第17卷第2期 209-214 控制理论与应用 Vol.17.No.2
2000年4月 CONTROL THEORY AND APPLICATIONS Apr.,2000

文章编号：1000－8152(2000)02－0209－06

基于遗传算法的单机提前/拖期调度方法研究*

于海斌 薛劲松 王浩波 徐心和
（中国科学院沈阳自动化研究所·沈阳，110015） （东北大学控制与仿真研究中心·沈阳，110006）

摘要：针对对工件有不同交货期要求，并对提前/拖期工件进行惩罚的一类单机调度问题，提出了基于遗传算法的优化方法，提出一种基于"非"一致次序交叉算子的遗传算法，用于排序优化；在分析了惩罚函数性质的基础上，给出了最优开工时间算法，对不同规模的调度问题，应用本文提出的算法与其它算法进行了比较，结果表明该方法具有优良的性能．

关键词：遗传算法；提前/拖期调度；排序优化；开工时间优化

文献标识码：A

GA-Based Approach to Single Machine Scheduling with General Early-Tardy Penalty Weights

YU Haibin, *XUE Jingsong* and *WANG Haobo*
(Shenyang Institute of Automation, Chinese Academy of Sciences·Shenyang, 110015, P.R.China)
XU Xinhe
(Research Center of Control and Simulations, Northeastern University·Shenyang, 110006, P.R.China)

Abstract: It is accordance with Just In Time (JIT) philosophy to penalize early/tardy jobs, A genetic algorithm based optimal method of solving single machine scheduling problem with general early-tardy penalty weights is presented in the paper, which is composed of sequencing optimization and timing optimization algorithms. A new crossover operator is constructed for optimal sequencing search and an effective optimal timing algorithm proposed based on the characteristic analyses of penalty function. For different scale of scheduling problems, a lot of comparative computational experiments were done and the results manifested the method effectiveness.

Key words: genetic algorithm; early-tardy scheduling; sequencing optimization; operational start time optimization

1 引言(Introduction)

提前/拖期惩罚生产调度问题的典型应用背景是JIT生产计划和调度，许多学者都曾研究过提前/拖期惩罚单机生产调度问题．Baker 和 Scudder[1]按交货期要求的不同，交货期的紧张程度和惩罚方式的差别对提前/拖期惩罚调度问题进行了分类．

对工件有已知的不同交货期要求的单机加工，工序间不插入空闲时间的一类调度问题得到了广泛的研究．其中，Held 和 Karp[2]给出了一种动态规划算法；Abdul-Razaq 和 Potts[3]给出了分枝定界算法的下界估算方法；Ow 和 Morton[4]给出的搜索束算法具有较快的搜索速度，但不能保证得到最优解．

实际上，各工件具有不同交货期要求的最优调度是应当允许插入空闲时间的．Fry and Armstrong et al，[5]Garey and Tarjan et al[6]和 Yano and Kim[7]研究了给定排序下的开工时间优化问题，但对惩罚权重有一定的限制．为此，Yano and Kim 提出了一个能够缩减排序空间的启发式算法，并将其应用于惩罚权重与加工时间成正比的单机提前/拖期调度问题．

本文研究已知不同交货期、对提前/拖期惩罚，且惩罚权重不受约束的单机调度问题．本文作者提出了已知加工排序，优化开工时间的一种算法，并给出了相应的最优性证明；同时，在对提前/拖期调度的特性进行分析的基础上，给出了一种基于"非"一致次序交叉算子的遗传算法．对比实验表明，该方法较文[8]的算法更为有效．

* 基金项目：本文得到国家自然科学基金(59990470, 69974039)的支持．
收稿日期：1998－05－20；收修改稿日期：1999－06－15．

2 调度问题建模及其解法策略(Modeling and solution strategy)

2.1 调度问题的数学模型(Model of the scheduling)

本文考虑各工件交货期已知但并不相同,对提前/拖期的惩罚系数允许任意设定,以极小化所有工件的提前/拖期惩罚成本为目标的一类单机生产调度优化问题.

