文章编号: 1001-4098(2005)09-0020-04

粒子群算法在柔性工作车间调度中的应用*

谷 峰, 陈华平, 卢冰原, 古春生

(中国科学技术大学 信息管理与决策科学系, 安徽 合肥 230026)

摘 要: 粒子群算法是一种新出现的群智能优化算法。本文针对柔性工作车间调度问题的特点构造了此问题的粒子表达方法, 给出了具体的算法应用过程, 并与遗传算法做了对比实验。实验结果表明粒子群算法在柔性工作车间调度问题的应用上是十分有效的。

关键词: 柔性工作车间调度; 粒子群算法; 遗传算法

中图分类号: TP301 文献标识码: A

1 引言

工作车间调度问题JSP(job shop problem)实际上是一个资源分配问题, 问题的求解目标主要是找到一个将一组资源安排到设备上去, 以使作业可被"最优"完成的方案。通常JSP约束条件很多, 使得JSP成为一个非常难解的组合问题(NP完全问题)。柔性工作车间调度问题由于减少了机器约束, 扩大了可行解的搜索范围, 提高了问题的复杂性, 所以与传统工作车间调度问题相比更加接近实际生产环境的模拟。迄今为止, 已经有很多用于求解柔性工作车间调度问题的最优化方法提出, 如传统的启发式算法、遗传算法、分枝定界法、动态规划法、拉格朗日松弛法和神经网络映射法等。

粒子群算法(Particle Swarm Optimization, PSO)是由Kennedy和Eberhart提出的一种源于对鸟群捕食行为模拟的新的进化计算技术[1]。与遗传算法类似, PSO是一种基于迭代的优化方法, 系统初始化为一组随机解, 通过迭代搜寻最优值。PSO有着个体数目少、计算简单、鲁棒性好等优点, 目前已广泛应用于函数优化、神经网络训练、模糊系统控制等遗传算法应用领域。本文将PSO应用于柔性工作车间调度问题的求解中, 取得了很好的效果。

2 柔性工作车间调度问题描述

柔性工作车间调度问题描述如下: 车间配有m台机器, 要加工n种工件, 每种工件都由n_j个工序组成, 总工序数为$\sum_{j=1}^{n} n_j$. 我们用O_{ij}代表工件J_j上的第i个工序, 见表1。加工过程还需满足以下约束条件:

(1) 同一时刻同一台机器只能加工一个零件;

(2) 每个工件在某一时刻只能在一台机器上加工, 不能中途中断每一个操作;

(3) 同一工件的工序之间有先后约束, 不同工件的工序之间没有先后约束;

(4) 不同工件具有相同的优先级。

表1 柔性工作车间调度加工时间表

		M_1	M_2	M_3	M_4
J_1	O_{11}	1	4	6	2
	O_{21}	4	1	1	3
	O_{31}	3	2	2	3
J_2	O_{12}	1	4	3	3
	O_{22}	4	8	7	1
	O_{32}	7	2	3	2
J_3	O_{13}	3	2	2	7
	O_{23}	9	3	6	1

3 粒子群算法

PSO源于对鸟群捕食行为的研究, 一群鸟在随机搜索食物, 如果这个区域内只有一块食物, 那么找到食物最

* 收稿日期: 2005-06-24
基金项目: 安徽省自然科学基金资助项目(050460404)
作者简介: 谷峰(1978-), 男, 安徽合肥人, 中国科学技术大学信息管理与决策科学系博士研究生, 研究方向: 商务智能。

