引用格式:李云龙,罗国富,文笑雨,等. 基于混合遗传算法的云制造环境下柔性作业车间调度方案[J]. 轻工学报,2020,35(3):99 – 108.
中图分类号:F416.2;TH165 **文献标识码**:A
DOI:10.12187/2020.03.012
文章编号:2096 – 1553(2020)03 – 0099 – 10

基于混合遗传算法的云制造环境下柔性作业车间调度方案

Flexible job shop scheduling in cloud manufacturing environment based on hybrid genetic algorithm

李云龙,罗国富,文笑雨,李凯,杨幸博,张俊豪
LI Yunlong, LUO Guofu, WEN Xiaoyu, LI Kai, YANG Xingbo, ZHANG Junhao

郑州轻工业大学 机电工程学院/河南省机械装备智能制造重点实验室,河南 郑州 450002
College of Mechanical and Electrical Engineering/He'nan Provincial Key Laboratory of Intelligent Manufacturing of Mechanical Equipment, Zhengzhou University of Light Industry, Zhengzhou 450002, *China*

关键词:
云制造;剩余能力;柔性作业车间调度;混合遗传算法

Key words:
cloud manufacturing; surplus capacity; flexible job shop scheduling; hybrid genetic algorithm

摘要:针对云制造环境下柔性作业车间调度产生的离散型加工设备的空闲时间利用及其冲突问题,提出了一种基于混合遗传算法的云制造环境下柔性作业车间调度方案:在保证车间任务顺利完成的前提下,对车间剩余能力进行界定后封装发布到云平台上;以最小惩罚总成本为目标,结合车间生产调度实际情况选择云订单任务一起加工;采用遗传变邻域混合算法求解云任务工件最优调度顺序. 基准算例测试结果表明,该方案实现了车间自身生产任务和云平台任务协同生产,提高了企业的收益和资源利用率.

收稿日期:2019 – 08 – 30
基金项目:国家自然科学基金项目(51905494);郑州轻工业大学硕士科技创新基金项目(2018015)
作者简介:李云龙(1993—),男,河南省周口市人,郑州轻工业大学硕士研究生,主要研究方向为数字化设计与制造.
通信作者:罗国富(1963—),男,河南省禹州市人,郑州轻工业大学教授,博士,主要研究方向为 CAD/CAM,敏捷制造,企业信息集成,MRPⅡ/ERP/MES 软件开发等.

Abstract: Aiming at the problem of idle time utilization and conflict of discrete processing equipment generated by flexible job shop scheduling in cloud manufacturing environment, a flexible job shop scheduling scheme in cloud manufacturing environment based on hybrid genetic algorithm was proposed. Under the premise of ensuring the smooth completion of workshop tasks, the residual capacity of the workshop was defined and then packaged and released to the cloud platform. Taking the minimum penalty total cost as the goal, combined with the actual situation of workshop production scheduling, the cloud order tasks were selected to process together, and the genetic variable neighborhood hybrid algorithm was used to solve the optimal scheduling sequence of cloud tasks, and the optimal scheduling scheme was formulated. The benchmark test results showed that the scheme realized the collaborative production of workshop production tasks and cloud platform tasks, and improved the enterprise's revenue and resource utilization.

0　引言

云制造是一种新兴的面向服务的业务模型，它集成了分布式制造资源，将其转换为制造服务，并集中管理服务[1]. 它利用信息技术将制造资源集中，为需求者和资源提供者建立了一个公共的资源服务平台[2]，允许多个用户将其需求任务提交到云制造平台，同时请求服务.

对大多数云制造企业而言，企业的生产任务不仅包括车间正在进行的生产任务，还包括从云平台选择接受的生产任务. 与大多数制造企业生产特点相同，云制造生产任务同样具有多品种、小批量的特点，云制造环境下也多涉及柔性作业车间问题. 由于柔性作业车间的离散型生产特点，车间生产任务在进行生产计划安排和初始调度后，车间的生产设备会存在空闲时间段，此时的机器设备处于等待下一个工件加工的待机空闲状态，这种车间生产加工的剩余能力，不可避免地给企业造成一定的生产资源浪费[3]. 于是，如何将车间本身的生产任务与接受的云平台任务进行合理调度，充分利用车间设备的剩余能力，使企业利益最大化，便成为云制造企业亟待解决的问题[4].