假定工件加工过程不允许中断,所有的工件可以在零时刻投入生产.定义与工件 i 相关的惩罚函数

$$g_i(ts_i)=h_i\cdot\max(0,d_i-ts_i-p_i)+w_i\cdot\max(0,ts_i+p_i-d_i)=$$
$$h_i\cdot\max(0,d_i-te_i)+w_i\cdot\max(0,te_i-d_i). \tag{1}$$

这里

h_i 是对工件 i 提前完工的惩罚系数;

w_i 是对工件 i 拖期完工的惩罚系数;

d_i 是工件 i 的交货期;

ts_i 是工件 i 的开工时间;

te_i 是工件 i 的完工时间;

p_i 是工件 i 的加工时间.

假定共有 n 个待调度工件,形成如下初始调度问题Ⅰ.

调度目标:

$$\min Z=\sum_{i=1}^{n}g_i(ts_i). \tag{2}$$

约束条件:

$$ts_i-ts_j+M(1-Y_{ij})\geqslant p_j,\ \text{对所有的}\ i,j\in[1,2,\cdots,n],i\neq j; \tag{3}$$

$$ts_j-ts_i+M\cdot Y_{ij}\geqslant p_i,\ \text{对所有的}\ i,j\in[1,2,\cdots,n],i\neq j; \tag{4}$$

$$ts_i\geqslant 0,\ \text{对所有的}\ i\in[1,2,\cdots,n]. \tag{5}$$

其中 M 是足够大的正数,0,1 整数变量 Y_{ij} 对应于实际加工问题的工件排序.不难发现调度问题Ⅰ是共有 $n(n-1)/2$ 个约束的混合非线性整数规划问题,每一组 Y_{ij},与一种可行加工排序相对应.

观察调度问题Ⅰ可以发现,调度方案由 Y_{ij},t_i 和 t_j 唯一确定.对这类整数规划问题,由于其典型的 NP 特性(可行排序共有 $n!$ 种),无法找到有效的多项式算法;即使在已知 Y_{ij} 的情况下,由于目标函数是关于工件开工时间 ts_i 的非正则(non-regular)指标[9],开工时间也需要进一步确定.

2.2 解法策略(Solution strategy)

将上述混合整数规划问题分解为两个层次,最优 0,1 变量 Y_{ij} 和最优开工时间 ts_i 的确定.此时,在 Y_{ij} 确定的条件下,调度问题Ⅰ退化为非线性规划问题,称为调度问题Ⅱ.

调度目标: $\min Z=\sum_{i=1}^{n}g_i(ts_i)$.

约束条件: $ts_i-ts_j\geqslant p_j$, 对所有的 $i,j\in[1,2,\cdots,n]$, $Q(i)=Q(j)+1$; (6)

$ts_i\geqslant 0$,对所有的 $i\in[1,2,\cdots,n]$. (7)

其中,$Q(i)$ 为工件 i 在给定的 Y_{ij} 排序下的加工顺序号.

显然,调度问题Ⅱ是有 $n-1$ 个约束的非线性规划问题,较之问题Ⅰ的规模大幅度降低.

3 最优开工时间算法(Algorithm for optimizing job's starting time)

3.1 目标函数 Z 的性质分析(Characteristic analysis of target function Z)

定义 1(连续工件集) 一组连续调度的 n 个工件 $1,2,\cdots,n$,若 $te_k=ts_{k+1},k\in[1,2,\cdots,n]$,称为连续工件集.若连续工件集满足 $te_1=d_1$,称为初始位置连续工件集.

显然,任意调度方案均由一些连续工件集唯一确定.

定义 2(排序函数) n 个工件 $1,2,\cdots,n$,其相应的某种指标测度为实数 $c_1,c_2,\cdots,c_n$.函数 $S:[1,2,\cdots,n]\to[1,2,\cdots,n]$ 称为排序函数,若

1) $c_{s(i)}<c_{s(j)}$,当 $i<j$;

2) $S(1)=\arg\{\min[c_1,c_2,\cdots,c_n]\}$;

3) $S(n)=\arg\{\max[cx_1,c_2,\cdots,c_n]\}$.

若测度指标为交货期,则 $S(i)$ 表示排序为 i 的工件的编号.

