基于遗传算法的多资源作业车间智能动态优化调度*

孙志峻 朱剑英 潘全科
(南京航空航天大学机电工程学院 南京 210016)

摘要：提出一种基于遗传算法的调度算法，用于解决多资源制约(机床、工人和机器人)条件下作业车间的动态优化调度。为了表达加工工件的批量，提出了一种新的染色体基因型，基因型的长度随加工环境的变化而变化。研究的动态环境包括：加工工件连续不断地到来；机床设备突然损坏；损坏的机床被修复；工件的预定订货时间被提前；有新类型的工件要求被加工等等。采用一种基于周期和事件驱动的滚动窗口调度，以适应连续加工过程中的环境变化。调度算法中采用权重可变的双目标评价函数来优化调度结果。仿真结果表明该算法是可行的，与传统的静态优化调度相比，其优越性是明显的。

关键词：多资源 动态作业车间调度 遗传算法
中图分类号：F406

0 前言

目前国际上，在作业车间调度问题中，将只有机床设备资源受制约的调度问题称为单资源调度问题。由于工人的技术熟练程度与工作能力的不同，不同工人操作设备的种类和数量也是不同的，这样将机床设备和操作工人等两种资源受制约的调度问题，称为双资源调度问题，而把多种生产资源受制约的调度问题，称为多资源调度问题。对于调度问题，如果不考虑加工环境的变化，则称为静态调度问题；否则称为动态调度问题。

静态作业车间调度由于不考虑加工过程中加工环境的变化，所以其调度结果很难应用于实际加工过程。鉴于这种原因，有学者提出了适合动态调度的框架或策略[1~4]，但这种策略只适用于单一资源受制约的动态作业车间调度。

有学者将静态作业车间调度由单一资源受制约推广到两种资源受制约的情况。以往采用的方法是仿真与启发式搜索相结合[5, 6]，但是该方法存在着弊端：一方面仿真太耗时，另一方面由于采用了贪婪搜索而不易找到全局最优解[7]。由于遗传算法在求解传统旅行商(TSP)问题获得成功，越来越多的学者在求解车间作业调度问题时开始关注遗传算法[8~11]。

本文对一种多资源作业车间的动态调度问题进行了研究，并提出了解决方法。文中针对机床设备、操作工人和机器人这三种生产资源受到制约的情况，进行了研究。机器人的作用是在加工之前和加工之后，负责工件的上料和下料。之所以选择这样的机器人作为第三种受制约的资源加以研究，是因为它代表了另一大类的生产资源。它与操作工人这种生产资源的不同之处在于：在一道工序的加工过程中，操作工人作为资源被占用的时间与加工时间一致，操作工人在这段时间内不可能为其他工序服务；而机器人作为资源被占用的时间被分成两段，分别是上料时间和下料时间，但机器人在该工序加工过程中，完全可能被另一道工序征用。如操作工人这样所代表的生产资源还有刀具和夹具等；如文中所述的机器人所代表的生产资源还有物料运输车等。如果选择机床设备、操作工人和刀具这三种生产资源受到制约的情况，纵然也属于多资源调度问题，但由于在调度过程中，操作工人和刀具对于同一道工序而言，在时间上具有相同的属性，因而在调度的代表性方面不如机床设备、操作工人和机器人这三种生产资源的组合广泛。

根据车间实际工作的情况，本文研究了多种作业车间的动态环境：待加工工件连续不断地到来，机床设备突然损坏，损坏的机床被修复，工件的预定订货时间被提前和有新类型的工件要求被加工等情况。在动态环境中，像静态作业车间调度那样研究单批量的调度是没有实际意义的。文中针对的是加工工件的批量调度，即同一种加工工件的个数不止一个，且它们被申请加工的时刻也各不相同。为能对批量进行调度，文中提出了一种新的染色体基因型，基因型的长度随加工环境的动态变化而变

