基于多层遗传算法的柔性 Job Shop 调度问题的研究*

朱 颢,曾益坤
(湖州职业技术学院,浙江 湖州 313000)

摘 要:介绍了柔性 Job Shop 调度问题的模型,并针对三级子问题,分别设计了相应的遗传算法,给出了其流程。通过实例证明,该算法的设计是行之有效的。

关键词:柔性 Job Shop;遗传算法
中图分类号:TP302 **文献标识码**:A

Research on Flexible Job Shop scheduling based on multi-layer genetic algorithm

ZHU Hao, ZENG Yi Kun
(Huzhou Vocational Technology College, Huzhou 313000, China)

Abstract: In this paper, a kind of flexible Job Shop is studied. At first, the model of the flexible Job Shop is promoted. Then three genetic algorithms are applied to three sub-problems of the flexible Job Shop, and the process is promoted. Lastly, an example is given and it can prove that the intelligent optimization algorithm is effective.

Key words: Flexible Job Shop; genetic algorithm

随着计算机技术和自动化技术的发展以及 JIT 思想在企业的运用,现代制造技术发生了很大的变化,产品的制造过程变得越来越具有柔性,特别是在计算机集成制造出现之后,工件加工工艺路线必须唯一确定的限制已被打破,工件的柔性制造成为生产的实际需求。因此,研究具有柔性特征的调度问题具有重要的理论和实际意义。SCHRICH C R 等人[1]考虑了一类目标函数为极小化拖期时间的柔性调度问题,并采用两层禁忌搜索算法来进行优化。BAYKASOGLU A[2]针对柔性 Job Shop 调度问题,采用启发式规则和模拟退火算法寻优,并给出了相应的实例。KACEM I[3]则对于柔性 Job Shop 调度问题的两级子问题,分别采用局部搜索和遗传算法进行寻优。TIENTE H[4]采用进化算法求解了多目标的柔性 Job Shop 调度问题,在变异的过程中,采用了一些启发式方法使得变异朝着优良解的方向进行。陈伟达、达庆利[5]针对工艺路线可变的 Job Shop 问题,提出了求解该问题的两层遗传算法。上述研究所考虑的柔性 Job Shop 问题的特点为:每个工件只有 1 条固定的工艺路线,工件所包含的工序能够在多台机器上进行加工。本文对上述文献考虑的柔性 Job Shop 问题进行了扩展,考虑每个工件的工艺路线可以有多条,工件在每条工艺路线下,其包含的工序不同。

1 柔性 Job Shop 调度问题的模型

为方便起见,先给出如下符号:

J 表示加工环境中工件的集合,J 中共有 n 个工件,$i=1,\cdots,n$,表示第 i 个工件。

M 表示加工环境中机器的集合,M 中共有 m 台机器,$k=1,\cdots,m$ 表示第 k 台机器。

P 表示加工环境中工艺路线的集合,工件 i 共有 P_i 条工艺路线,$p=1,\cdots,P_i$ 表示工件 i 的第 p 条工艺路线。

JO 表示加工环境中所有工序编号的集合,$JO=\{1,\cdots,q\}$,q 表示共有 q 个工序,$j\in JO$ 表示编号为 j 的工序。

$M_{allow}(j)$表示可以加工工序 j 的机器集合,$M_{allow}(j)\subset M$。

$OP(i,p)$表示工件 i 采用工艺路线 p 时,其包含的工序集合,$OP(i,p)\subset JO$。

* 基金项目:湖州市自然科学基金(2008YZ10)

$$x_{ip}=\begin{cases}1 & \text{工件 } i \text{ 采用工艺路线 } p\\ 0 & \text{其他}\end{cases}$$

$$y_j(i,p)=\begin{cases}1 & \text{工件 } i \text{ 采用工艺路线 } p \text{ 时,包含工序 } j\\ 0 & \text{其他}\end{cases}$$

$$z_{jk}(i,p)=\begin{cases}1 & \text{工件 } i \text{ 采用工艺路线 } p \text{ 时,}\\ & \text{其工序 } j \text{ 选择机器 } k \text{ 加工}\\ 0 & \text{其他}\end{cases}$$

$$w_{jk}(r,p1)_{lk}(s,p2)=\begin{cases}1 & \text{工件 } r \text{ 采用工艺路线 } p1 \text{ 时,}\\ & \text{其工序 } j \text{ 在机器 } k \text{ 加工领先于}\\ & \text{工件 } s \text{,采用工艺路线 } p2 \text{ 时,}\\ & \text{其工序 } l \text{ 在机器 } k \text{ 上加工}\\ 0 & \text{其他}\end{cases}$$