考虑连续工件集 B_j 的移动成本

$$Z_j(x)=\sum_{i\in B_j}g_i(ts_i), \tag{8}$$

其中,x 是 B_j 相对于初始位置的偏移,即: $x=te_1-d_1$.如果 n 个待调度工件组成 m 个连续工件集 $B_j,j\in[1,2,\cdots,m]$,$m\leqslant n$,显然调度目标函数 $Z(x)$ 为 $Z_j(x)$ 的加和.

命题 1　连续工件集 B_j 的移动成本函数 $Z_j(x)$ 是分段线性凸函数(证明略).

命题 2　对连续工件集 B_j 的移动成本函数 $Z_j(x)$, $x \in (-\infty, +\infty)$, $\exists$ 存在 $i \in [1,2,\cdots,n]$, 当且仅当 $te_i = d_i$, $Z_j(x)$ 达到极小(证明略).

3.2　开工时间优化算法(Algorithm for optimizing *job's starting time*)

根据上节分析,对单独的连续工件集,只要从一侧搜索各分段线性函数的折点,就可以找到确定加工顺序下的最优开工时间.问题在于命题 2 的成立是有条件的,必须保证 B_j 是一连续工件集.而 3.1 节并没有针对确定排序的 n 个工件,给出如何分组构成连续工件集才能确保得到最优调度的方法.

针对以上分析,给出开工时间优化算法 A,见附录 A.此算法通过迭代,自动得到最优连续工件集以及工件的开工时间.

命题 3　有确定加工顺序的 n 个工件,在单机上加工,则应用开工时间优化算法 A,可以得到使指标 Z 最小的优化调度(证明见附录 B).

4　加工排序优化的遗传算法(GA for sequence optimization)

针对提前/拖期调度优化的遗传算法结构(见图 1),下面就编码、初始群体生成、选择、交叉、变异、适应度标定等具体环节分别予以阐述.

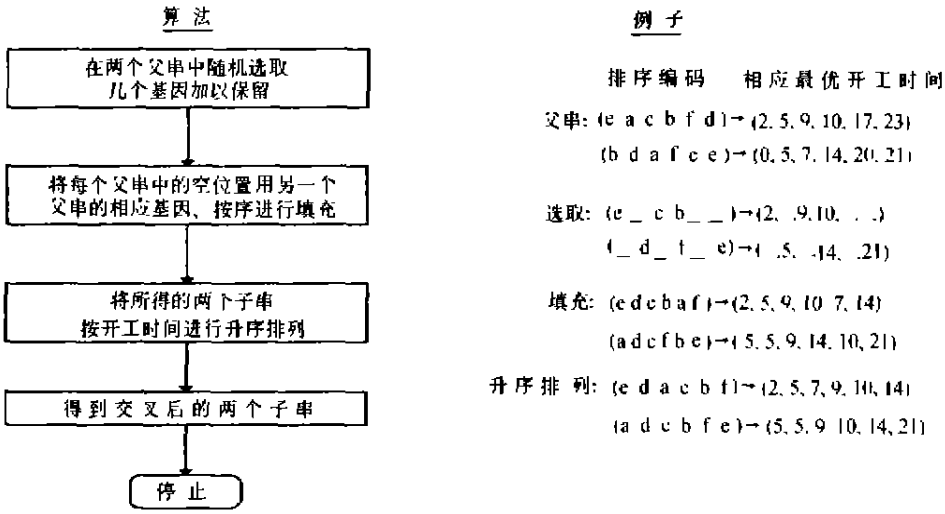

图 1　"非"一致次序交叉算子

Fig. 1　Non-consistent crossover operator based on sequence

1) 编码.

本文采用排列排序(permutation)编码,如某些待加工的工件 $J_1, J_2, \cdots, J_6$ 的一个加工序列 $J_1, J_6, J_4, J_2, J_5, J_3$ 相应的编码为[1,6,4,2,5,3].

2) 初始化.

本文的初始群体由两部分组成:按 EDD(Earliest Due Date)和 EST(Earliest Starting Time)规则生成的群体和按均匀等概率随机生成的群体.

3) 遗传算子.

遗传算法对于新的可行空间的探索,依赖于交叉和变异操作的效果.其中,交叉操作的有效性是以模块组装假设[10]成立的条件为前提的,基于排列排序编码的交叉算子,已经提出多种[11].常见的有 PMX(Partial Mapped Crossover), OX(Order Crossover), OX2 (Order Crossover 2), PBX (Position Based Crossover), CX (Cycle Crossover) 和 UX (Union Crossover)等.文[11]针对 TSP 问题,研究了上述各种交叉算子的应用效果,结果表明,OX, CX 和 UX 具有较高的搜索效率.