简单有效的方法就是搜寻目前离食物最近的鸟的周围区域。PSO 就是从这种模型中得到启示并用于解决优化问题。

PSO 中, 每个优化问题的解都是搜索空间中的一只鸟, 称之为"粒子", 每个粒子都有自己的位置和速度(决定飞行的方向和距离), 还有一个由被优化函数决定的适应度值。各个粒子记忆, 追随当前的最优粒子在解空间中搜索。在每一次迭代中, 粒子通过跟踪两个"极值"来更新自己: 一是粒子本身所找到的最优解, 这个解叫做个体极值 $pbest$, 另一个是整个种群目前找到的最优解, 这个解叫做全局极值 $gbest$. 在找到这两个最优值时, 每个粒子根据如下的公式来更新自己的速度和位置[2]。粒子 i 的信息可以用 D 维向量表示, 位置表示为 $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$, 速度为 $V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$, 则速度和位置更新方程为

$$v_{id}(t+1) = w \cdot v_{id}(t) + c_1 \cdot rand_1() \cdot [pbest_{id}(t) - x_{id}(t)] + c_2 \cdot rand_2() \cdot [gbest_d(t) - x_{id}(t)] \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (2)$$

其中, c_1, c_2 为正常数, 称为加速因子, 分别调节向全局最好粒子和个体最好粒子方向飞行的最大步长, 若太小, 则粒子可能远离目标区域, 若太大则会导致突然向目标区域飞去, 或飞过目标区域。合适的 c_1, c_2 可以加快收敛且不易陷入局部最优, 通常取 $c_1 = c_2 = 2$; $rand_1()$ 和 $rand_2()$ 为[0, 1]之间的随机数; w 称惯性因子, w 较大适于对解空间进行大范围探察, w 较小适于进行小范围开挖[3]。式(1)由三部分组成, 第一部分是粒子先前的速度, 说明了粒子目前的状态, 第二部分是认知部分, 表示了粒子本身的思考, 第三部分为社会部分。三个部分共同决定了粒子的空间搜索能力。第一部分起到了平衡全局和局部搜索的能力; 第二部分使粒子有了足够强的全局搜索能力, 避免局部最小; 第三部分体现了粒子间的信息共享。在这三部分的共同作用下粒子才能够有效的到达最好位置。PSO 的程序框架可以描述如下:

```
初始化粒子群;
while (迭代次数< 规定迭代次数) do
  计算每个粒子新位置的适应度;
  对每个粒子, 若粒子的适应度优于原来的个体极
    值pbest, 设置当前适应值为个体极值pbest;
  根据各个粒子的个体极值pbest找出全局极
    值gbest;
  按(1)式更新自己的速度;
  按(2)式更新当前的位置;
endwhile
```

4　面向柔性工作车间调度问题的粒子群算法

4.1　粒子的编码和解码

根据第2节柔性工作车间调度问题的描述, 我们定义总工序数 $L = \sum_{j=1}^{n} n_j$ 并把一个粒子表示为一个 $2L$ 维的向量。为表达和计算方便, 将每个粒子对应的 $2L$ 维的向量分成两个 L 维的向量, 则粒子的位置向量 $X[L]$ 可以表示成 $X_{process}[L]$ 和 $X_{machine}[L]$。$X_{process}[L]$ 由 L 个非0自然数组成, 自然数的顺序决定了工序调度的顺序。我们借鉴遗传算法中基于工序顺序的编码方法[4]给所有同一工件的工序指定相同的符号, 然后根据它们在向量中出现的顺序加以解释。比如说给定一个染色体(1, 2, 2, 3, 3, 1, 1, 2), 染色体中第一次出现的"1"表示工件1的第一道工序, 第二次出现的"1"表示工件1的第二道工序, 第三次出现的"1"表示工件1的第三道工序。$X_{machine}[L]$ 也由 L 个非0的不大于 m (机器总数)的自然数组成, 表示各工序加工时的机器号。

下面, 我们以表1的柔性工作车间调度加工时间表为例来做说明。表1的例子总工序数为8, 所以我们随机构造一个维向量(即粒子)如下:

维数:	1	2	3	4	5	6	7	8
$X_{process}[L]$:	1	3	1	2	2	1	3	2
$X_{machine}[L]$:	1	3	2	1	4	2	4	2

