Journal of Computer Applications
计算机应用,2020,40(1):292 - 298
ISSN 1001-9081
CODEN JYIIDU
2020-01-10
http://www.joca.cn

文章编号:1001-9081(2020)01-0292-07
DOI:10.11772/j.issn.1001-9081.2019060981

染缸排产建模及滑动时间窗启发式调度算法

隗千千,董兴业*,王焕政
(北京交通大学 计算机与信息技术学院,北京 100044)
(*通信作者电子邮箱 xydong@bjtu.edu.cn)

摘　要:针对染缸排产问题约束复杂、任务规模大、排产效率要求高的特点,为了提高问题模型和算法在实际场景中的适用性,建立了染缸排产增量调度模型,提出了滑动时间窗启发式调度(STWS)算法。该算法以最小化延误代价、洗缸成本、染缸切换成本为优化目标,使用启发式调度规则,按照优先级顺序调度产品;对于每个产品的调度,先用动态拼缸算法和拆缸算法进行批次划分,然后调用批次最佳排序算法调度批次。使用某染纱企业车间实际生产数据仿真调度,所提算法可在10 s内完成月度计划的调度。相对于人工排产方式,所提算法提高了排产效率,显著优化了三个目标,在增量调度中洗缸成本和染缸切换成本也有明显优化。实验结果表明所提算法具有很好的调度能力。

关键词:染缸排产;启发式算法;增量调度模型;异构并行机;批处理调度
中图分类号:TP18;TP301　　**文献标志码**:A

Modeling of dyeing vat scheduling and slide time window scheduling heuristic algorithm

WEI Qianqian, DONG Xingye*, WANG Huanzheng
(*School of Computer and Information Technology, Beijing Jiaotong University, Beijing* 100044, *China*)

Abstract: Considering the characteristics of dyeing vat scheduling problem, such as complex constraints, large task scales, high efficiency request, an incremental dyeing vat scheduling model was established and the Slide Time Window Scheduling heuristic (STWS) algorithm was proposed to improve the applicability of the problem model and the algorithm in real scenario. In order to meet the optimization target of minimizing delay cost, washing cost and the switching cost of dyeing vat, the heuristic scheduling rules were applied to schedule the products according to the priority order. For each product scheduling, the dynamic combination batch algorithm and the batch split algorithm were used to divide batches, and then the batch optimal sorting algorithm was used to schedule the batches. The simulated scheduling results on actual production data provided by a dyeing enterprise show that the algorithm can complete the scheduling for monthly plan within 10 s. Compared with the manual scheduling, the proposed algorithm improves the scheduling efficiency and significantly optimizes three objectives. Additionally, experiments on incremental scheduling show obvious optimization of the algorithm on reducing the washing cost and the switching cost of dyeing vat. All the results indicate that the proposed algorithm has excellent scheduling ability.

Key words: dyeing vat scheduling; heuristic algorithm; incremental scheduling model; heterogeneous parallel machine; batch scheduling

0　引言

染缸排产是为多染缸、多订单、分批次的染色任务制定生产计划的问题,需要考虑生产订单分配、工艺约束、订单要求和客户要求等限制因素,是一个组合优化问题。由于缺乏可用的软件工具,我国的印染车间制定染缸排产计划时很大程度上还依赖于计划人员的经验。其染色工艺约束复杂,生产任务规模大,给计划人员带来了巨大的压力,排产方案的质量也难以保证。

近年来,许多学者在染缸排产模型及调度算法方面作了许多研究。通常,染缸排产模型主要考虑完成时间、洗缸成本、染缸切换成本等。文献[1]以最小化染色成本、洗缸成本、库存成本、延误代价为优化目标,提出了一种基于启发式规则的改进极值优化算法用于求解小规模问题;文献[2]考虑了产品拆分分批,最小化产品延误代价和切换成本,使用遗传算法求解;文献[3]考虑了订单的拆分和合并分批,以最小化延误时间和最大化染缸利用率为目标,设计了构造型启发式方法;文献[4]以延误代价和染缸利用率为目标,考虑相容作业组的拼缸、与作业顺序相关的设置时间,构建了问题模型并提出了一种多目标人工蜂群算法;文献[5]针对机器相同的排产调度问题,考虑产品分批、与作业顺序相关的设置时间等因素,以最小化完工时间为目标,设计了混合整数线性规划模型,将产品的调度过程分为批次构建和批次调度两个子问题,使用多种群遗传算法求解;文献[6]在染缸容量上限和下限约束、订单优先级约束、洗缸约束等约束条件下,使用量子遗传算法,对订单进行拆分与合并分批,可完成小规模问题的染缸排产调度。除了针对印染问题的研究,与之最相似的问题是批处理调度问题[7-10]。该类问题应用背景广泛,包括半

