文章编号:1001 - 9081(2006)07 - 1694 - 03

基于遗传算法的 Job-Shop 调度问题求解

柳 林
(长沙理工大学 计算机与通信工程学院,湖南 长沙 410076)
(liulin_cscu@yahoo.com.cn)

摘 要:针对 Job-Shop 调度问题,详细讨论了遗传算法以及染色体编码方法,建立了算法模型。通过仿真实验,验证了该算法的有效性。

关键词:遗传算法;Job-Shop 调度问题;模型
中图分类号:TP301.6;TP18 **文献标识码**:A

Genetic-algorithm-based resolution on Job-Shop scheduling problem

LIU Lin
(*School of Computer & Telecommunication Engineering, Changsha University of Science & Technology, Changsha Hunan* 410076, *China*)

Abstract: The genetic algorithm and chromosome coding method for Job-Shop were discussed. The model for the genetic algorithm was constructed. The simulation result shows the effectiveness of this method.

Key words: genetic algorithm; Job-Shop scheduling problem; model

遗传算法(Genetic Algorithms,GA)是模拟生物在自然环境中的遗传和进化过程而形成的一种自适应全局优化概率搜索算法。它将问题的求解表示成染色体的适者生存过程,通过染色体群的一代代不断进化,包括复制、交叉和变异等操作,最终收敛到"最适应环境"的个体,从而求得问题的最优解或满意解[1]。

Job-Shop 调度问题是经典的组合优化问题之一,许多实际工程问题可与之相互转化。Job-Shop 调度问题由于存在众多的约束,是非常难解的 NP 完全问题[2]。如何利用遗传算法高效求解 Job-Shop 调度问题是一个具有挑战意义的难题并成为研究的热点[3]。

1 Job-Shop 调度问题描述

典型的 Job-Shop 调度问题可描述为:n 个工件在 m 台设备上加工,事先规定各工件在各设备上的加工工序,各工件在各设备上的操作时间已知,要求确定各设备上所有工件的加工次序,使某些加工性能指标达到最优。不失一般性,设 n 个工件在 m 台设备上加工(表示为 n/m)满足以下约束:1)每个工件由 m 道工序组成,每道工序在不同的设备上加工;2)每道工序必须在指定的设备上加工;3)按照加工工艺的规定,每道工序必须在它前面的工序加工完毕后再加工;4)每道工序从开始到结束,不会被另外的工序所中断。

则 $n \times m$ 问题可描述如下:

1)工件的集合 $J = \{J_1, J_2, \cdots, J_n\}$;

2)设备的集合 $M = \{M_1, M_2, \cdots, M_m\}$;

3)工件 J_i 的第 j 道工序为 O_{ij},其开始时间为 S_{ij},且满足 $S_{ij} \geqslant 0$;

4)$S_{ij} + w_{ij} - S_{i(j+1)} \leqslant 0$,$w_{ij}$ 为工序 O_{ij} 的加工时间;

5)对于设备 k,若 O_{ij} 在 O_{pq} 之前加工,则满足 $S_{ijk} + w_{ijk} - S_{pqk} \leqslant 0$;

6)一个调度被定义为 $S = (S_1, S_2, \cdots, S_m)$,其中 $S_i = (g_{i1}, g_{i2}, \cdots, g_{in})$ 是第 i 台设备对 n 个工件相应加工工序的加工序列,m 台设备的加工序列组成一个调度;

7)$T_s(a)$ 表示在调度策略 S 下,机器 M_a 上全部工序的完成时间(包括机器的执行时间和等待时间),则调度 S 的完成时间 $T(S) = \max(T_s(1), T_s(2), \cdots, T_s(m))$;

8)基于最短完成时间的 Job-Shop 问题的目标函数为 $\min(T(S), \forall S)$,即寻找一个满足约束的调度 S,使得 $T(S)$ 最小。

2 求解 Job-Shop 问题的遗传算法

将遗传算法应用于 Job-Shop 调度问题中的关键是采用有效的编码方式以及适当的交叉、变异操作。遗传算法对种群重复地进行选择、交叉、变异等基本遗传操作,不断产生出比父代更适应环境的新一代种群,直到满足要求条件为止。

2.1 个体编码

用 GA 求解 Job-Shop 调度问题时,实际是把 Job-Shop 调度问题用 GA 编码加以表示并对各工序的优化排序进行研究。由于同一工件的不同工序被分离为单独的对象来考虑,因此同时出现在不同的设备队列中,由此可能产生单一设备队列合法(不会有两个工件同时占用同一台设备),而不同设备队列间潜伏着死锁的情况[2]。