本文基于对提前/拖期调度问题的认识和分析,提出如下基于位置编码和加工时间信息的"非"一致次序交叉算子.

变异采用位置交换(Swap)算子;适应度标定应用排序(Ranking)准则[10,11].

5　计算结果及分析(Computational results and analyses)

本文所提的单机提前/拖期调度算法已在 PC/486/100 上用 C 语言实现,并针对不同规模工件加工,随机生成的调度问题进行了应用实验研究.当某代的平均性能较上一代的改进少于 0.01% 时,GA 搜索过程停止.

表 1 和表 2 分别为 50,80 个工件不同规模调度问题的计算结果.

表 1 50 个不同交货期的工件

Table 1 50 jobs with different due dates

选择策略	交叉	Z 最小值	停止代数	时间(秒)	适应值总和
回转轮	SGA	5141	157	192	1248681
回转轮	CX	6203	196	199	1085679
回转轮	OX	7680	235	267	987653
回转轮	UX	7864	208	221	897631
回转轮	UOX	7002	291	393	983188
回转轮	"非"一致次序交叉算子	2375	119	162	1677015
精英选择	SGA	2405	137	147	1568700
精英选择	CX	4756	144	152	1356309
精英选择	OX	6201	108	116	1152890
精英选择	UX	7074	133	136	1078654
精英选择	UOX	6297	107	141	1102212
精英选择	"非"一致次序交叉算子	2052	120	165	1679775

计算适应度的方法是用 20000 减去 Z 最小值.

表 2 80 个不同交货期的工件

Table 2 80 jobs with different due dates

选择策略	交叉	Z 最小值	停止代数	时间(秒)	适应值总和
回转轮	SGA	24758	29	41	3665631
回转轮	CX	21865	56	43	4352043
回转轮	OX	27859	72	66	3882476
回转轮	UX	29843	49	58	3269053
回转轮	UOX	24912	89	154	3769958
回转轮	"非"一致次序交叉算子	5103	18	21	6267344
精英选择	SGA	19021	21	23	4287059
精英选择	CX	22074	43	51	4005421
精英选择	OX	16853	82	77	5006731
精英选择	UX	27852	52	50	3452789
精英选择	UOX	24877	99	170	4033768
精英选择	"非"一致次序交叉算子	5558	13	16	6125740

计算适应度的方法是用 70000 减去 Z 最小值.

表 1 和表 2 中的实验结果是在表中指定的"选择"和"交叉"条件下,各自算法的最优搜索结果.

通过观察上述结果,可以得出如下结论:

1) 本文所提出的"非"一致次交叉算子总能在较少的代数内或占用较少的 CPU 时间,找到最好的调度方案,而且整个染色体种群的平均性能也最好.

2) 交叉算子对遗传算法的搜索效率起最终决定性作用.

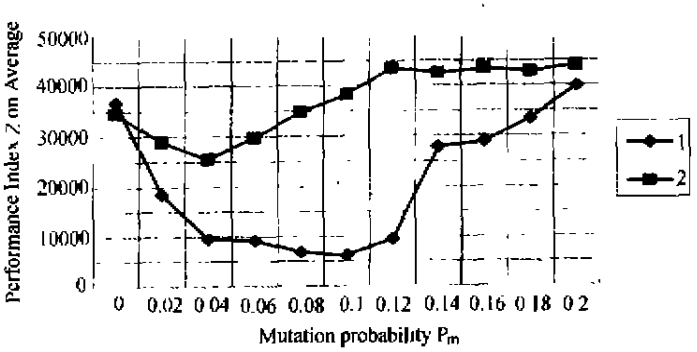

图 2 变异概率与性能指标间的关系

Fig. 2 Performance vs. mutation probability

在实际应用过程中，基于"非"一致次序交叉算子的遗传算法要取得好的优化效果，需要设定较高的变异概率，并且对变异概率更为敏感，见图2.