$pt_{jk}(i,p)$表示工件 i 采用工艺路线 p 时,其工序 j 在机器 k 上加工的加工时间。

$st_{jk}(i,p)$表示工件 i 采用工艺路线 p 时,其工序 j 在机器 k 上加工的开始加工时间。

$et_{jk}(i,p)$表示工件 i 采用工艺路线 p 时,其工序 j 在机器 k 上加工的加工结束时间,$et_{jk}(i,p)=st_{jk}(i,p)+pt_{jk}(i,p)$。

et_i 表示工件 i 的加工完成时间:

$$et_i=\sum_{p=1}^{P_i}x_{ip}\left\{\max_{j\in OP(i,p)}\left\{\sum_{k\in M_{allow}(j)}et_{jk}(i,p)Z_{jk}(i,p)\right\}\right\} \quad (1)$$

问题的目标函数为:

$$C_{\max}=\max_{i\in J}(et_i) \quad (2)$$

问题的决策目标为:选择工件的工艺路线、工序所需的加工机器,排定各工序在相应机器上的加工顺序使得 $C_{\max}$ 最小。

约束条件为(3)~(7)式:

$$st_{(j+1)k}(i,p)\geqslant et_{jt}(i,p) \quad (3)$$

其中,$i\in J,p=1,\cdots,P_i,j\in OP(i,p),k\in M_{allow}(j+1),t\in M_{allow}(j)$

$$st_{lk}(s,p2)+N(1-W_{jk}(r,p1)_{lk}(s,p2))\geqslant et_{jk}(r,p1) \quad (4)$$

其中,$s,r\in J,p1=1,\cdots,P_r,p2=1,\cdots,P_s,j\in OP(r,p1)$

$l\in OP(s,p2),k\in M_{allow}(j)\cap M_{allow}(l)$

$$st_{jk}(r,p1)+N\times W_{jk}(r,p1)_{lk}(s,p2))\geqslant et_{lk}(s,p2) \quad (5)$$

其中,$s,r\in J,p1=1,\cdots,P_r,p2=1,\cdots,P_s,j\in OP(r,p1)$

$l\in OP(s,p2),k\in M_{allow}(j)\cap M_{allow}(l)$

式(4)和式(5)中的 N 为一个足够大的正数。

$$\sum_{p=1}^{P_i}x_{ip}=1\quad i\in J \quad (6)$$

$$\sum_{k\in M_{allow}(j)}Z_{jk}(i,p)=1\quad i\in J,p=1,\cdots,P_i,j\in OP(i,p) \quad (7)$$

式(3)表明了同一工件的 2 个连续的工序之间的关系,即只有前面的工序加工完成之后,其后续工序才能开始加工。式(4)和式(5)表示在同一机器上不能同时加工 2 个工序,只有 1 个工序加工完,才能加工另外的工序。式(6)表明每个工件只能选择其中的 1 条工艺路线。式(7)表明每个工件在确定工艺路线之后,其包含的每一个工序只能选择 1 台机器进行加工。

此类柔性的 Job Shop 调度问题可以分解为 3 个子问题:(1):为每个工件选择 1 条工艺路线,称为工艺路线子问题。(2):在(1)的基础上为每个工件所包含的工序选定相应的加工机器,称为加工机器子问题。(3):在(2)的基础上完成每个机器排定工序的加工顺序,称为排序子问题,即属于传统的 Job Shop 调度问题。

对于这样一个递阶的优化问题,本文将采用 3 层遗传算法分别来进行优化。

2 柔性 Job Shop 调度问题的算法介绍

2.1 工艺路线子问题的遗传算法

设工艺路线染色体种群为 *chrom_plan*,种群规模为 *popsize_plan*。

(1)编码:本层次遗传算法采用基于整数的编码,从左至右每个基因分别表示工件 1,2,…,n 的工艺路线号。

(2)解码:对于每个染色体 *chrom_plan*[i],其解码过程将要调用加工机器子问题的遗传算法,以求得 *chrom_plan*[i]所对应的目标函数值 $C_{\max}(i)$。

(3)适应度函数:采用 $f_i=\alpha/C_{\max}(i)$。α 为正常数,$C_{\max}(i)=C^{machine}_{\max\_best}$,$C^{machine}_{\max\_best}$ 为后面所述加工机器子问题遗传算法所求得的最优目标函数值。