* 国家自然科学基金资助项目(59990470)。20010403 收到初稿，20010902 收到修改稿

化。受预测控制技术中滚动窗口优化方法的启发，文中采用一种基于周期和事件驱动的滚动窗口调度方法。

1　多资源动态调度问题描述

在本文的作业车间调度问题中，有四种不同类型的工件，标记为 J_i (i=1，2，3，4)；每一种工件的批量是 5 个，且加工工序都是三道，每道工序均可被不止一台的机床加工；各种工件的到期时刻分别为 60，80，95 和 100。车间的生产能力受到机床设备、操作工人和机器人这三种生产资源的制约。各种不同类型的机床设备数是 6 台，标记为 M_k (k=1，2，…，6)；操作工人的人数是 4 人，标记为 W_w(w=1，2，3，4)，每个操作工人均可操纵 1 台以上的机床设备；1 台机器人需要为 2 台机床服务，为在该机床上加工的工件进行上料和下料，机器人数为 3 台，标记为 R_r (r=1，2，3)。调度的目标是寻找一个可行的调度方案，使得在给定的约束条件下，获得某种评价指标的最优解。

每个操作工人的加工能力和机器人为特定机床服务的信息由工人/机器人工作表提供，如表 1 所示。四种不同类型工件的加工时间以及每道工序的上下料时间信息由加工时间/上下料时间表提供，如表 2 所示。从表中可以看出工序在不同机床设备的加工时间各不相同。不同类型的各个工件的到达时间由表 3 提供。

表 1　工人/机器人工作表

W/R	M_1	M_2	M_3	M_4	M_5	M_6
W_1	操纵	操纵	—	—	—	—
W_2	—	操纵	操纵	—	—	—
W_3	—	—	—	操纵	操纵	—
W_4	—	—	—	—	操纵	操纵
R_1	服务	服务	—	—	—	—
R_2	—	—	服务	服务	—	—
R_3	—	—	—	—	服务	服务

表 2　加工时间/上下料时间表

J	O	M_1	M_2	M_3	M_4	M_5	M_6	L	UnL
	1—1	2	3	4	—	—	—	0.2	0.1
J_1	1—2	—	3	—	2	4	—	0.2	0.1
	1—3	1	4	5	—	—	—	0.4	0.3
	2—1	3	—	5	—	2	—	0.3	0.2
J_2	2—2	4	3	—	—	7	—	0.4	0.3
	2—3	—	—	4	—	7	11	0.4	0.3
	3—1	5	6	—	—	—	—	0.3	0.2
J_3	3—2	—	—	—	4	3	5	0.3	0.2
	3—3	—	—	13	—	9	12	0.3	0.2
	4—1	9	—	7	9	—	—	0.4	0.3
J_4	4—2	—	6	—	4	—	5	0.4	0.3
	4—3	1	—	3	—	—	3	0.5	0.4

表 3　工件的到达时间表

工件	到达时间	工件	到达时间
11(A)	8	31(K)	4
12(B)	19	32(L)	28
13(C)	28	33(M)	0
14(D)	20	34(N)	13
15(E)	35	35(O)	17
21(F)	2	41(P)	20
22(G)	32	42(Q)	24
23(H)	15	43(R)	29
24(I)	10	44(S)	18
25(J)	26	45(T)	34

这里研究的动态环境有：在时刻 18，第四种类型的工件突然到达；在时刻 25，第二台机床损坏；在时刻 31，第二种类型的工件订货到期时间从 80 提前到 65；在时刻 40，第二台机床被修复。