图2是针对100个随机产生的80个工件调度的实例得到的平均性能曲线，曲线1是应用"非"一致次序交叉算子，而曲线2是用UOX的GA算法的性能. 产生这种现象的原因是由于"非"一致次序交叉算子有良好的搜索性能，若变异概率过小，容易陷入局部最小、发生早熟现象.

6 结论(Conclusion)

本文针对不同交货期要求，对提前/拖期进行惩罚的一类单机加工问题，提出了基于遗传算法的调度优化方法. 根据分析和实验考察，本文所提的"非"一致次序交叉算子对各类提前/拖期排序问题都有较好的搜索性能.

参考文献(References)

[1] Baker K R and Scudder G D. Sequencing with earliness and tardiness penalities: a review[J]. Ops. Res., 1990, 38(1): 22 - 36

[2] Held M and Karp R M. A dynamic programming approach to sequencing problems[J]. J. Soc Ind. Appl. Math., 1962, 10(1): 196 - 210

[3] Abdul-Razaq T S and Potts C N. Dynamic programming state-space relaxation for single-machine scheduling[J]. J. Ops. Res. Soc., 1988, 36(1): 141 - 152

[4] Ow P S and Morton T E. The single machine early/tardy problem[J]. Mgmt Sci, 1989, 35(2): 177 - 191

[5] Fry T D, Armstrong R D and Blackstone J H. Minimizing weighted absolute deviation in single machine scheduling[J]. IIE Trans, 1987, 19(4): 445 - 450

[6] Garey M, Tarjan R E and Wilfong G T. One-processor scheduling with symmetric earliness and tardiness penalities[J]. Maths. Ops. Res., 1988, 13(2): 330 - 348

[7] Yano C A and Kim Y D. Algorithms for a class of single-machine weighted tardiness and earliness problems[J]. Eur. J. Ops Res., 1991, 52(2): 167 - 178

[8] Lee C Y and Choi J Y. A genetic algorithm for job sequencing problems with distinct due dates and general early-tardy penalty weights[J]. Computers Ops. Res., 1995, 22(8): 857 - 869

[9] 陈荣秋编著. 排序的理论与方法[M]. 武汉：华中理工大学出版社，1987

[10] Goldberg D E. Genetic Algorithms in Search, Optimization, and Machine Learning[M]. Reading, MA: Addison-Wesley, 1989

[11] Poon P W and Carter J N. Genetic algorithm crossover operators for ordering applications[J]. Computers Ops. Res., 1995, 22(1): 135 - 147

附录 A(Appendix A)

开工时间优化算法 A:

groupno: = first(1): = last(1): = 1;

ts_1: = max(0, $d_1 - p_1$); te_1: = max(d_1, p_1); B_1 = {1}; // 初始化第一个工件

for i: = 2 to n do

  if $te_{i-1} + p_i < d_i$ then // 工件 $i-1$ 不影响工件 i 的最优位置

    groupno: = groupno + 1;

    first(groupno): last(groupno): = i;

    ts_i: = $d_i - p_i$; te_i = d_i; $B_{groupno}$: = {i} // 置工件 i 于最优位置上

  else if $te_{i-1} + p_i = d_i$ then // 工件 $i-1$ 不影响工件 i 的最优位置

    last(groupno): = i; // 将工件 i 纳入工件 $i-1$ 所在的连续工件组

    ts_i: = te_{i-1}; te_i: = d_i; $B_{groupno}$: = $B_{groupno} \cup \{i\}$

  else if $te_{i-1} + p_i > d_i$ then // 工件 $i-1$ 影响工件 i 的最优位置

    last(groupno): = i; // 将工件 i 纳入工件 $i-1$ 所在的连续工件组

    ts_i: = te_{i-1}; te_i: = $ts_i + p_i$; $B_{groupno}$: = $B_{groupno} \cup \{i\}$;

    do $Z_0(B_{groupno}) = \sum_{i \in B_{groupno}} g_i(ts_i)$;

    step: = min($x = te_i - d_i \mid x > 0, i \in B_{groupno}$);

    if ($ts_{first(groupno)}$ - step < 0)

      step: = $ts_{first(groupno)}$;

    else if ($ts_{first(groupno)}$ - step < $te_{(last(groupno-1))}$);