在 Job-Shop 调度问题的个体编码中,最常用的是直接用工序编号对染色体编码,然而这种编码方案在其后的遗传进化中,会产生表达非法解的个体。对这一问题的处理,很自然的方法是采取拒绝策略,对存在死锁的非法解予以抛弃,只处理合法的个体[4]。然而,对于 Job-Shop 调度问题这样的强约束问题,这种编码方式,合法解在搜索空间中的比率很小,寻找合法解非常困难[5]。

为了解决这个难题,采用基于工序的编码方式,个体由所有工序的排序构成,每个基因代表一个工序,同一工件的所有工序采用同一工件序号表示,然后,根据它们在个体排序中的顺序决定它们在不同机器上的加工顺序。一个 $n \times m$ Job-Shop 调度问题,其个体由 $n \times m$ 个基因组成,每个工件序号只能在个体中出现 m 次,对于某一工件序号,它的第 i 次出现,表示该工件的第 i 道工序($i = 1, 2, \cdots, m$)。基于工序的编

收稿日期:2006 - 01 - 19;**修订日期**:2006 - 03 - 23 **基金项目**:湖南省教育厅科研项目(05C248)
作者简介:柳林(1964 -),女,湖南长沙人,高级工程师,主要研究方向:人工智能.

码方式,其基因有明确的意义。

表 1,表 2 表示了 3×3 Job-Shop 调度问题的一个个体[2 1 1 1 2 2 3 3 3],其中 1 表示工件 J_1,2 和 3 意义相同。染色体中的 3 个 2 表示工件 J_2 的三个工序,第 1 个 2 对应于工件 J_2 的第 1 道工序,以此类推。

表 1　加工时间表

J	M		
	M_1	M_2	M_3
J_1	3	3	2
J_2	1	5	3
J_3	3	2	3

表 2　加工顺序表

J	M		
	M_1	M_2	M_3
J_1	1	2	3
J_2	1	3	2
J_3	2	1	3

所谓解码算法就是根据染色体的编码和 Job-Shop 调度问题的约束条件推导出可行的调度方案(问题的可行解),解码方法如下:

染色体:　2　1　1　1　2　2　3　3　3

↓　　…　　↓

机器:　1　1　2　3　3　2　2　1　3

其含义为:在调度生成过程中,首先安排第 2 个工件的第 1 道工序,接着安排第 1 个工件的第 1 道工序,然后安排第 1 个工件的第 2 道工序,按这种方式,依次从左到右将染色体上的工序都安排完为止。最终生成的调度方案(问题的可行解)为:机器 1 上的加工顺序是 2 -1 -3,机器 2 上的加工顺序是 1 -2 -3,机器 3 上的加工顺序是 1 -2 -3。

2.2　个体的适应度函数

适应度函数 $F(\cdot)$ 的值可以从目标函数 $T(\cdot)$ 转换得到。由于遗传算法中要求适应度函数的取值为非负数,因此可用下式表示:

$$F(S) = f(T(S)) = \sum_{i=1}^{m} \sum_{j=1}^{k_i} w_{ij} - T(S) \qquad (1)$$

调度的过程是一个受各种约束条件制约的动态过程,其完成时间不能由简单的解析式给出,下面给出求调度方案 S 完成时间 $T(s)$ 的算法 1。

算法 1　调度方案完成时间计算

输入　调度方案 S;

输出　调度方案 S 的完成时间 $T(S)$;

```
Begin
  t: =0;
  所有工件的第一道工序置为就绪状态,其余置为等待状态;
  所有机器的状态置为空闲;
  从调度 S 中获得所有机器的加工队列;
  while not (所有机器上的工序都已处理完毕)
  Begin
    for (每台空闲机器)
      if ( Ma 未处理工序且队首工序已就绪)
      Begin
        队首工序的状态改为运行态;
        将 Ma 的状态改为忙;
        time ( a ) = 队首工序的工时;
        从 Ma 的加工队列中取出队首工序;
      End;
      t: = t +1;                                    //时间推进
      for(每台忙的机器 Ma
      Begin
        time( a ): = time( a ) -1;
        if (time ( a ) =0)
        Begin
          将该工序的状态改为完成状态;
          将 Ma 的状态改为空闲;
          若该工序还有后续工序,则将其后续工序的状态改为就绪状态;
        End;
      End;
    End;
    return t;
  End
```