2　实现方法

2.1　遗传算法

遗传算法是基于群体进化的一种方法，其最大优势在于它同时使用多个搜索点的搜索信息，有能力在各种调度方案之间进行选择、交叉和变异等运算，可以跳出局部最优的陷阱。

2.1.1　调度编码

在文献[12]中分析了多种调度染色体基因型的优劣，但它们的适用范围仅仅是单件调度。从表 2 的工序加工信息表中可以获得这样的信息：每道工序在调度确定之前，无法确定为其加工的机床。考虑到这种情况，采用的染色体基因型是基于工序的。具体的调度编码形式对文献[12]中的编码形式进行了扩展和修改，以适应这种情况以及批量的调度。给每个工件一个两位十进制的编号，如第三种类型的第二个工件编号是“32”；工件的工序编号则是三位十进制编号，如第三种类型的第二个工件的三道工序的编号分别是“321”，“322”和“323”。在染色体编码中，给所有同一工件的工序指定相同的符号，即工件编号，根据它们在给定染色体中出现的先后顺序就可以确认是第几道工序，如表 4 所示。很容易看出染色体的任意排列总能产生可行调度，而且可以肯定这种编码方式一定含有最优调度。

表 4　调度的染色体基因型(片段)

工序	321	111	311	412	421	322
编码	32	11	31	41	42	32
工序	222	422	323	112	131	423
编码	22	42	32	11	13	42

2.1.2 选择、交叉和变异

由于每种类型的工件都有订货到期时间的约束，所以调度的目标是双重的：一方面要求在最短的时间内(Makespan)将这批工件全部加工完毕，另一方面又要求工件加工完毕时间早于订货到期时间或者延迟时间(Tardiness)越少越好。这是一个典型的双目标优化调度问题，且这两个指标都要求越小越好。为解决这个问题，采用文献[13]提出的变权重的选择方法。在式(1)中，$f(x)$是一个综合的适应度方程，$f_1(x)$和$f_2(x)$分别代表 Makespan 和 Tardiness 目标方程

$$f(x)=w_1f_1(x)+w_2f_2(x) \quad (1)$$

$$\begin{cases} w_1=(i-1)/(N-1) \\ w_2=1-w_1 \end{cases}$$

式中 N——每代的种群个数

i——整数集合[1，N]的一个随机数

由于权重 w 不是常量而是变量，使得每个染色体即使在某一个单独的代中被选择的概率也是变化的。这样可以在多目标遗传算法中，实现可变方向上的寻优，能获得较全面的 Pareto 解。随后采用 Holland 正比选择法生成生存概率。然后，用随机通用采样法只旋转一次转轮而得到后代数。

交叉操作可以将父代的良好基因通过信息互换，而产生更好的子代。然而，对于作业车间调度问题(JSP)和旅行商问题(TSP)，交叉操作是最棘手的难题。考虑到染色体基因型的实际意义，采用随机两点交叉操作方式，这样两个父代产生一个子代。一般这样操作很容易产生不合理的染色体基因型。这里，采用了一种非常好的交叉操作方式[14]，不仅可以完全避免不合理染色体基因型的产生，而且在操作时间上也大大缩短。最后，将两个父代与一个子代染色体比较，选择最好的两个染色体放入种群中。

变异操作在本问题中，相对而言比较简单。在随机选择的染色体内随机交换两个基因，而形成一个新的染色体即可。

2.2 基于周期和事件驱动的滚动窗口调度

为了解决作业车间动态环境调度问题，采用基于周期和事件驱动的滚动窗口技术来实现调度，这是预测控制中滚动优化原理的一种具体表现形式。

2.2.1 工件窗口

滚动窗口调度策略中的优化区间是工件窗口，即一定数目的工件。可将工件分成三个集合：可得工件集 S1、窗口工件集 S2 和已完成加工工件集 S3。可得工件集 S1 是所有到达时间已知的尚未调度工件的集合，窗口工件集 S2 就是当前优化区间中的工件集合，已完成加工工件集 S3 是已经加工结束的工件集合。当一个工件的到达时间已知时，该工件加入可得工件集 S1，并等待进入窗口工件集 S2。当一个工件的所有工序被加工完毕，该工件从窗口工件集 S2 进入已完成加工工件集 S3。由于工件都有订货到期的时间约束，所以选择工件进入窗口工件集 S2 的依据，就是工件加工的紧迫度值。用工件的订货到期时刻减去此时即将调度的时刻就可获得工件的紧迫度值。紧迫度值越小，进入窗口工件集 S2 的优先级越高。