目前，国内外的业内专家对于云制造的研究大多集中在云平台的制造资源分类和优化匹配等方面，而对于云制造环境下企业车间级的生产调度问题鲜有人研究. 文献[4]提出了一种确定工序加工时间序列的方法，以设计更新策略求解具有空闲时间段的车间调度问题，在进行基于工件的编码设计时，采用精简编码方法，以改进二阶粒子群算法来求解工件最优调度顺序. 但该方法只针对作业车间最基本的调度情况，对于复杂的柔性作业车间的调度问题探讨得不够深入. 文献[5]针对柔性作业车间调度问题提出了一种染色体编码方法，有效地改进了遗传算法求解柔性作业车间的调度问题，但该方法仅适用于车间级的生产情况，并没有考虑云制造复杂环境下具有空闲时间段的车间调度问题. 文献[6]基于工序的编码设计，提出了一种新的 POX 交叉算子，该方法通过改进传统的遗传算法能够更有效地求解作业车间调度问题，较其他交叉算子更高效. 但是该方法局限性强，面向对象比较单一，对于柔性作业车间问题的考虑不够全面. 文献[7]通过传统蚁群算法来求解车间剩余能力，试图解决云制造企业车间空闲设备的利用问题，但得出的优化结果不够理想，不易获得高质量的调度解.

鉴于此，本文拟在分析云制造环境下柔性作业车间调度问题新特点的基础上，针对云制造企业生产车间剩余能力的利用问题，提出一种基于混合遗传算法的云制造环境下柔性作业车间调度方案，以期在保证车间初始调度任务顺利进行的同时，充分利用生产设备的空闲时间段，为云制造企业车间生产的科学调度与资

源的合理利用提供技术参考.

1 云制造环境下柔性作业车间调度问题描述

1.1 问题特点

在云制造环境下,企业在进行车间自身生产任务的同时,将车间机器设备的剩余能力封装发布到云平台上,以寻求更多合适的生产任务,提高企业本身的收益和设备利用率. 云制造服务平台的虚拟资源池中不仅有生产企业提供的制造资源,还有客户提出的服务请求. 云平台根据企业的生产能力和客户的服务需求进行供需智能匹配调度,并将匹配的结果反馈给企业和客户,使企业的柔性作业车间能够对云平台反馈的结果进行生产调度.

但是云制造企业在调度来自云平台的生产任务时,还需考虑车间本身正在进行的调度任务,这种云制造环境下柔性作业车间调度产生的新特点,给传统的车间生产调度带来了新的问题. 柔性作业车间生产特点如图 1 所示. 传统制造企业的柔性作业车间在开始进行生产任务之前,机器都处于空闲状态,可以在任意时刻选择相应机器进行加工任务,只要能够保证完成生产任务即可. 但是选择接受云平台的任务订单后,企业在进行车间本身生产任务的同时还要安排云平台订单,这就要考虑两种调度任务产生的机器冲突、时间冲突等问题,从图 1 所示云制造企业柔性作业车间可以看到,机器 M_1, M_2, M_3 在阴影部分时间内要进行车间本身的生产任务,那么从云平台接受的生产任务就不能在这些被锁定的时间段内进行,这就给传统的柔性作业车间的调度带来了新的挑战.

1.2 问题分析

云制造企业将车间设备的剩余能力封装发布到云平台上,接受云平台反馈的客户订单任务,将其与车间本身下达的任务一起安排加工,虽然提高了企业设备资源的利用率,但是也很可能会导致云平台的任务无法按时完成,产生惩罚成本.

不同车间任务加工甘特图如图 2 所示(图中阴影部分为已经被车间生产任务占用的时间段). 由图 2b)可以看出,由于车间任务的工序占用加工机器,当进行云任务加工时,在加工机器 M_1 和 M_3 上的开始时间和结束时间不能落在已经被车间本身任务锁定的时间区间内,需满足云平台任务加工工序的开始时间大于被锁定时间区域的结束时间,或者云任务加工工序的结束时间小于被锁定的时间区域的开始时间.

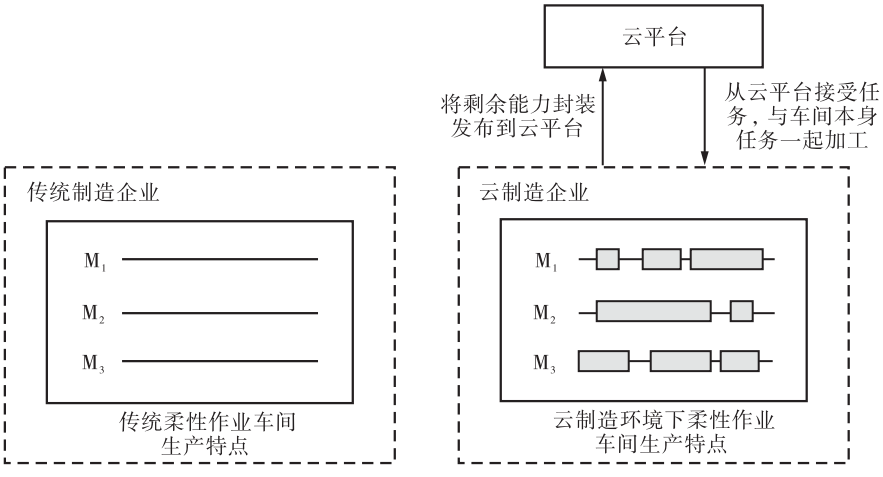
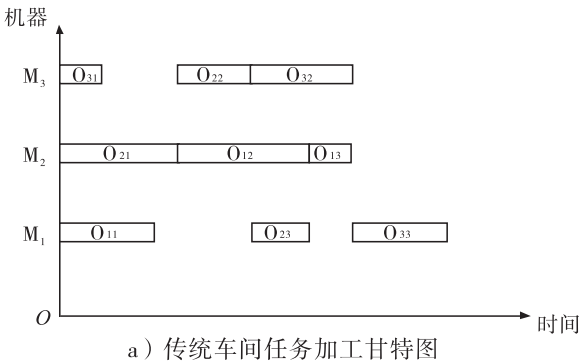
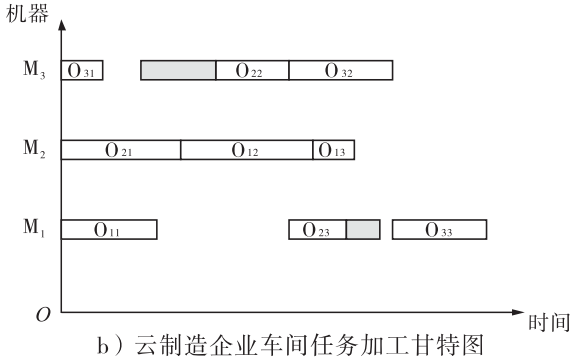