(4)选择:采用赌轮选择。

(5)交叉:根据本层次染色体的特性,采用两点交叉的方式。

(6)变异:随机选出某一个基因,从该基因所属工件的工艺路线集合中随机地选出另外的一条不同的工艺路线,代替掉原来的工艺路线。

2.2 加工机器子问题的遗传算法

根据上述文中解码部分可知,对于每个工艺路线染色体 *chrom_plan*[i],都可以生成不同的加工机器矩阵,而每个加工机器矩阵会使得最后的目标函数值不同,因此,本层次的遗传算法的目的是为每个工艺路线染色体 *chrom_plan*[i]寻找到最优的加工机器矩阵。设加工机器矩阵染色体种群为 *chrom_machine*,种群规模为 *popsize_machine*。

(1)编码:当工件确定其工艺路线后,根据工艺路线确定出每个工件所包含的工序,并生成工序矩阵,对于不包含的工序可以用 0 代替。

由于每个工序可以选择的加工机器是多样的,因此,可以根据工序矩阵来随机地生成相应的加工机器矩阵染色体种群,且加工机器矩阵与工序矩阵同维数,生成加工机器矩阵染色体的原则为:

对于工序矩阵中为 0 的位置,在加工机器矩阵同位置也用 0 表示,表示此虚工序 0 不在任何机器上加工。对于工序矩阵中非 0 的其他工序,从各工序对应的可选加工机器集合中随机地选出 1 台机器,放入加工机器矩阵中相同的位置,这样就生成了1 个加工机器

矩阵染色体。

对于每个工艺路线染色体 $chrom\_plan[i]$，都可随机地生成加工机器矩阵染色体种群 $chrom\_machine$。

(2)解码：对于每个加工机器矩阵染色体 $chrom\_machine[j]$，可以根据相应工序对应时间表生成每个工序在相应的机器上的加工时间矩阵 ***pt***，然后调用下一级子问题(排序子问题)的遗传算法，求得该 $chrom\_machine[j]$ 所对应的目标函数值 $C_{\max}^{machine}(j)$，且 $C_{\max}^{machine}(j)=C_{\max\_best}^{schedule}$，$C_{\max\_best}^{schedule}$。本层次遗传算法所求得的最优目标函数值为 $C_{\max\_best}^{machine}$。

(3)适应度函数：本层次遗传算法的适应度函数采用 $f_j=\alpha/C_{\max}^{machine}(j)$。

(4)选择：采用赌轮选择。

(5)交叉：随机地选出 1 个工件，将 2 个加工机器矩阵中该工件所对应的行交换，即完成交叉操作。

(6)变异：对于加工机器矩阵，随机地选出其中的 1 个非零机器元素，找出此非零元素所对应的工件和工序，从该工序的可选机器集合中随机选择 1 个与当前不同的机器，替代当前机器。

2.3 排序子问题的遗传算法

在工件确定了其工艺路线和加工机器矩阵之后，还需要排定在各机器上加工的工序的顺序，这属于排序子问题的范畴，排序子问题即传统的 Job Shop 调度问题。目标是为每个加工机器矩阵染色体 $chrom\_machine[j]$，找到在各机器上加工的工序排序，使得目标函数值最小。

设本层次遗传算法的染色体种群为 $chrom\_machine$，规模为 $popsize\_schedule$。

(1)编码：本层次遗传算法中染色体采用基于工序的编码方法，再将工序用该工件的工件号代替。

(2)解码：本层次遗传算法解码的过程即为将染色体中各个基因表示的工序以一定的规则安排在相应的机器上的过程。具体方法见参考文献[5]。

(3)选择：适应度函数采用 $f_s=\alpha/C_{\max}^{schedule}(s)$。$C_{\max}^{schedule}(s)$ 染色体 $chrom\_schedule[s]$ 所对应的目标函数值，$C_{\max\_best}^{schedule}$ 为经过本层次遗传算法优化后的最优目标函数值。

(4)交叉：为了扩大搜索的范围，本节采用了单性交叉和双性交叉 2 种算子。在单性交叉中，从同一个染色体随机地选出的 2 个基因段，将它们互换位置；在双性交叉中，随机地组合 2 个染色体，随机地交换这 2 个染色体中的某长度的基因段。由于每个染色体是基于工件号编码，其中各工件号的数目是一定的，交叉过后很可能会产生非法的染色体，为此设计了专门的调整函数 binarycross()。