2.2.2 调度描述

当进入窗口工件集中的工件被确定之后，就需要应用调度算法进行优化，以确定在当前时间段，处于窗口工件集中工件的最优调度。调度算法的过程如下所述。

先说明在文中出现的标号：

i——工件号 ($i=1, 2, \cdots, n$)

k——机床号 ($k=1, 2, \cdots, m$)

w——工人号 ($w=1, 2, \cdots, d$)

r——机器人号 ($r=1, 2, \cdots, e$)

$Machine_k$——可以使用机床 k 的时刻(机床 k 在此刻开始空闲)

$Person_w$——可以使用工人 w 的时刻(工人 w 在此刻开始空闲)

$Robot_r$——机器人 r 工作时间区间集合

$Time_{ij}$——工件 i 的第 j 道工序可被加工的时刻点

Job_i——工件 i 的总工序号

t_{ijk}——在机床 k 上加工工序 O_{ij} 的所需时间段

$Loading_{ij}$——工件 i 的第 j 道工序的安装所需时间段

$Unloading_{ij}$——工件 i 的第 j 道工序的拆卸所需时间段

$Finish_{ijk}$——在机床 k 上完成工件 i 的第 j 道工序的时刻点

$Early_{ij}$——工件 i 的第 j 道工序的最早完工时刻点

$Finish_i$——工件 i 的完工时刻点

Set——一种调度方案中所有未被调度的工序集合

StartLoad(k,1)——机器人在机床 k 上开始安装工件的理想时刻点

EndLoad(k,2)——机器人在机床 k 上完成安装工件的理想时刻点

StartUnload(k,1)——机器人在机床 k 上开始拆卸工

件的理想时刻点

EndUnload(k，2)——机器人在机床 k 上完成拆卸工件的理想时刻点

算法过程如下所述。

步骤 1：计算所有工序的理想可被加工启始时刻

设 $\text{Machine}_k=0$，$\text{Person}_w=0$，$\text{Robot}_r=(0,0)$，$\text{Time}_{i1}=0$

$\text{Time}_{ij}=\text{Loading}_{i(j-1)}+\text{Time}_{i\ (j-1)}+t_{i(j-1)k}+\text{Unloading}_{i\ (j-1)}$ ($i=1,2,\cdots,n$ 和 $j=2,\cdots,\text{Job}_i$)

步骤 2：如果在集合工序中还有未被调度的工序，从集合中取出此刻排在第一位未被调度的工序转步骤 3。

否则，设 $\text{Makespan}=\max(\text{Finish}_i)$ ($i=1,2,\cdots,n$)

步骤 3：$\text{StartLoad}(k,1)=\max\{\text{Machine}_k, \text{Person}_{w,}, \text{Time}_{ij}\}$

$\text{EndLoad}(k,2)=\text{StartLoad}(k,1)+\text{Loading}_{ij}$

如果($\text{StartLoad}(k,1)$，$\text{EndLoad}(k,2)$) $\in \text{Robot}_r$，则调整 $\text{StartLoad}(k,1)$ 和 $\text{EndLoad}(k,2)$

$\text{StartUnload}(k,1)=\text{EndLoad}(k,2)+t_{ijk}$

$\text{EndUnload}(k,2)=\text{StartUnload}(k,1)+\text{Unloading}_{ij}$

如果 ($\text{StartUnload}(k,1)$，$\text{EndUnload}(k,2)$) $\in \text{Robot}_r$，则调整 $\text{StartUnload}(k,1)$ 和 $\text{EndUnload}(k,2)$ $\text{finish}_{ijk}=U(k,2)$

步骤 4：设 $\text{Early}_{ij}=\min(\text{finish}_{ijk})$

如果 $j=\text{Job}_i$，则 $\text{Finish}_i=\text{Early}_{ij}$

否则，计算 $\text{Time}_{i(j+h)}=\text{Time}_{i(j+h)}+(\text{Early}_{ij}-\text{Time}_{i(j+1)})$，($h=1,2,\cdots,(\text{Job}_i-j)$)