2.3　遗传算子的设计

2.3.1　交叉算子

对于车间作业调度问题如果采用传统的交叉方法,可能会产生表达非法解的个体,通过对前面的编码方案的分析可以发现,如果交叉操作限制在只在表示其工序加工是在同一设备上进行的那些基因(等位设备基因)之间进行,则可以保证在进行交叉操作后子代个体表达的仍是问题的可行解。

交叉操作步骤如下:1)按交叉概率选择参与交叉操作的个体,并将其配成对。2)产生一个随机数,表示参与交叉操作的同设备等位基因所表示工序加工使用的机床编号。3)对两个个体中的同设备等位基因进行交换。

例:设个体 X_1 = 2 1 1 1 2 2 3 3 3,和个体 X_2 = 3 3 1 2 1 1 2 3 2,现在要对表示使用设备 M_1 的同设备等位基因(工序)进行交叉操作。设备 M_1 的同设备等位基因用 ⊙ 表示。交叉操作后产生的子代个体分别为 X'_1 和 X'_2,则有:

X_1 = ② ① 1 1 2 2 3 ③ 3, X'_1 = 3 1 1 1 2 2 3 2 3

X_2 = 3 ③ ① ② 1 1 2 3 2, X'_2 = 3 2 1 3 1 1 2 3 2

2.3.2　变异算子

由于同样的理由,如果变异操作采用传统的方法,经过变异操作后,同样会产生表达非法解的个体。因此只有同设备等位基因之间进行变异操作才有实际意义。

变异操作方法如下:1)依照变异概率选择进行变异操作的个体;2)产生一个随机数,表示变异点基因表达工序加工使用设备;3)分别产生两个随机数,表示参与变异操作的同设备等位基因在个体子串中的位置;4)互换两个变异点基因位的基因,从而形成一个新的子代个体。

2.4　求解 Job-Shop 调度问题的遗传算法

综合上述对编码方法、适应度函数及遗传算子的讨论,下面给出求解 Job-Shop 调度问题的算法 2。

算法 2　求解 Job-Shop 调度问题的遗传算法

输入　加工时间表、加工顺序表、最大进化代数 M、适应度值变化的最小阈值 ε、交叉概率 P_c、变异概率 P_m;

输出　最优(满意)的调度方案;

```
Begin
  根据加工顺序表随机产生 N 个初始个体,组成初始种群(第 1 代) pop(1);
  代数 T: =1;
  上代最佳个体适应值: =0;
  对种群中的所有个体进行解码,调用算法,计算个体的完成时间,找出本代种群中最优个体;
  While not( T > M OR |本代最优个体适应度值 - 上代最优个体适应度值| < ε)
  Begin
    T: = T +1;
    上代最优个体适应值: = 本代最优个体适应值;
    n: =0;
    While ( n = N)
    Begin
      用轮盘赌法从 T 代种群中选择个体作为父代个体,按交叉概率 Pc 进行交叉操作产生新个体;
```

```
        对新个体按变异概率 Pm 进行变异操作;
        将新个体加入 T +1 代种群 Pop( T +1);
        n: = n +1;
      End;
      Pop( T): =Pop( T +1);
      T: = T +1;
    End;
End
```

3 调度问题的遗传算法仿真结果

设有工件 8 个(J_1,J_2,…,J_8),5 台加工设备(M_1,M_2,…,M_5)的生产线作业调度问题,加工时间以及加工顺序分别见表 3,表 4。

表 3 8×5 Job-Shop 调度问题加工时间表

M	J							
	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
M_1	10	18	7.6	17.0	13.6	0	9	6
M_2	9.8	15	1.9	15	10.0	5.4	9.4	9
M_3	11.2	18.5	13	5.6	8	12	34	0
M_4	13	24	20.5	16.4	14.9	14.8	13.6	19.0
M_5	20.8	22	11	30	0	24	20	57

表 3 中,加工时间为 0 表示该工件在该设备上不加工。

表 4 8×5 Job-Shop 调度问题加工顺序表

M	J							
	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
M_1	1	2	1	3	1	5	3	4
M_2	2	1	3	4	2	1	1	1
M_3	3	3	2	5	4	2	2	5
M_4	4	4	5	2	3	4	5	2
M_5	5	5	4	1	5	3	4	3

用本文提出的遗传算法在计算机上仿真实验,$P_c = 0.6$,$P_m = 0.1$,$\varepsilon = 0.1$,群体规模设定为 100,最大进化代数为 100,得出调度方案如表 5 所示。

为了进行比较,使用相同的参数,分别用直接用工序编号的个体编码方法和本文所提的编码方法对上述问题各进行 10 次计算,取平均运算时间和平均最优解,结果如表 6 所示。

