题的时间矩阵,所消耗的存储空间为 $O(2md)$,所以 IGSA 算法的空间复杂度为 $O((6n + 2m) \times d)$. 求解每个物体适应度时需要遍历与之相对应的时间矩阵,其时间复杂度为 $O(2md)$,由于需要计算所有物体的适应度,那么需要消耗的时间为 $O(2mnd)$,在最坏情况下,在每一次迭代中每个物体都要更新它的当前位置和速度,质量和加速度的时间复杂度为 $O(6mnd^2)$,当规模逐渐增大即 $m \times n$ 趋于无穷时,$6mnd^2 \gg 12nd \gg 2n$,IGSA 的时间复杂度为$O(mnd^2)$.

5　仿真实验

实验采用 Carlier[11]、Reeves[12]、Trillard[13] 所提供的不同规模的数据集,为了检测算法的性能,在处理器为 Pentium 3.0 GHz、内存为 1.0 GB 的 PC 机上进行测试,采用 Matlab7.0 编码. 算法的参数设置为:种群为作业数 n 的 2 倍,最大迭代次数为1 000,局部搜索的迭代次数为 $5n(n-1)$. 每个作业运行 20 次. 在 Cariler、Reeves 数据集中,使用本文提出的 IGSA 算法分别与几种构造启发式算法[14] 和元启发式算法[15] 进行比较,实验结果如表 4 ~ 7 所示. 表 4、6 是统计的最小完工时间,表 5、7 是最小完工时间的百分比. 实验结果表明 IGSA 是优于其他算法的.

表 4　IGSA 算法在 Cariler 上与构造启发式算法比较

Table 4　Comparison of IGSA and construction heuristics on Carlier

问题	最优解	问题规模		构造启发式算法					IGSA 算法
		n	m	Palme	Gupta	CDS	RA	NEH	
Car1	7 038	11	5	7 472	7 348	7 202	7 817	7 038	7 038
Car2	7 166	13	4	7 940	7 534	7 410	7 509	7 940	7 166
Car3	7 312	12	5	7 725	7 399	7 399	7 399	7 503	7 312
Car4	8 003	14	4	8 423	8 423	8 423	8 357	8 003	8 003
Car5	7 720	10	6	8 520	8 773	8 627	8 940	8 190	7 720
Car6	8 505	8	9	9 487	9 441	9 553	9 514	9 519	8 505
Car7	6 590	7	7	7 639	7 639	6 819	6 923	7 668	6 590
Car8	8 366	8	8	9 023	9 224	8 903	9 062	9 032	8 366

表 5　IGSA 算法在 Cariler 上与元启发式算法比较

Table 5　Comparison of IGSA and meta-heuristics on Cariler

问题	问题规模		元启发式算法					IGSA 算法
	n	m	GA	H_GA	DPSO	DPSO + LS	SPSOA	
Car1	11	5	0.00	0.00	0.00	0.00	0.00	0.00
Car2	13	4	0.00	0.00	0.00	0.00	0.00	0.00
Car3	12	5	2.42	0.00	1.09	0.00	0.00	0.00
Car4	14	4	0.00	0.00	0.00	0.00	0.00	0.00
Car5	10	6	0.36	0.00	0.62	0.00	0.00	0.00
Car6	8	9	0.00	0.76	0.00	0.00	0.00	0.00
Car7	7	7	0.00	0.00	0.00	0.00	0.00	0.00
Car8	8	8	0.00	0.00	0.00	0.00	0.00	0.00

表6 IGSA算法在Reeves上与构造启发式算法比较

Table 6 Comparison of IGSA and construction heuristics on Reeves

问题	最优解	问题规模		构造启发式算法					IGSA算法
		n	m	Palme	Gupta	CDS	RA	NEH	
Rec01	1 247	20	5	1 391	1 434	1 399	1 399	1 334	1 247
Rec03	1 109	20	5	1 223	1 380	1 273	1 159	1 136	1 109
Rec05	1 242	20	5	1 290	1 429	1 338	1 434	1 294	1 245
Rec07	1 566	20	10	1 715	1 678	1 697	1 722	1 637	1 566
Rec09	1 537	20	10	1 915	1 792	1 639	1 714	1 692	1 537
Rec11	1 431	20	10	1 685	1 765	1 597	1 636	1 635	1 431