      step: = $te_{(last(groupno-1)} - ts_{first(groupno)}$);

    for $i \in B_{groupno}$ // 调整包括工件 i 的新的连续工件组的位置

      do te_i: = te_i - step; ts_i: = ts_i - step;

    end for

```
Z_1(B_groupno): = Σ_{i∈B_groupno} g_i(ts_i);        // 测试连续工件组的性能
if (Z_1(B_groupno) <= Z_0(B_groupno))
  {if (ts_first(groupno) = 0) break;                // 连续工件组在调整后的位置上更优
  if (ts_first(groupno) = te_last(groupno-1))
    {groupno: = groupno - 1;
    last(groupno): = last(groupno + 1);
    B_groupno: = B_groupno ∪ B_groupno-1
    }
  }
else                                                // 连续工件组在调整前的位置上更优
  {for i ∈ B_groupno
  do te_i: = te_i + step; ts_i: = ts_i + step;
  end for
  break;
  }
}
end do
    }
end for
end
```

附录 B(Appendix B)

命题 3 的证明：

证　用数学归纳法证明.

$n = 1$ 时，命题显然成立.

设 $n = k$ 时，命题成立，即 n 个工件由 p 个连续工件集 $B_j, j \in (1,2,\cdots,p)$ 组成.

$\forall B_j, j \in (1,2,\cdots,p)$，不失一般性，设 B_j 由 m 个工件组成，其中工件 i 用 b_{ji} 标识.

$\forall h \in (1,2,\cdots,m)$，$B_j$ 可以分成两个相邻连续工件集 $B_{j1} = b_{j1},\cdots,b_{jh}$；$B_{j2} = b_{jh+1},\cdots,b_{jm}$.

根据命题假定，B_j 处于最优位置. 若 $ts_{\text{first}(B_{j1})} > 0$ 成立，应用反证法容易证得：

$$Z(B_{j1} - \Delta) > Z(B_{j1}) > Z(B_{j1} + \Delta), \qquad (B1)$$

$$Z(B_{j2} + \Delta) > Z(B_{j2}) > Z(B_{j2} - \Delta). \qquad (B2)$$

其中，$B_j - \Delta$ 代表 B_j 连续工件集内所有工件左移 Δ 时间；$B_j + \Delta$ 代表 B_j 连续工件集内所有工件右移 Δ 时间，Δ 是大于零的正数.

当 $n = k + 1$ 时，根据算法 A 可知，若 $te_k + p_{k+1} \leqslant d_{k+1}$，置 $te_{k+1} = d_{k+1}$.

显然，$\sum_{i=1}^{k} g_i(ts_i) = \sum_{i=1}^{k+1} g_i(ts_i)$.

因为 $\forall i \in [1,2,\cdots,m]$，$g_i(ts_i) \geqslant 0$，所以 $n = k + 1$ 时命题成立.

若 $te_k + p_{k+1} \geqslant d_{k+1}$，算法 A 首先将工件 $k+1$ 与 B_m 合并，然后不断地向左搜索、合并，搜索直至达到最小点.

由于篇幅的限制，此处只给出命题成立的原则性说明. 由式(B1)和(B2)可知，任意模块只有向左调整，调度性能指标 Z 才会改善.

对算法所处理的各种情况进行考察，结合命题 1 和命题 2 的结果，容易得出 $n = k + 1$ 时命题依然成立的结论.

证毕.

本文作者简介

于海斌　1964 年生. 1987 年、1997 年东北大学自动控制系分别获硕士、博士学位. 现为中国科学院沈阳自动化研究所研究员，机电控制及自动化专业博士生导师. 感兴趣的研究领域为智能生产调度，分布式控制系统，离散事件动态系统等.

薛劲松　1938 年生. 1964 年清华大学自动化系毕业，1968 年中国科学院自适应控制专业研究生毕业. 现为中国科学院沈阳自动化研究所研究员，机电控制及自动化专业博士生导师. 感兴趣的研究领域为 CIMS 总体设计方法、CIMS 系统分析和优化技术等.

王浩波　1972 年生，于中国科学院沈阳自动化研究所硕士毕业，研究兴趣为人工智能及软件设计开发.

徐心和　见本刊 2000 年第 1 期第 8 页