则粒子对应的甘特图见图1。

4.2　位置向量和速度向量计算方法

对每一个粒子, 粒子速度向量 $V[L]$ 可以表示为 $V_{process}[L]$ 和 $V_{machine}[L]$, 按式(1)计算 $V_{process}[L]$ 和 $V_{machine}[L]$, 按式(2)计算 $X_{process}[L]$ 和 $X_{machine}[L]$。由于前面所述的PSO算法为连续空间算法, 而柔性工作车间调度问题为整数规划问题且有其工序先后顺序的约束条件, 所以我们在迭代过程中做了相应的修改, 具体见下:

(1) $X_{machine}[L]$

按式(2)计算出的 $X_{machine}[L]$ 如果为小数, 我们采取向上取整的方法, 其次, 由于机器数是个常数 m, 所以我们定义第 l 维的向量取值区间为[1, m], 当向量超过取值区间时按边界取值。

(2) $X_{process}[L]$

由于柔性工作车间调度问题的工序先后顺序的约束条件, 我们采取以下方法进行 $X_{process}[L]$ 的迭代运算。比如说经过某一次迭代我们得到的结果如下:

维数:	1	2	3	4	5	6	7	8
$X_{process}[L]$:	1	3	1	2	2	1	3	2(初始值)
	3.2	3.9	6.4	5.8	4.3	1.5	3.6	2.4
	(按式(2)计算过后的值)							
$X_{machine}[L]$:	1	3	2	1	4	2	4	2

我们将计算过后 $X_{process}[L]$ 的按数值大小进行排序, 1.5(对应于初始值1) < 2.4(2) < 3.2(1) < 3.6(3) < 3.9

(3)< 4.3(2)< 5.8(2)< 6.4(1),则对应的迭代值为(1, 2, 1, 3, 3, 2, 2, 1),经过这样的转化我们就可以满足工序顺序的约束条件,得到问题的一个可行解。

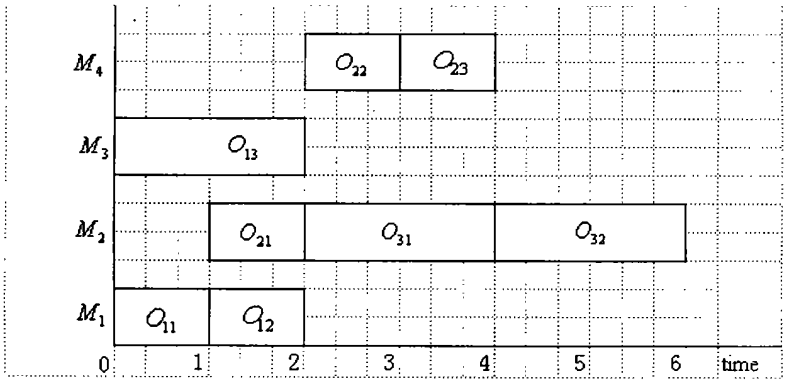

图1 柔性工作车间调度甘特图

4.3 算法流程

根据前面的叙述,我们可以得到面向柔性工作车间调度问题的粒子群算法,算法具体步骤如下:

(1) 初始化粒子群;

(2) 根据适应度函数评价所有粒子,寻找各子群内的最优解$p\ best$和总群体内最优解$gbest$;

(3) 对每一个粒子,按4.2节的方法计算速度向量$V_{process}[L]$,$V_{machine}[L]$和位置向量$X_{process}[L]$,$X_{machine}[L]$;

(4) 若某个粒子的当前适应度优于其历史适应度,则记当前适应度为历史最优适应度,同时记当前位置为该粒子历史最优位置;

(5) 寻找当前各子群内最优解和总群体内最优解,若优于历史最优解则更新$p\ best$和$gbest$;

(6) 迭代次数小于规定迭代次数,转(3);

(7) 结束,输出最优解。

5 实验

随机生成6个柔性工作车间调度实例以最小化最大完工时间C_{max}为优化目标来做仿真实验。为验证算法的有效性,采用Zribi在文献[5]中提出的下界($LowerBounds$)计算方法并与Kacem提出的解决柔性工作车间调度问题的遗传算法[6]做比较。算法用Visual C++实现。实验主要参数如下:遗传算法参数:主群体规模:40;交叉概率为0.9,变异概率为0.01,最大代数取100。PSO参数:粒子群规模:40;$w=0.7$;$c_1=c_2=2$,最大代数取100。两种算法各运行30次,实验结果见表2,表3。