收稿日期:2019-06-12;**修回日期**:2019-08-06;**录用日期**:2019-09-06。

作者简介:隗千千(1995—),女,河北保定人,硕士研究生,主要研究方向:调度理论和算法、运筹优化、智能优化算法;　董兴业(1974—),男,河南漯河人,副教授,博士,主要研究方向:调度理论和算法、运筹优化、智能优化算法;　王焕政(1995—),男,山东烟台人,硕士研究生,主要研究方向:调度理论和算法、运筹优化、智能优化算法。

导体生产[11]、钢铁制造[12]、轮胎制造[13]、印刷线路板生产[14]等。若简化约束条件,染缸排产调度问题属于异构并行机、作业大小不同、作业组不相容、设置时间与作业顺序相关的批处理调度问题,是 NP-难问题。

实际生产中问题的规模往往很大,约束条件和生产限制因素更加复杂,对排产效率要求也比较高。一般情况下,一个染纱车间的月订单量至少有 500 个,染缸数量有几十个,排产调度时需要考虑产品的特殊工艺(如荧光产品)、头缸生产以及染缸的维护计划等约束。此外,现有生产计划对染缸排产还存在一定的限制。相比之下,上述对染缸排产问题的研究中所构建的问题模型未考虑特殊工艺、头缸生产、染缸维护计划、进出缸并发约束,也未考虑增量调度,与实际生产条件相差较大,限制了算法的适用性;另外,虽然文献中批处理调度问题的研究成果较多,但是这些算法不能用于本文所提出的问题模型。综上所述,染缸排产模型及优化算法在实用性和适用性方面亟待提高。

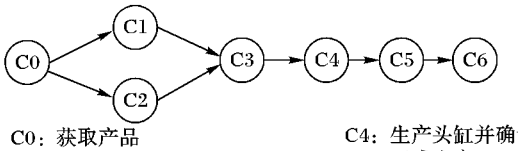
本文根据印染企业的生产需求,以最小化延误代价、染缸切换成本、洗缸成本为目标,在已有问题模型的基础上,考虑产品头缸、禁荧光、染缸维护、现有生产计划安排等新的生产约束,建立更符合实际生产条件的染缸排产增量调度模型,并基于启发式规则设计求解算法,提高排产效率,节约生产成本。

1　染缸排产调度问题模型

1.1　问题描述

染缸排产调度问题是通过对产品进行拆分分批或者合并分批将多个产品安排到多台染缸中排产,需解决的问题是在不改变原生产计划中产品批次划分的情况下,对新增产品进行批次划分,为各批次安排染缸,确定批次开始加工时间,在满足各种约束条件的同时使调度目标最优。

印染企业的生产订单首先由销售部门下发给计划部门,然后计划部门根据实际生产情况制定生产计划,并传达给生产车间。每个产品从进入企业资源计划系统到包装入库的处理过程可表示为如图 1 所示的顶点活动(Activity On Vertex, AOV)图。产品调度仅需要确定 C4 和 C5 的开始时间;由于排产过程不涉及 C0 和 C6,因此它们可看作虚节点,时长为 0 天,其他节点的时长与具体产品有关;当产品没有头缸时,不需要工序 C4。



C0:获取产品　　C4:生产头缸并确认
C1:向客户确认颜色　　C5:正式生产
C2:原料备库　　C6:包装入库
C3:生产复样并确定复样配方卡

图 1　生产流程 AOV 图

Fig. 1　AOV diagram of production process

染缸排产调度问题的约束条件包括以下几个方面:

1)染缸容量限制约束。批次加工量必须在所选染缸容量上限和下限之间,相同的容量上限为同一缸型。

2)染缸维护时间约束。染缸有多个维护时间段,维护期间染缸不可用。

3)加工时间约束。图 1 中各工序在执行前,其前序工序必须完成,否则应延迟排产,延迟时间为其前序工序的最大准备时长。

4)禁荧光约束。染缸加工荧光产品后需连续加工一定缸数的非荧光产品之后才能加工禁荧光产品。

5)进出缸并发约束。批次在开始加工前和加工结束后有进缸和出缸操作,同一时刻执行进缸和出缸操作的数量有一个上限,该限制和工人资源量有关。

6)拼缸约束。几个产品合并在一个批次加工时,称为拼缸;拼缸批次的各产品必须来自同一产品组。

7)拆缸约束。一个产品拆分成几个小的批次加工以适应染缸容量,称为拆缸。为了保证大货产品的生产质量,不同量级的产品的最小拆分阈值不同,拆分的各个批次中,最多有一个批次的生产量小于该拆分阈值,拆缸需尽可能拆分成大的批次;原生产计划中已经拆缸的批次不允许再次拼缸;头缸不允许拆缸。

8)原生产计划中已经拆缸或者拼缸的产品,再次调度时不再进行批次划分。

本文首先对数据作了预处理:依次检查产品的各项工序,计算产品的延迟排产时间,记为产品的释放时间。例如,某产品的 C1、C2、C3 工序未完成,则其延迟排产天数为上述三工序的最大准备时长。由加工时间约束可知,产品的最早加工时间应不早于释放时间。