图 1 柔性作业车间生产特点

Fig. 1 Characteristics of flexible job shop production



a）传统车间任务加工甘特图



b）云制造企业车间任务加工甘特图

图 2　不同车间任务加工甘特图

Fig. 2　Gantt chart of different workshop task processing

这样,云平台任务最后的完工时间会延后甚至拖期. 但若接受云平台任务最后的收益大于惩罚成本,那么企业仍存在收益,这对企业的经济增长和资源的合理利用具有一定意义[7].

1.3　问题的数学模型

建立云订单的拖期惩罚成本目标函数:

$$G = \min\left[e_i \sum_{i=1}^{n} \max(0, c_i - d_i)\right]$$

其中,e_i 为惩罚系数,c_i 为完工时间,d_i 为交货期,n 为工件个数.

云制造环境下柔性作业车间调度问题描述为:车间本身正在进行的加工任务,有 n 个工件,要在 m 台机器上进行加工,每个加工工件都包括一道或多道加工工序,而每道工序都能够在不止一台加工机器上进行,但是不同机器加工同一道工序所需的加工时间不同[8]. 求解该问题时还需要以下假设条件:

1)一台机器在同一时刻仅可以处理一道工序;

2)各个工件之间具有一样的优先级;

3)各个工件的工序彼此不存在相互制约的关系;

4)每个工件的工序都是确定的,且有先后约束,即任意一道工件的前一道工序加工完成后,才能进行下一道工序的加工;

5)在云平台上接受的订单任务的最后收益要大于因拖期而产生的惩罚成本;

6)云平台接受任务的各个工序在相应加工机器上的开始时间和结束时间均不能落在已经被车间自身任务锁定的时间区间内[7].

2　云制造环境下柔性作业车间调度问题解决方案

基于以上分析,本文提出云模式下面向车间级的调度框架,对车间剩余能力进行界定后封装发布到云平台上,并选择接受合适的云订单. 根据车间本身正在进行的生产任务确定云订单生产任务各工序的开始时间和结束时间,得出合理而高效的调度方案,最后提出遗传变邻域混合算法来求解工件最优调度顺序.

2.1　调度框架构建

对云平台任务进行合理调度的前提是优先保证车间自身任务顺利完成,基于该原则,构建云制造企业柔性作业车间调度框架(如图 3 所示),具体步骤如下.

步骤 1　云制造企业按照柔性作业车间自身的生产任务制定初始调度方案,并通过对机器设备的能力分析和调度分配,确定车间的剩余生产能力,将其封装发布到云平台上[4].

步骤 2　云平台接收并储存云制造企业车间设备的生产能力后,结合已上传云订单的需求方任务要求进行匹配和调度,得到符合供需双方要求的匹配结果.