步骤 5：设 $\text{Machine}_k=\text{Early}_{ij}$，$\text{Person}_w=\text{Early}_{ij}$，把 ($\text{StartLoad}(k,1)$，$\text{EndLoad}(k,2)$) 和 ($\text{StartUnload}$，$(k,1)$ $\text{EndUnload}(k,2)$) 并入相应的集合 Robot_r。

步骤 6：检查集合 Set 中是否还有未被调度的工序;如果还有,则转步骤 2。

否则，makespan = max (F_i) 和加工完毕工件的延误时间。

2.2.3　基于周期和事件驱动的调度策略

对于再调度一般有两种策略：第一种是当系统状态一旦有所变化，立即进行调度；第二种则是每隔一段时间进行周期调度。Church 和 Uzsoy[3] 提出了将这两种策略混合在一起，同时考虑的再调度方法。本文的再调度策略沿用了这种方法。

(1) 周期性再调度

本例中，周期性再调度的时间间隔是 10。从时刻 10 开始从可得工件集 S1 中，依据紧迫度值的大小，选择工件进入窗口工件集 S2。对窗口工件集 S2 中工件的调度算法如上所述。在时刻 10 和时刻 20 之间，如果没有突发事件发生，则下一次再调度的时刻就是 20。可得工件集 S1 中工件的数量和种类由于有新工件不断地到来，会发生变化。再调度时，把可得工件集 S1 和窗口工件集 S2 中未加工结束的工件集中起来，根据紧迫度值的大小重新选择工件进入窗口工件集 S2。如果系统的加工状态始终没有突发事件产生，则再调度的时刻应该是 20、30、40 等 10 的整数倍，直到全部加工工件进入已完成加工工件集 S3。如果在这些整数段时间内有突发事件发生，则再调度的时刻就是突发事件产生的时刻。

(2) 事件驱动再调度

当某台机床设备突然损坏，正在该机床上加工的工件只能安排到其他机床上进行加工。为该工件的当前工序选择合理的生产资源，必然会改变窗口工件集 S2 中其他工件对生产资源的选择。同时，从上一次调度时刻到该突发事件产生的时刻，可得工件集 S1 中可能会有较紧迫的工件到达，为使这些工件尽快地进入窗口工件集 S2，再调度时，把可得工件集 S1 中的工件和窗口工件集 S2 中未加工结束的工件集中起来，根据紧迫度值的大小重新选择工件进入窗口工件集 S2。

同样，在下面的几种突发事件发生的时刻，都应该立即进行相同的再调度：①当损坏的机床设备被修复后，该机床立刻成为可利用的生产资源，为了尽快使它投入生产。②当某种类型工件的预定订货时间被提前，意味着该类工件的加工紧迫度值变小，有可能立即需要被加工。③当有新类型的工件突然提出要求加工时，有可能它们的加工紧迫度比正在加工的工件还要紧迫。

根据上述的分析，结合本文的算例，周期性的再调度的时间间隔取为 10，再根据一些突发事件的发生时刻，则再调度的时间点应该是 18、25、31、40、50、60、70、80、90 和 100。

3　仿真结果和结论

在上述的调度算法中，采用的分派规则是优先选择最短加工时间的工序(Shortest processing time，SPT)和优先选择具有最早交货期的工件(Earliest due date，EDD)。最优调度的评价指标是综合考虑最短生产周期(Makespan)和最少延误时间(Tardiness)。通过遗传算法获得的最佳调度的 Gantt 图，如图 1 和图 2 所示。

图 1 显示了操作工人和工件的对应关系。图中横坐标表明了这批工件加工的时间历程，纵坐标表明操作工人，用三个字符标识的方框代表一道工

序。例如，第三个操作工人加工的第一道工序是“I15”，“I”表示该工件是第二种类型的第四个工件；中间的“1”表示这是该工件的第一道工序；“5”表示该工序在第五台机床上加工。从横坐标上可以查出加工该工序从开始上料和加工完毕后的下料时刻，同理可以得到各工件被最终加工完毕的时间。每一个工件和一个字母的一一对应关系从表 3 中可以得到。这样做的目的，仅仅是为了在图中标识方便。