表7 IGSA算法在Reeves上与元启发式算法比较

Table 7 Comparison of IGSA and meta-heuristics on Reeves

问题	问题规模		元启发式算法				IGSA算法
	n	m	GA	H_GA	DPSO	DPSO+LS	
Rec01	20	5	8.26	0.14	1.36	0.16	0.00
Rec03	20	5	7.21	0.09	0.54	0.18	0.00
Rec05	20	5	5.23	0.29	0.97	0.24	0.24
Rec07	20	10	8.56	0.69	3.70	1.15	0.00
Rec09	20	10	5.14	0.64	5.47	2.41	0.13
Rec11	20	10	8.32	1.10	11.11	1.05	0.51

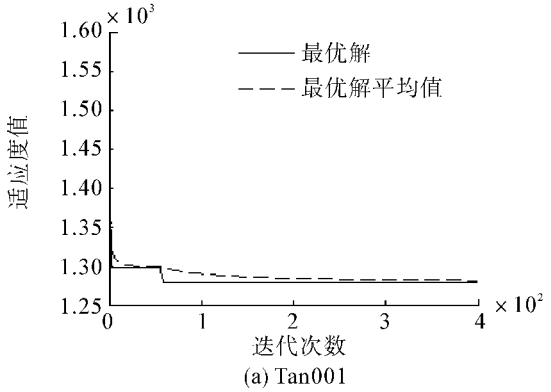
对于Trillard数据集,将本文的IGSA算法与Lian提出的NPSO算法[8]比较,IGSA使用和NPSO相同的参数设置,种群个数为50,各个规模的实验最大迭代次数依次为:400、400、400、500、2 000、2 000、800.实验结果如表8所示.

表8 IGSA算法与NPSO比较的结果

Table 8 Comparison of IGSA and NPSO

FSSP	问题规模		最优解	NPSO(Lian,et al,2008)			IGSA算法		
	n	m		Min	Max	Avg	Min	Max	Avg
Ta001	20	5	1 278	1 278	1 297	1 297.9	1 278	1 278	1 278.0
Ta011	20	10	1 582	1 582	1 639	1 605.8	1 583	1 614	1 600.7
Ta021	20	20	2 297	2 297	2 367	2 334.9	2 297	2 356	2 331.4
Ta031	50	5	2 724	2 724	2 729	2 425.0	2 724	2 724	2 724.0
Ta041	50	10	2 991	3 034	3 129	3 086.9	3 025	3 046	3 032.2
Ta051	50	20	3 771 ~ 3 847	3 938	3 989	3 964.3	3 933	3 952	3 940.7
Ta061	100	5	5 493	5 493	5 495	5 493.0	5 493	5 493	5 493.0

表8表明在求解调度问题时IGSA算法明显优于NPSO算法,从表中可以看出除了Tan011的最小值,IGSA算法不能发现最优解,对于另外的数据集,IGSA算法都能提供比较好的最小值、最大值和平均值.图2显示了不同规模数据的IGSA算法的收敛曲线以及最优解平均值的收敛曲线.从图中看出,IGSA算法能很快地收敛到最优解,说明该算法的收敛性比较好,同时也证明了局部搜索的有效性.