步骤 3　云制造企业接受云订单任务后,

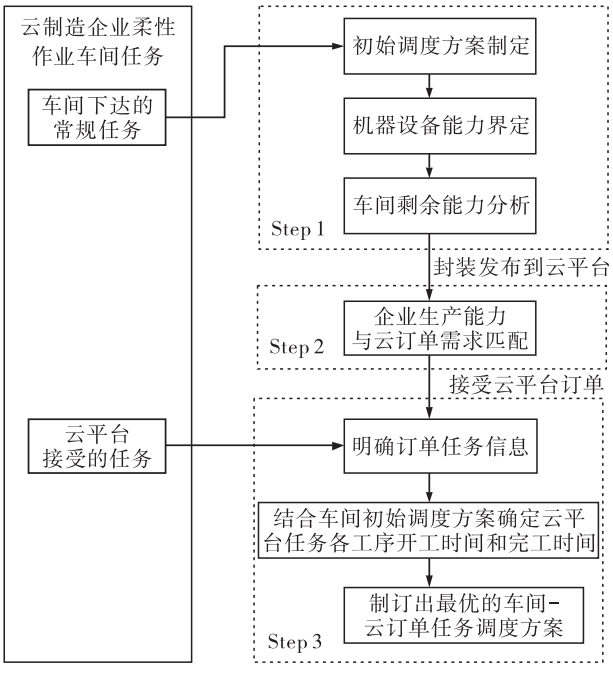


图 3 云制造企业柔性作业车间调度框架

Fig. 3 Flexible job shop scheduling framework of cloud manufacturing enterprise

明确工件调度顺序与各工序开工时间和完工时间,结合车间自身正在进行的常规生产任务调度方案,制订出最优的车间 - 云订单任务调度方案[8].

2.2 邻域结构确定

柔性作业车间调度问题 FJSP(flexible job shop scheduling problem)是较为复杂的 NP-hard 问题,求解难度比较大. 目前,多采用遗传算法、变邻域搜索算法等启发式算法来求解柔性作业车间调度优化问题. 遗传算法通过选择、交叉、变异等操作,实现对调度解的全局搜索,但其本身也容易陷入局部解和早熟. 而变邻域搜索算法在搜索过程中,通过采用不同的邻域结构,使算法具有较强的局部搜索能力,可以避免最终的结果陷入局部最优解,但全局搜索能力不及遗传算法. 采用单一的算法求解柔性作业车间调度问题都有一定的局限性,不易获得高质量的调度解. 本文将变邻域搜索算法嵌入遗传算法中,使混合算法同时具有较强的全局搜索能力和局部搜索能力,以提高求解柔性作业车间调度问题的速度和调度解的质量.

邻域结构为变邻域搜索算法提供搜索的方向,在搜索过程中采用不同的邻域结构可以有效地避免陷入局部最优解,是实现变邻域搜索算法的关键.

2.2.1 关键路径 由于关键路径的长短决定了最大完工时间,因此,大多数邻域结构都是基于关键路径设计的. 关键路径的确定方法[9]为:从完工时间最大的工序开始向前查找相连接的工序,当某一工序 w 的开工时间与工序前续工序 $JP[w]$ 和机器前续工序 $JS[w]$ 的完工时间相等时,取工序 w 的工序前续工序 $JP[w]$ 为关键工序,最后得到调度解的关键路径.

2.2.2 跨机器移动工序的邻域结构 本文采用基于机器空闲时间的关键工序移动策略,在空闲时间段 $[c^E(JP[w]),s^L(JS[w])]$ 内,关键工序 w 的机器空闲时间有两种情况:一是关键工序 w 在其他可选加工机器上的空闲时间;二是关键工序 w 在已选加工机器上的空闲时间. 而本文采用的跨机器移动工序是对第一种机器空闲时间的利用. 对关键工序 w 的跨机器邻域操作如图 4 所示,其中,x 和 y 为相邻的两道工序,关键工序 w 可以选择 x 和 y 所在的机器进行加工,且关键工序 w 在该机器上的加工时间为 $t(w)$. 关键工序 w 跨机器移动后,最大完工时间是否缩短,与工序组 (x,y) 的空闲时间有关. 工序组 (x,y) 的空闲时间段 $[c^E(x),s^L(y)]$ 与关键工序 w 的空闲时间段 $[c^E(JP[w]),s^L(JS[w])]$ 的交集大于 $t(w)$ 时,关键工序 w 跨机器移动到工序组 (x,y) 之间,才能缩短最大完工时间[10],即关键工序 w 的跨机器移动条件为

$$[c^E(x),s^L(y)]\cap[c^E(JP[w]),s^L(JS[w])]>t(w)$$

由图 4 可知,$t(w)=2$,$[c^E(x),s^L(y)]\cap[c^E(JP[w]),s^L(JS[w])]=3>t(w)$,说明可

以将关键工序 w 移动到工序组(x,y)之间. 对于工序组(a,b),有:

$$[c^E(a),s^L(b)]\cap[c^E(JP[w]),s^L(JS[w])]=2=t(w)$$

对于工序组(e,f),有:

$$[c^E(e),s^L(f)]\cap[c^E(JP[w]),s^L(JS[w])]=1<t(w)$$

因此,无论是将关键工序 w 移动到工序组(a,b),还是工序组(e,f),都不能缩短最大完工时间.