表 5 8×5 Job-Shop 调度问题仿真结果表

机器	时间	工件							
		J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
M_1	开始	7.6	31.2	0	91.5	17.6	199.5	190.5	150.5
	结束	17.6	49.2	7.6	108.5	31.2	199.5	199.5	156.5
M_2	开始	57.5	0	41.2	108.5	31.2	52.1	123.5	43.1
52.1	结束	67.3	15	43.1	123.5	41.2	57.5	132.9	
M_3	开始	137.1	69.5	7.6	123.5	129.1	57.5	156.5	156.5
	结束	148.3	88	20.6	129.1	137.1	69.5	190.5	156.5
M_4	开始	148.3	91.5	161.3	75.1	41.2	115.5	219.5	56.1
	结束	161.3	115.5	181.8	91.5	56.1	130.3	233.1	75.1
M_5	开始	161.3	219.5	43.1	0	150.5	69.5	199.5	93.5
	结束	182.1	241.5	54.1	30	150.5	93.5	219.5	150.5

表 6 不同编码方案的计算结果比较

编码方案	平均最优解	平均进化代数	平均算法运行时间/s
直接用工序编号进行编码并采用拒绝策略	242.2	87.2	278.3
本文所提的编码方案	241.7	75.4	29.5

参考文献:

[1] 周明,孙树栋. 遗传算法原理及应用[M]. 北京:国防工业出版社,1999.

[2] 王凌. 智能优化算法及其应用[M]. 北京:清华大学出版社, 2001.

[3] ZHANG H-F, LI X-P, ZHOU P. A job shop oriented virus genetic algorithm[A]. Fifth World Congress on Intelligent Control and Automation[C], 2004,3.2132 -2136.

[4] WU CG, XING XL, LEE HP, *et al*. Genetic algorithm application on the job shop scheduling problem[A]. Proceedings of 2004 International Conference on Machine Learning and Cybernetics[C], 2004,4.2102 -2106.

[5] PANWALKAR SS, ISKANDER W. A Survey of Scheduling Rule [J]. Operation Research, 1997,25(1):45 -61.

(上接第 1693 页)

三种算法的寻优效率。设给定目标解为 425,以达到目标值为停止条件。

表 2 ACS,GAAA,ACSGA 的实验结果

算法名称	平均迭代次数	平均运行时间/s
ACS	60.37	13.92
GAAA	45.53	10.73
ACSGA	28.53	9.01

由表 2 可以看出,30 次实验三种方法平均运行时间分别为 13.92s,10.73s,9.01s,用达到目标值所需时间衡量方法的效率,ACSGA 比 ACS 提升了 35.27%,比 GAAA 也提升了 16.03%。

参考文献:

[1] 王凌. 智能优化算法及其应用[M]. 北京:清华大学出版社, 2001.154 -159.

[2] DORIGO M, GAMBARDELLA LM. Ant Colonies for the Traveling Salesman Problem[J]. BioSystems, 1997, 43(2):73 -81.

[3] MANIEZZO V, DORIGO M, COLORNI A. IRIDIA/94-28, The ant System Applied to the Quadratic Assignment Problem[R]. Belgium: Universite de Bruxelles, 1994.

[4] COLORNI A, DORIGO M, MANIEZZO V, *et al*. Ant System for Job-shop Scheduling[J]. Belgian Journal of Operations Research, Statistics and Computer Science, 1994,34(1):39 -53.

[5] 胡小兵, 黄席樾. 蚁群优化算法及其应用[J]. 计算机仿真, 2004, 24(5):81 -85.

[6] DORIGO M, GAMBARDELLA LM. Ant Colony System: A cooperative learning approach to the traveling salesman problem[J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1):53 -66.

[7] STüTZLE T, HOOS H. The MAX - MIN Ant System and Local Search for the Traveling Salesman Problem[A]. Proceedings of IEEE 4th International Conference on Evolutionary Computation [C]. IEEE Press, 1997.308 -313.

[8] BULLNHERMER B, HARTL RF, STRAUSS C. A new rank-based version of the ant system: a computational study[J]. Central European Journal of Operations Research, 1999, 7(1):25 -38.

[9] 王小平, 曹立明. 遗传算法:理论、应用及软件实现[M]. 西安:西安交通大学出版社, 2002.123 -130.

[10] 丁建立, 陈增强, 袁著祉. 遗传算法与蚂蚁算法的融合[J]. 计算机研究与发展, 2003, 40(9):1351 -1356.