(a) Tan001

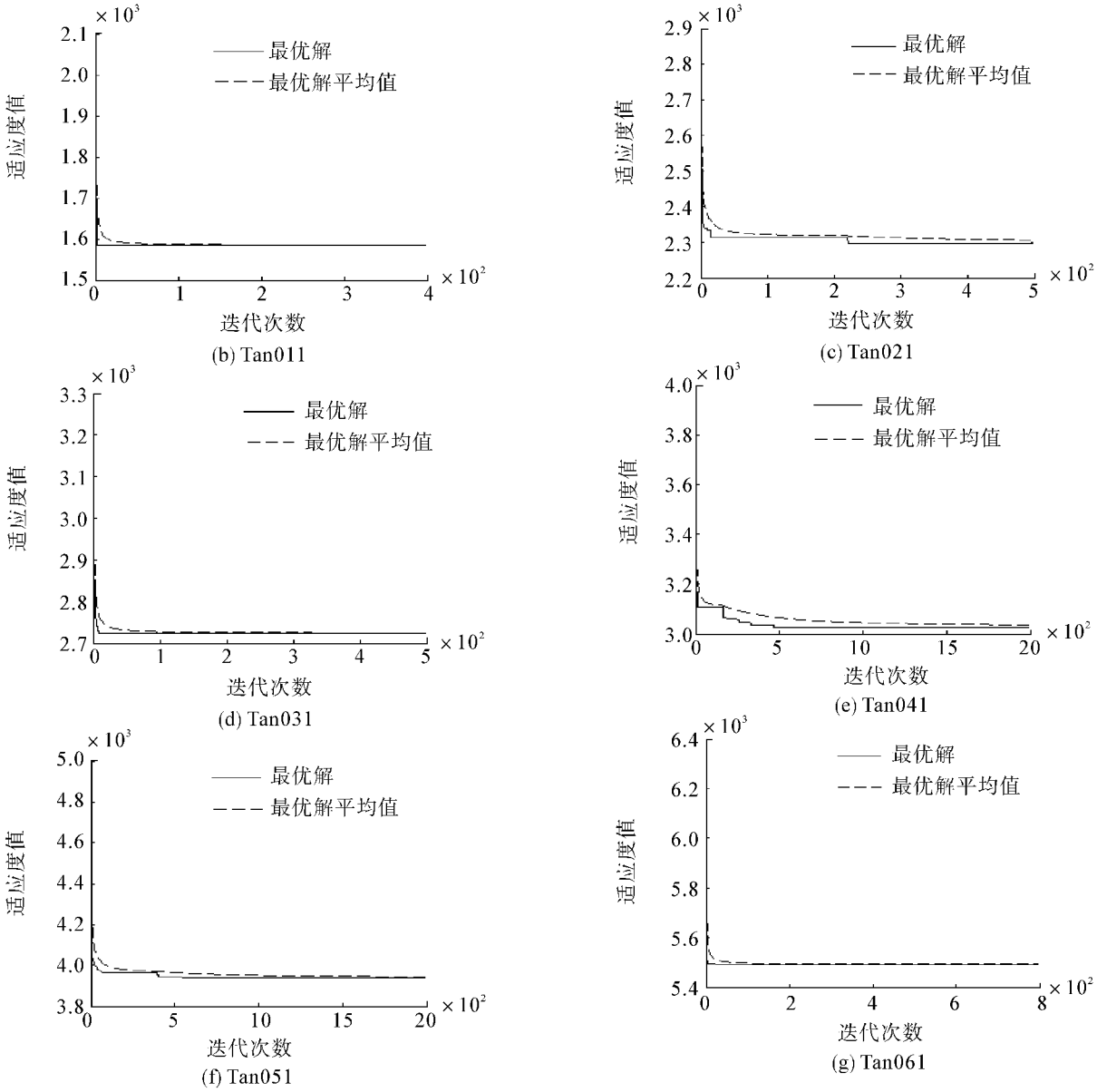


图 2　Trillard 数据集上 IGSA 算法的最优解以及最优解平均值的收敛曲线

Fig. 2　The evolution curve of the optimal solution and the average optimal solution on Trillard using IGSA algorithm

6　结束语

本文工作主要体现在以下方面:

1)提出了一种最大排序规则,此规则运用物体间各个位置分量值的大小次序关系,并且结合随机键编码的方法产生的,将物体的连续位置转变成了一个可行的调度方案. 使得 GSA 算法可以运用到调度问题上.

2)还提出一种边界变异的策略使得越界的物体不再聚集在边界上,而是分布在离边界 $c \times \mathrm{rand}$ 附近的可行空间内,从而增加种群的多样性.

3)为了提高解的质量,提出了一种新的局部搜索策略,结合交换算子和插入算子,从而避免算法陷入局部最优解.