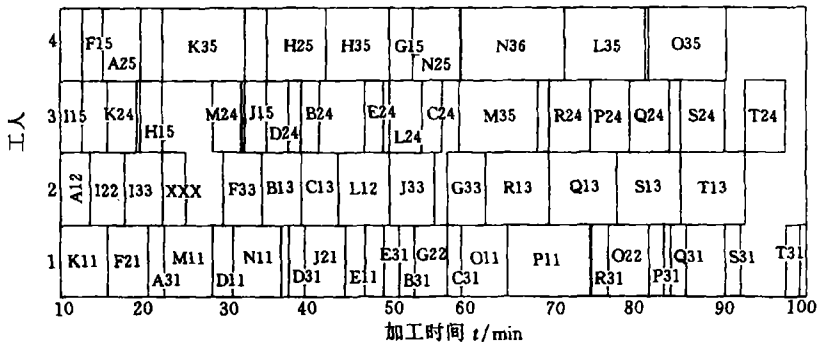

图 1 操作工人和工件的对应关系图

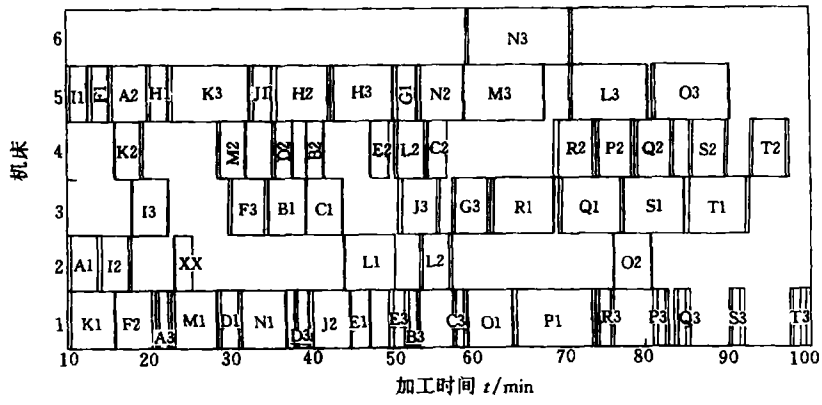

图 2 工序和机床的分配关系图

图 2 显示了工序和机床的分配关系，其间还标明了机器人为每道工序服务的时间。图中横坐标表明了这批工件加工的时间历程，纵坐标表明机床设备，用两个字符标识的方框代表一道工序。例如，第一台机床加工的第一道工序是“K1”，通过查表 3，可以得知该工件是第三种类型的第一个工件；“1”表示这是该工件的第一道工序，从横坐标上可以查出加工该工序的启始和完毕时刻。在该工序的前后有两条细的方框，它表示了相关的机器人上下料的时间。

从这两张图中均可看出这批工件最终加工完毕的时间是 99.26。这是通过调度算法获得的最佳调度的完工时间，而且没有工件发生延误。在图 1 中，第二个操作工人加工的第四个工序，以及图 2 中，在第二台机床上加工的第三道工序，分别被标识为“xxx”和“xx”。它们表明了同样的一个过程：第二个操作工人在第二台机床上加工这道工序过程中，机床发生了损坏，不能继续加工这道工序。从图 2 中可以看出该机床一直无法加工任何工件，直到它被修复以后，而操作工人可以在其他机床，加工其他工序。从图 1 中，可以清楚地看到在与图 2 相应的这段时间内该操作工人加工了“F33”、“B13”和“C13”等工序，所以图中用“xxx”和“xx”来标识这是一道无效工序。