调整规则为：如果交换 2 个基因段段 1 和段 2，若其中同一工件的工件号数目相同，则直接进行交换即可；否则，先计算出段 1 比段 2 每种工件号多出或减少的数目，然后在段 1 多出的工件号上随机地用缺少的工件号代替。同样，在段 2 多出的工件号上随机地用缺少的工件号代替，然后交换段 1 和段 2。

(5)变异：同时采用了交换、逆序、插入 3 种算子，对于某个决定变异的染色体，分别采用上述 3 种操作，寻找到该染色体的 3 个邻域染色体，运用公式求得每个邻域染色体的适应度，选择适应度最高的邻域染色体放入变异后的染色体种群。

(6)扰动操作：由于遗传算法的变异概率往往较小，只有少部分的染色体最后能够得到变异的机会，通过实验发现，在迭代一定次数之后，染色体种群的适应度相差不大，还有一些染色体适应度一样，使得算法搜索停滞不前。为了在一定程度上克服此缺点，本节设计了扰动操作，即在最好的那些染色体中，除只保留 1 个外，其他的染色体均采用逆序操作，用其邻域染色体来代替。

3 仿真实例

以一个规模较大的柔性 Job Shop 调度问题来验证算法的可行性，表 1 为该问题对应的工件：工艺路线、工序操作。表 2 为每个工序对应的可选加工机器，表 3 为每个工件在特定的工艺路线下，其包含的工序在特定机器上加工时间。算法的参数设计为：$popsize\_plan=10$，$maxiter\_1=20$，$p_c^{plan}=0.9$，$p_m^{plan}=0.1$；$popsize\_machine=10$，$maxiter\_2=5$，$p_c^{machine}=0.9$，$p_m^{machine}=0.1$；$popsize\_schedule=10$，$maxiter\_3=5$，$p_c^{schedule}=0.9$，$p_m^{schedule}=0.1$。

表 1 工件：工艺路线、工序操作

	$i=1$			$i=2$				$i=3$				$i=4$		$i=5$			$i=6$	
	1	2	3	1	2	3	4	1	2	3	4	1	2	1	2	3	1	2
$j=1$	1	1	0	1	1	1	1	0	1	1	1	0	1	1	1	1	0	1
$j=2$	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	0	1
$j=3$	1	0	0	1	0	1	0	1	0	1	1	0	0	1	0	1	1	0
$j=4$	0	1	1	1	0	0	1	1	0	0	1	1	0	1	0	1	1	0
$j=5$	1	1	1	1	1	0	1	1	1	0	1	0	0	0	1	0	1	0
$j=6$	0	1	0	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1

表 2 各工序对应的可选加工机器

机器 / 工序	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
$j=1$	1	1	0	0	1	1
$j=2$	1	0	1	0	1	0
$j=3$	0	1	1	1	0	1
$j=4$	1	0	1	0	0	1
$j=5$	1	1	1	1	1	1
$j=6$	0	1	1	0	1	1