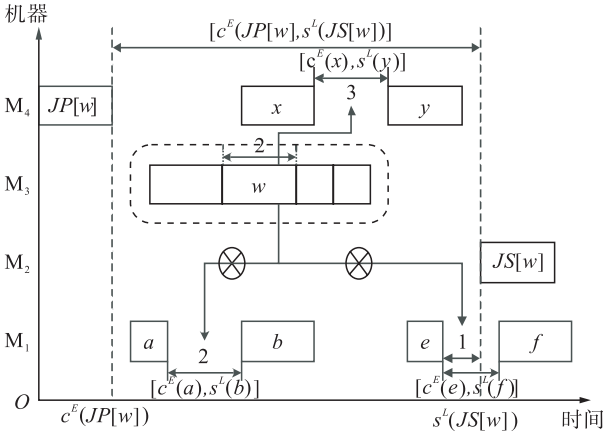


图 4　跨机器移动工序邻域操作示意图

Fig. 4　Schematic diagram of neighborhood operation of cross-machine moving process

2.2.3　同机器移动工序的邻域结构　对于同一台机器上的工序移动,本文采用邻域结构,该结构同样是基于关键路径上的关键块进行操作的,具体的操作步骤如图 5 所示[11]. 由图 5 可以看出,若第一个关键块有两道以上的关键工序,那么仅交换块尾的两道工序;若最后一个关键块有两道以上的关键工序,那么仅交换块首的两道工序;若它们只有两道关键工序,那么仅交换这两道工序;其他的关键块既要交换块首的两道工序,又要交换块尾的两道工序;若进行交换的两道工序属于同一个工件,则不进行任何操作;若关键块中只有一道关键工序,则不进行任何操作.

2.3　混合遗传算法设计

2.3.1　染色体编码和解码　遗传算法实现过程中首先要解决的问题是编码方式的选用,本文采用文献[12]的染色体编码方法,由工序排序和机器选择两部分共同构成一条染色体,工序排序部分确定工件的工序排列顺序,机器选择部分确定每道工序对应的加工机器. 云制造环境下柔性作业车间调度问题的染色体编码如图 6 所示,其中染色体长度是 12,前部分染色体为工序排列,后部分染色体为机器选择,染色体长度为总工序数的 2 倍. 工序染色体为[2 3 3 1 2 1],其中 2 个“1”表示工件的两个工序,第一个“1”表示工件 I_1 的工序 O_{11},第二个“1”表示工件 I_1 的工序 O_{12},因此,每个工件的工序加工顺序为 $O_{21}\rightarrow O_{31}\rightarrow O_{32}\rightarrow O_{11}\rightarrow O_{22}\rightarrow O_{12}$;机器染色体为[3 5 2 4 1 2],表示工序 O_{11} 的加工机器是 M_4,工序 O_{12} 的加工机器是 M_6,工序 O_{21} 的

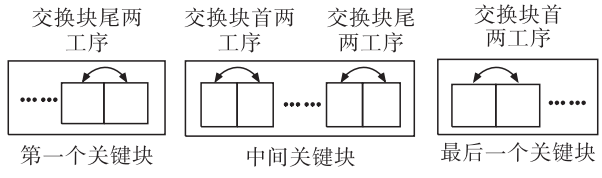


图 5　同一机器移动工序邻域操作示意图

Fig. 5　Schematic diagram of the neighborhood operation of the same machine moving process

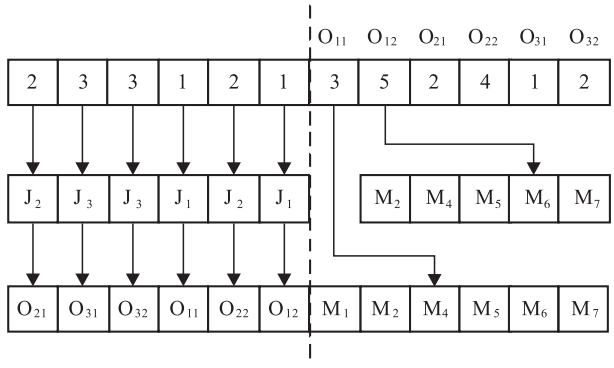


图 6　云制造环境下柔性作业车间调度问题的染色体编码方案

Fig. 6　Chromosome coding scheme for flexible job shop scheduling in cloud manufacturing environment

加工机器是 M_2,工序 O_{22}的加工机器是 M_5,工序 O_{31}的加工机器是 M_2,工序 O_{32}的加工机器是 M_5.

本文采用一种插入式贪婪解码方法进行求解[13]:先按照染色体上工序顺序安排和加工机器选择结果进行解码,加工染色体上的第一道工序;再确定已选加工机器的最佳开始时间,将染色体上的第二道工序插入其中;以此类推,将染色体上的所有工序都插入到合适位置.

2.3.2 适应度函数选取 适应度函数一般由目标函数线性变换得到,它直接影响算法性能的好坏. 本文采用基于排序的适应度分配规则(ranking 函数)分配适应度值,最大完工时间($C_{\max}$)越小,个体的适应度值越大.