表2 实验结果比较

编号	$n\times m$	总工序数	$LowerBounds$	C_{max}最大值	
				GA	PSO
1	5×5	15	15	16	15
2	10×10	20	18	22	18
3	10×15	30	20	26	20
4	20×10	40	28	32	30
5	20×15	50	36	44	36
6	30×15	60	49	62	52

从表2可以看出,对于所有测试实例,PSO算法的优化结果都明显好于GA,其中在1、2、3、5四个实例中获得了理论上的最优值。

表3的实验结果表明PSO算法达到最优值的成功率都在90%以上,其中1、2、3三个实例成功率为100%,远远高于GA。另外,PSO算法达到最优值的平均代数和平均时间也低于GA,这说明了PSO算法的收敛速度比GA要快。

表3 收敛性能比较

编号	$n \times m$	达到最优值的次数		达到最优值的平均代数		达到最优值的平均时间/s	
		GA	PSO	GA	PSO	GA	PSO
1	5×5	26	30	20.5	10.7	3.6	1.5
2	10×10	22	30	26.8	12.6	4.6	2.1
3	10×15	24	30	28.7	13.5	4.8	2.3
4	20×10	18	28	33.2	17.6	5.4	3.1
5	20×15	21	27	44.3	19.5	6.1	3.4
6	30×15	16	28	60.2	26.7	11.6	4.7

6 结语

本文将粒子群算法应用在柔性工作车间调度问题中, 实验结果表明不论从优化结果还是收敛速度上粒子群算法都要远远优于遗传算法。今后我们将致力于补充和扩展PSO 和其它算法或技术的结合, 解决PSO 容易陷入局部最优的问题。

参考文献:

[1] Kennedy J, Eberhart R C. Particle swarm optimisation[A]. Proc. IEEE International Conference on Neural Networks, IV[C]. Piscataway, NJ: IEEE Service Center, 1995: 1942～1948.

[2] Eberhart R C, Shi Y. Particle swarm optimisation: develoements, applications and resources[A]. Proc. Congress on Evolutionary Computation 2001[C]. Piscataway, NJ: IEEE Press, 2001: 81～86.

[3] Maurice C, Kennedy J. The particle swarm - explosion, stability and convergence in a multidimensional complex space[J]. IEEE Transactions on Evolutionary computation, 2002, 6(1): 58～73.

[4] Qiao B, Sun Z J, Zhu J Y. Solving flexible job shop scheduling problem with genetic algorithm[J]. Transactions of Nanjing University of Aeronautics & Astronautics, 2001, 18(1): 108～113.

[5] Zribi N, Kacem I, El Kamel A, Borne P. Optimization by phases for the flexible job-shop scheduling problem[A]. The 5th Asian Control Conference(vol. 3)[C]. 2004: 1889～1895.

[6] Kacem I. Genetic algorithm for the flexible job-shop scheduling problem[J]. IEEE International Conference on Systems, Man and Cybernetics, 2003, 4: 3464～3469.

Particle Swarm Optimization For Flexible Job Shop Scheduling

GU Feng, CHEN Hua-ping, LU Bing-yuan, GU Chun-sheng

(Department of Information Management and Decision Science,
University of Science and Technology of China, Hefei 230026, China)

Abstract: Particle swarm optimization is a newly emerging method for swarm intelligence optimization. We construct the particle presentation for flexible job shop scheduling according to the characteristic of flexible job shop scheduling and give the detailed process for application. Finally, we compare the genetic algorithm with particle swarm optimisation. The results have shown the effectiveness of particle swarm optimisation for flexible job shop scheduling.

Key words: Flexible Job Shop Scheduling; Particle Swarm Optimization; Genetic Algorithm