所得到的最终结果为：$schedule^*=[1\ 6\ 3\ 1\ 6\ 6\ 5\ 2\ 4$

表 3 各工件在特定的工艺路线下所包含工序在特定机器上加工时间

工序	工件 / 工艺路线 / 时间	$i=1$			$i=2$				$i=3$				$i=4$		$i=5$			$i=6$	
		1	2	3	1	2	3	4	1	2	3	4	1	2	1	2	3	1	2
$j=1$	$k=1$	8	9	0	4	5	7	2	0	7	4	10	0	9	4	12	14	0	7
	$k=2$	10	15	0	3	3	10	6	0	8	9	12	0	10	5	7	9	0	9
	$k=5$	8	10	0	10	12	15	9	0	7	10	13	0	15	7	10	12	0	10
	$k=6$	5	12	0	9	13	8	3	0	10	10	10	0	10	10	4	13	0	4
$j=2$	$k=1$	3	0	7	0	0	4	5	12	15	7	5	10	7	7	15	12	0	15
	$k=3$	8	0	10	0	0	9	9	13	16	9	7	12	5	9	12	10	0	12
	$k=5$	6	0	8	0	0	10	10	9	20	15	12	10	9	7	13	9	0	11
$j=3$	$k=2$	10	0	0	5	0	2	0	3	0	4	10	0	0	10	0	7	10	0
	$k=3$	15	0	0	10	0	7	0	10	0	9	9	0	0	12	0	7	10	0
	$k=4$	9	0	0	7	0	9	0	4	0	8	3	0	0	9	0	12	9	0
	$k=6$	7	0	0	8	0	7	0	9	0	9	7	0	0	10	0	14	7	0
$j=4$	$k=1$	0	9	10	10	0	0	6	10	0	0	6	7	0	7	0	12	10	0
	$k=3$	0	12	15	12	0	0	6	5	0	0	9	5	0	8	0	11	9	0
	$k=6$	0	9	9	15	0	0	8	7	0	0	9	3	0	10	0	9	15	0
$j=5$	$k=1$	3	8	5	20	12	0	9	9	10	0	5	0	0	0	7	0	7	0
	$k=2$	7	12	10	15	9	0	8	10	8	0	7	0	0	0	7	0	8	0
	$k=3$	6	15	9	15	8	0	9	12	12	0	7	0	0	0	10	0	4	0
	$k=4$	7	13	8	12	13	0	10	15	15	0	10	0	0	0	15	0	4	0
	$k=5$	10	20	10	9	10	0	15	7	7	0	7	0	0	0	14	0	9	0
	$k=6$	12	15	13	10	13	0	19	9	8	0	9	0	0	0	15	0	10	0
$j=6$	$k=2$	0	3	0	2	9	10	17	11	0	0	12	6	7	0	7	0	12	20
	$k=3$	0	9	0	3	8	11	12	9	0	0	12	9	8	0	9	0	8	13
	$k=5$	0	8	0	6	6	20	21	8	0	0	15	6	12	0	12	0	8	12
	$k=6$	0	4	0	9	7	15	10	10	0	0	20	10	11	0	15	0	10	18

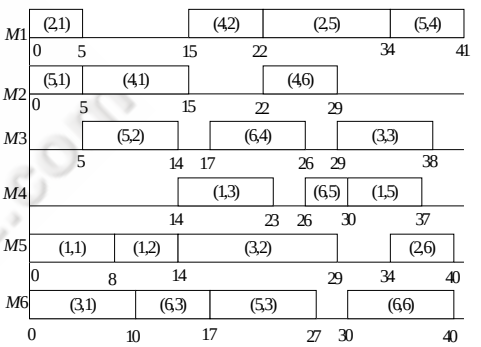

图 1 最优排序调度的甘特图

1 4 5 6 2 2 2 5 4 2 6 5 5 3 3 4 1 4 2 1 4 1 3 3 5 6 3]

*chrom_plan**=[1 2 3 2 1 1]

$$machine^{*}=\begin{bmatrix}5&5&4&0&4&0\\1&0&0&0&1&5\\6&5&3&0&0&0\\2&1&0&0&0&2\\2&3&6&1&0&0\\0&0&6&3&4&6\end{bmatrix};\ pt=\begin{bmatrix}8&6&9&0&7&0\\5&0&0&0&12&6\\10&15&9&0&0&0\\10&7&0&0&0&7\\5&9&10&7&0&0\\0&0&7&9&4&10\end{bmatrix};$$

$C_{max\_best}=41$。

按照最终结果将工序安排在机器上，所得到的甘特图如图 1 所示。

本文首先详细介绍了柔性 Job Shop 调度问题的模型，然后介绍了基本遗传算法的原理、构成、特点和流程。针对三级子问题，分别设计了其相应的遗传算法，并给出了其流程，最后通过一个实例，得到了相应的结果，结果证明，该算法的设计是行之有效的。

参考文献

[1] SCHRICH C R, ARMENTANO V A, LAGUNA M. Tardiness minimization in a Flexible Job Shop: A tabu search approach [J]. Journal of intelligent manufacturing, 2004, 15 (1): 103–115.

[2] BAYKASOGLU, A Linguistic–based meta–heuristic optimization model for Flexible Job Shop scheduling[J]. International Journal of Production Research, 2002, 40 (17): 4523–4543.

[3] KACEM I, HAMMADI S, BORNE P. Approach by localization and multiobjective evolutionary optimization for Flexible Job–Shop scheduling problems[J]. IEEE Transactions on Systems Man and Cybernetics Part C–Applications and Reviews, 2002, 32(2): 172–179.

[4] TIENTE H, DUPAS R, JOLLY D. Evaluation of mutation heuristics for the solving of multiobjective Flexible Job Shop by an evolutionary algorithm[C]. IEEE SMC, 2002: 450–455.

[5] 陈伟达，达庆利. 工艺路线可变的 Job–shop 准时生产调度研究. 管理工程学报, 2003, 17(1): 61–64.

(收稿日期：2009–04–07)