2.3.3 选择操作 选择操作是将适应度值高的个体以一定的概率遗传到子代,其作用是避免有效基因的缺失,使高性能的基因遗传到子代的概率更大,从而提高算法的全局收敛能力和运算效率. 本文采用轮盘赌法(rws 函数)来进行选择操作,假设种群的个数为 N,个体 i 的适应度值为 f_i,则个体 i 的选择概率为

$$P_i = f_i \Big/ \sum_{k=1}^{N} f_k$$

在确定了个体的选择概率后,被选择的个体由产生[0,1]之间的均匀随机数来决定. 个体的选择概率大,就会有更多的机会被多次选中,那么这些个体的遗传基因就会在种群中扩大;个体的选择概率小,则会增大被淘汰的概率.

2.3.4 交叉操作 交叉操作在遗传算法中的作用非常重要,交叉操作可以挑选种群中质量较好的基因并将其保留下来;通过基因的交叉重组同样能够产生新的个体,令遗传算法具备较好的全局搜索能力. 本文采用两种交叉方式:一是基于工序排序的交叉操作;二是基于机器选择的交叉操作.

工序排序部分的交叉操作如图 7 所示,其过程描述[14]为:首先随机将工件集合分成两个子集 G_1 和 G_2;然后将父代 F_1 中包含的子集 G_1 复制到子代 C_1 中相同的位置,将父代 F_2 中包含的子集复制到子代 C_2 中相同的位置;最后将父代 F_1 中包含的子集 G_1 填充到子代 G_2 中空缺的位置,父代 F_2 中包含的子集 G_2 复制到子代 C_1 中空缺的位置,基因顺序不变.

机器选择部分的交叉操作如图 8 所示,其过程包括[15]:先随机选定机器染色体中的两个交叉点,再将父代 F_1,F_2 相对应位置的染色体基因相互交换,保持其他机器的工序加工顺序不变,直到整个交叉操作全部完成.

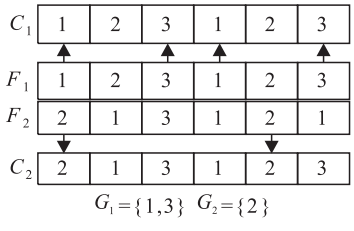


图 7 工序排序部分的交叉操作

Fig. 7 Cross operation of operation sequencing

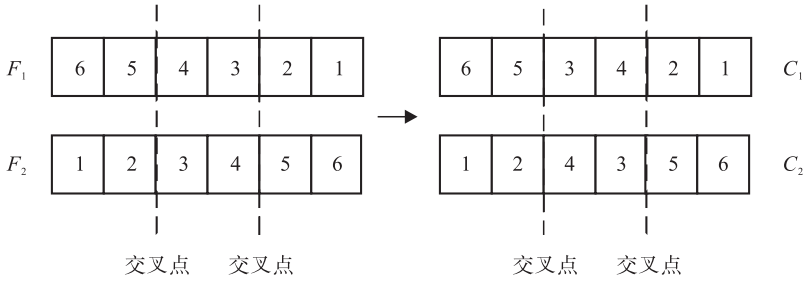


图 8 机器选择部分的交叉操作结果

Fig. 8 Cross operation results of machine selection part

2.3.5　变异操作　本文采用一种简单有效的染色体变异方式,变异操作分两步[16]:第一步,在工序排序部分的染色体中随机选取多个基因并将其互换位置,工序排序部分染色体的编码方式可以避免变异后无效解的产生;第二步,在机器选择部分的染色体中随机选取要变异的基因,集中在相应工序的可选加工机器中,替换原来的加工机器,并且保证选取的机器与原来的加工机器不同.

2.3.6　混合遗传算法流程　混合遗传算法流程如图 9 所示,其具体步骤如下.

步骤 1　进行参数设置.

步骤 2　进行初始化,随机产生 N 个个体.

步骤 3　主动解码,计算和评价种群中个体的适应度值.

步骤 4　若迭代次数达到设定的值,则终止运算并输出结果;否则执行步骤 5.

步骤 5　执行选择操作,用轮盘赌的方法产生子代种群.

步骤 6　执行交叉操作.

步骤 7　执行变异操作.

步骤 8　主动解码,计算和评价个体适应度值,确定关键工序和关键路径.

步骤 9　对种群中 20% 的个体进行变邻域搜索求得较优解,将较优的个体取代当前较差的个体;否则,保持原本个体不变.

步骤 10　生成新种群,再执行步骤 3.