4)接着证明算法的收敛性和算法的空间复杂度和时间复杂度. 并将算法在 21 个不同规模的问题上测试,且与国际中著名的算法进行比较,结果表明不论从算法的稳定性还是从解的质量上都好于其他算法.

下一步工作主要集中在找到较好的局部搜索算法,并将本文的算法运用到其他的调度问题上.

参考文献:

[1] GAREY M R, JOHNSON D S. Computers and intractability: a guide to the theory of NP-completeness [M]. New York, USA: Freeman, 1990.

[2] RINNOOY KAN A H G. Machine scheduling problems: classification, complexity, and computations [M]. The

Hague, The Netherlands: Nijhoff, 1976.

[3]CROCE F D, NARAYAN V, TADEI R. The two-machine total completion time flow shop problem[J]. European Journal of Operational Research, 1996, 90: 227-237.

[4] CROCE F D, GHIRARDI M, TADEI R. An improved branch-and-bound algorithm for the two machine total completion time flow shop problem[J]. European Journal of Operational Research, 2002, 139: 293-301.

[5]PALMER D S. Sequencing jobs through a multistage process in the minimum total time: a quick method of obtaining a near-optimum[J]. Operational Research Quarterly, 1965, 16: 101-107.

[6]GUPTA J N D. Heuristic algorithms for multistage flowshop scheduling problem[J]. AIIE Transactions, 1972, 4: 11-18.

[7]KUO Ihong, HORNG Shijinn, KAO Tzongwann, et al. An efficient flow-shop scheduling algorithm based on a hybrid particle swarm optimization model[J]. Lecture Notes in Computer Science, 2007, 4570: 303-312.

[8]LIAN Zhigang, GU Xingsheng, JIAO Bin. A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan[J]. Chaos, Solitons and Fractals, 2008, 35: 851-861.

[9] RASHEDI E, NEZAMABADI-POUR H, SARYAZDI S. GSA: a gravitational search algorithm[J]. Information Science, 2009, 179(13): 2232-2248.

[10]张长胜,孙吉贵,欧阳丹彤,等. 求解车间调度问题的自适应混合粒子群算法[J]. 计算机学报, 2009, 32(11): 2137-2146.
ZHANG Changsheng, SUN Jigui, OUYANG Dantong, et al. A self-adaptive hybrid particle swarm optimization algorithm for flow shop scheduling problem[J]. Chinese Journal of Computers, 2009, 32(11): 2137-2146.

[11] CARLIER J. Ordonnancements a contraintes disjonctives [J]. RAIRO Recherche Operationelle, 1978, 12: 333-351.

[12]REEVES C R, YAMADA T. Genetic algorithms, path relinking and the flowshop sequencing problem[J]. Evolutionary Computation, 1998, 6(1): 45-60.

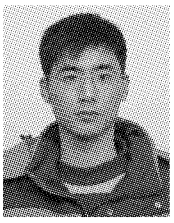
[13]TAILLARD E. Benchmarks for basic scheduling problems [J]. European Journal of Operational Research, 1993, 64: 278-285.

[14]PONNAMBALAM S G, ARAVINDAN P, CHANDRASEKARAN S. Constructive and improvement flow shop scheduling heuristics: an extensive evaluation[J]. Production Planning & Control, 2001, 12(4): 335-344.

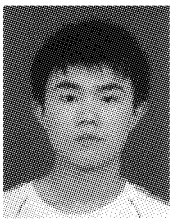
[15]WANG L, ZHENG D Z. An effective hybrid heuristic for flowshop scheduling[J]. International Journal of Advanced Manufacturing Technology, 2003, 21: 38-44.

作者简介:

谷文祥,男,1947 年生,教授、博士生导师,主要研究方向为智能规划与规划识别、形式语言与自动机、模糊数学及其应用. 参加或承担国家自然科学基金、教育部重点项目、省科委项目多项,发表学术论文 100 余篇.



李向涛,男,1987 年生,硕士研究生,主要研究方向为智能规划、智能信息处理.



朱 磊,男,1987 年生,硕士研究生,主要研究方向为智能规划、智能信息处理.