如果采用静态的车间调度策略来处理文中提到的环境变化，显然是行不通的。首先，静态调度策略无法应付待加工工件不断到达的情况；其次，对环境的变化不能作出相应的变化。

从仿真的结果看，文中提出的调度算法完全可行，由于采用滚动窗口技术，可以使一个大而复杂的系统问题，分割成许多较小的问题，从而该算法完全可以适应连续加工过程中复杂的环境变化和实时处理。同时，笔者认为该算法有很强的扩展性，可以应用到更多生产资源受制约的情况。

参 考 文 献

1 Laurak Church, Reha Uzsoy. Analysis of periodic and event-driven rescheduling policies in dynamic shops. International Journal of Computer Integrated Manufacturing, 1992,5(3): 153~163

2 Sun D, Lin L. A dynamic job shop scheduling framework: a backward approach. International Journal of Production Research, 1994, 32(4): 967~985

3 Jian Fang, Yugeng Xi. A rolling horizon jop shop rescheduling strategy in the dynamic environment. Advanced Manufacturing Technology, 1997, 13: 227~232

4 Monostori L, Kadar B, Hornyak J. Approaches to managing changes and uncertainties in manufacturing. Annals of the CIRP, 1998, 47(1): 365~368

5 Treleven M D, Elvers D A. An investigation of labor assignment rules in a dual resource constrained fob-shop. The Journal of Operation management, 1985, 6(1): 51~68

6 Treleven M D. A review of the dual-resource constrained system research. IIE Transactions, 1989, 21: 279~287

7 Miller J G, Berry W L. Heuristic methods for assignment men to machines: an experiment analysis. AIIE Transactions, 1974, 6: 97~104

8 Hon K K B, Chi H. A new approach of group technology parts families optimization. Annals of the CIRP, 1994, 43(1): 425~428

9 Wiendahl H P, Garlichs R. Decentral production scheduling of assembly systems with genetic algorithm. Annals of the CIRP, 1994, 48(1): 389~392

10 Wong N, Leu M C. Adaptive genetic algorithm for optimal printed circuit board assembly planning. Annals of the CIRP, 1993, 42(1): 17~20

11 Gargeyal V B, Deane R H. Scheduling research in multiple resource constrained job shop: a review and critique. International Journal of Production Research, 1996, 34(8): 2 077~2 097

12 Runwei Cheng, Mitsuo Gen, Yasuhiro Tsujimura. A tutorial survey of job-shop scheduling problems using genetic algorithms-1. Computers Industry Engineering, 1996, 30(4): 983~997

13 Tasahiko Murata, Hisao Ishibuchi, Hideo Tanaka. Multi-objective genetic algorithm and its applications to flowshop scheduling. Computer and Engineering, 1996, 30(4): 957~968

14 Guoyong Shi. A genetic algorithm applied to a class job-shop scheduling problem. International Journal of Systems Science, 1997, 28(1): 25~32

GENETIC ALGORITHM BASED APPROACH TO THE INTELLIGENT OPTIMUM SCHEDULING OF MULTI-RESOURCES IN THE DYNAMIC ENVIRONMENT

Sun Zhijun　Zhu Jianying　Pan Quanke

(Nanjing University of Aeronautics and Astronautics)

Abstract: Based on genetic agorithms (GAs), a scheduling approach is presented, which can be used to address the job shop scheduling problem in dynamic manufacturing systems constrained by machines, workers and robots. A new chromosome representation is also presented for batch process scheduling and its length is variable. In the dynamic environment, jobs arrive continuously, machines may be broken and repaired, due date of job may change, a new class job comes up during processing. Inspired by the rolling horizon optimization method from predictive control technology, a periodic and event-driven rolling horizon scheduling is utilized for adaptation to continuous processing in a changing environment. The algorithm takes into account dispatching rules with variable weights in the performance function. Simulation results show that the strategy is more suitable for a dynamic job shop environment than the static scheduling strategy.

Key words: Multi-resources　Dynamic job-shop scheduling　Genetic algorithm

作者简介：孙志峻，男，1970 年出生。博士研究生。主要从事智能制造和生产自动化方面的研究工作，发表论文 10 余篇。