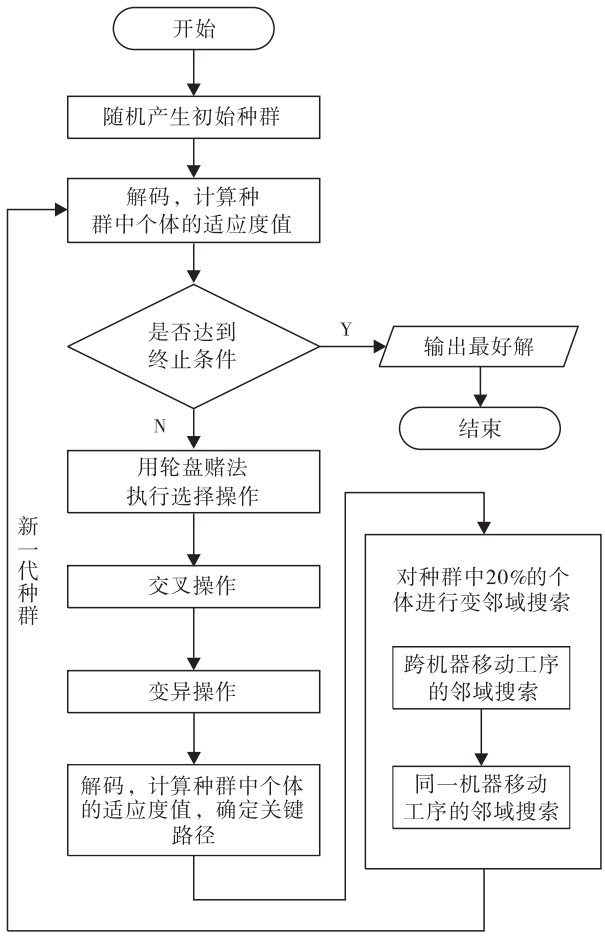


图 9　混合遗传算法流程图

Fig. 9　Flow chart of hybrid genetic algorithm

3　实验结果与分析

为了验证本文混合遗传算法的有效性,采用 C + + 编程语言编写了算法的实现代码,运行环境:处理器为 Intel(R) Core(TM) i5 - 4590 CPU @ 3.30 GHz,内存(RAM)为 8.00 GB. 实验参数设置如下:初始种群规模 400—1200;最大迭代次数 200—800;交叉概率 0.8;变异概率 0.02 ~ 0.3;选取 Brandimarte 标准算例和Kacem 标准算例(共 16 个标准算例)进行测试;安排 n 个工件在 m 台机器上进行加工($n = 4, 5, \cdots, 20$, $m = 4, 5, \cdots, 15$);为防止完工时间过早或过晚而产生滞留惩罚成本或拖期惩罚成本,设置合适时间上下限. 将本文算法与文献[4]算法、文献[6]算法和文献[7]算法进行对比,它们的 Brandimarte 和 Kacem算例结果见表 1.

由表 1 可知,本文算法在求解 MK1—MK16 这 16 个算例时,结果更为理想,最大完工时间 $C_{\max}$ 接近或达到历史最优解,其中 MK1—MK3 和 MK8—MK16 这 12 个算例达到历史最优解;MK4—MK7 这 4 个算例接近历史最优解,与其他 3 种算法相比,本文算法的平均相对误差最小,仅为 1.34%. 这表明,本文算法优化效果更为明显,容易求得最优解.

表 1 不同算法的 Brandimarte 和 Kacem 算例结果

Table 1 Example results of Brandimarte and Kacem of different algorithms

算例	n 个工件 × m 台机器	时间下限/min	时间上限/min	C_{best}/min	文献[4]算法		文献[6]算法		文献[7]算法		本文算法	
					C_{max}/min	相对误差/%	C_{max}/min	相对误差/%	C_{max}/min	相对误差/%	C_{max}/min	相对误差/%
MK1	10 × 6	36	42	40	40	0.0	40	0.0	40	0	40	0.0
MK2	10 × 6	24	32	26	26	0.0	26	0.0	27	3.7	26	0.0
MK3	15 × 8	204	211	204	204	0.0	214	4.9	204	0	204	0.0
MK4	15 × 8	48	81	60	68	13.0	65	8.3	64	6.7	62	3.0
MK5	15 × 4	168	186	173	172	0.0	179	3.9	175	1.8	174	1.1
MK6	10 × 15	33	86	58	60	5.3	58	1.8	65	14.0	63	5.0
MK7	20 × 5	133	157	144	139	0.0	145	4.3	144	3.6	145	4.3
MK8	20 × 10	523	625	523	523	0.0	523	0.0	523	0	523	0.0
MK9	20 × 15	299	369	307	307	0.0	307	0.0	309	0.7	307	0.0
MK10	20 × 15	165	296	198	206	9.6	197	4.2	234	23.8	198	0.0
MK11	4 × 6	17	17	17	17	0.0	17	0.0	17	0	17	0.0
MK12	6 × 6	33	33	33	33	0.0	33	0.0	33	0	33	0.0
MK13	6 × 10	44	44	44	44	0.0	44	0.0	44	0	44	0.0
MK14	8 × 8	14	14	14	14	0.0	14	0.0	14	0	14	0.0
MK15	10 × 10	7	7	7	7	0.0	7	0.0	7	0	7	0.0
MK16	15 × 10	11	11	11	11	0.0	11	0.0	11	0	11	0.0
平均相对误差/%						1.74		1.71		3.39		1.34

为了验证应用本文算法求解云制造环境下的柔性作业车间调度产生的新问题的有效性,设定调度的初始时间从 0 开始,即车间从时间 0 开始加工工序,并设定 5 ~ 10 min 时间段内,机器 M_2 被车间常规任务锁定,不能用于加工云订单任务;在时间段 20 ~ 30 min 内,机器 M_4 被车间常规任务锁定. 实验参数设置交货期时间为 90 min,云平台上的收益 $P = 0.5$,拖期惩罚成本 $C = 0.1$,工件数为 10,工序数为 6,机器数为 6. 分别运行本文算法、文献[12]算法和文献[13]算法 100 次后,得到实验平均解分别为 101 min,105 min,103 min,得到最优解分别为 96 min,100 min,98 min. 由此可见,用本文算法求解云制造环境下柔性作业车间问题更为高效,且有效解决了对云订单进行生产调度安排需要避开被锁定时间段的问题,虽然云订单任务完工时间大于给定的交货期,但云订单上的收益 P 大于惩罚成本 C,表明企业是有收益的.

4 结语

本文针对云制造环境下柔性作业车间调度产生的离散型加工设备的空闲时间利用及自身任务与云平台任务冲突问题,构建云制造作业车间调度框架,建立了以云订单拖期产生的最小惩罚成本为优化目标的数学模型,设计了最小拖期惩罚成本的目标函数,提出了一种基于混合遗传算法求解机器被占用情况的车间调度方法. 该方法将云平台上接受的生产任务与车间下达的常规任务一起进行,采用了基于工序排序的编码方法和基于机器选择的编码方法,在进行云任务调度设计时避开被车间自身任务所占用的时间段,求解工件最优调度顺序,保证云订单收益大于拖期产生的惩罚成本. 通过基准算例进行测试,验证了所提方法的可行性和高效性.

该研究充分利用机器设备的空闲时间,合

理调度接受的云平台生产任务,实现车间自身生产任务和云平台任务协同生产,提高了企业的收益和资源利用率,为云制造环境下柔性作业车间调度问题提供了一种新的解决思路,对云制造企业的经济增长和资源合理利用具有重要意义. 今后将会考虑加入多个目标函数研究云制造环境下柔性作业车间调度问题,并进一步改进算法设计出更有效的多目标优化方法.

参考文献:

[1] 许春安,李芳. 面向云制造服务的制造资源优化配置研究[J]. 工业工程,2019,22(3):44.

[2] 李鹏飞,李海波. 云制造环境下基于功能需求的资源发现方法[J]. 微型机与应用,2014,33(8):71.

[3] XU X. From cloud computing to cloud manufacturing[J]. Robotics and Computer-Integrated Manufacturing,2012,28(1):75.

[4] 王贞,张纪会,齐元青. 具有空闲时间的云制造作业车间调度方法[J]. 控制与决策,2017,32(5):811

[5] 张国辉,石杨. 基于改进遗传算法求解柔性作业车间调度问题[J]. 机械科学与技术,2011,30(11):1890.

[6] 张超勇,饶运清,刘向军,等. 基于 POX 交叉的遗传算法求解 Job-Shop 调度问题[J]. 中国机械工程,2004(23):83.

[7] 何林燕. 云制造环境下柔性作业车间调度算法的研究[D]. 哈尔滨:哈尔滨理工大学,2017.

[8] 王超. 基于混合遗传禁忌搜索算法的多目标柔性作业车间调度问题研究[D]. 重庆:重庆大学,2012.

[9] NORMAN B A,BEAN J C. Random keys genetic algorithm for job-shop scheduling[J]. Engineering Design & Automation,1997,3(2):145.

[10] 刘志虎. 基于改进蚁群算法的柔性车间调度研究[D]. 芜湖:安徽工程大学,2016.

[11] 杜文丽,原亮. 遗传算法的特点及应用领域研究[J]. 科技信息(科学教研),2008(10):31.

[12] 高亮,张国辉,王晓娟. 柔性作业车间调度智能算法及其应用[M]. 武汉:华中科技大学出版社,2012.

[13] GEN M, TSUJIMURA Y, KUBOTA E. Solving job-shop scheduling problems by genetic algorithm[C]// Proceeding of IEEE International Conference on Systems, Man and Cybernetics. Piscataway: IEEE Conference Publications, 1994:1577.

[14] 王岚. 基于自适应交叉和变异概率的遗传算法收敛性研究[J]. 云南师范大学学报(自然科学版),2010,30(3):32.

[15] ALTER T B,BLANK L M,EBERT B E. Genetic optimization algorithm for metabolic engineering revisited[J]. Metabolites,2018,8(2):33.

[16] ZOBOLAS G I, TARANTILIS C D, IOANNOU G. A hybrid evolutionary algorithm for the job shop scheduling problem[J]. The Journal of the Operational Research Society, 2009, 60(2): 221.