1.2　模型构建

1.2.1　符号定义

1)集合和索引。

M 为染缸集合,索引 m;J 为产品集合,索引 i、j;B 为批次集合,索引 b、c;J^b 为批次 b 的产品集合,B^j 为产品 j 的批次集合;F 为产品分组集合,索引 f,J^f 为组 f 的产品集合;FL_J 为荧光的产品集合,FL_B 为荧光的批次集合,$FL0\_J$ 为禁荧光的产品集合,$FL0\_B$ 为禁荧光的批次集合;SP 为并发时间段集合,SM_m 为染缸 m 的维护时间集合,索引 sp。

2)变量。

u_m 表示染缸 m 的容量上限,l_m 表示染缸 m 的容量下限,b_m^0 表示染缸 m 的初始加工批次,a_m 表示染缸 m 的最早可用时间,n_m^f 表示染缸 m 的连续染非荧光的批次数量。

d_j 为产品 j 的交货日期,r_j 为产品 j 的释放日期,t_j 为产品 j 的染色类型,w_j 为产品 j 的优先级权重,th_j 为产品 j 的最小拆分阈值,f_j 为产品 j 所属产品组,q_j 为产品 j 的总量,q_j^f 为产品 j 的已完成加工量,q_j^h 为产品 j 的头缸生产量,h_j 表示产品 j 头缸完成状态(若产品 j 的头缸已完成,则值为 1,否则为 0)。

m^b 表示批次 b 使用的染缸,s_b 为批次 b 的开始加工时间,e_b 为批次 b 的结束加工时间,t_b 为批次 b 的染色类型;q_{jb} 表示产品 j 在批次 b 中的加工数量,h_{jb} 表示产品在批次中的生产状态(若产品 j 在批次 b 中为头缸,则值为 1,否则为 0)。

$S_{tt'}$ 为从染色类型为 t 到染色类型为 t' 的洗缸时间;P_{ju} 为产品 j 在缸型为 u 的染缸的加工时长;s_{sp} 为时间段 sp 的开始时间,e_{sp} 为时间段 sp 的结束时间,n_{sp} 为时间段 sp 的进出缸并发数量。

3)参数。

Z 表示头缸未确认时延迟排产时长,U 表示批次任务的出缸时长,L 表示批次任务的进缸时长,V 表示荧光批次和禁荧光批次间隔序列长度,P 表示进出缸最大并发数目。

1.2.2　模型表示

在染缸排产建模时,目标函数中主要考虑排产延误、洗缸成本、染缸切换,而模型实用性主要体现在约束条件中。

用 S 表示一个可行解，则每个染缸的执行序列为一个部分解，染缸 m 部分序列可表示为 $S_m=(b_1,b_2,\cdots,b_k)$。同一产品的不同批次之间更换染缸会造成批次的质量差异，产生染缸切换成本，染缸 m 上的批次序列的染缸切换成本为：

$$C1_m=\sum_{k'=1}^{k-1}f_1(b_{k'},b_{k'+1}) \tag{1}$$

其中：

$$f_1(b_{k'},b_{k'+1})=\begin{cases}1, & J^{b_{k'}}=J^{b_{k'+1}}\\0, & J^{b_{k'}}\neq J^{b_{k'+1}}\end{cases} \tag{2}$$

同一染缸的相邻批次之间需要洗缸，洗缸时间与相邻批次的染色类型有关，染色类型有白、浅、中、深四种，一般来说，浅色批次到深色批次的洗缸时间小于深色到浅色批次的洗缸时间。$S_{tt'}$ 表示加工完染色类型为 t 的批次后，在加工染色类型为 t' 的批次之前的洗缸时间。染缸 m 上的批次序列的洗缸成本为：

$$C2_m=\sum_{k'=1}^{k-1}S_{t_{b_{k'}}t_{b_{k'+1}}} \tag{3}$$

每个产品都有交货日期，延期交货会有一定的惩罚，惩罚大小取决于产品的优先级权重和延误时长，产品 j 的延误代价可表示如下：

$$C3_j=\begin{cases}(c_j-d_j)w_j, & c_j>d_j\\0, & \text{其他}\end{cases} \tag{4}$$

其中：

$$c_j=\max\{e_h\mid b\in B^j\} \tag{5}$$

目标函数及约束条件如下：

$$\min\sum_{m\in M}C1_m+\sum_{m\in M}C2_m+\sum_{j\in J}C3_j \tag{6}$$

s.t.

$$l_m\leqslant\sum_{j\in J^b}q_{jb}\leqslant u_m;\ \forall m\in M,b\in S_m \tag{7}$$

$$lap(sp,b)=0;\ \forall m\in M,\ sp\in SM_m,\ b\in S_m \tag{8}$$

$$lap(sp,b)=\begin{cases}1,\text{批次 } b \text{ 加工时间段和时段 } sp \text{ 重叠}\\0,\text{其他}\end{cases} \tag{9}$$

$$\min\{s_b\}\geqslant r_j;\forall j\in J,\ b\in B^j \tag{10}$$

$$In(f0(b,S_m),S_m)-In(b,S_m)\geqslant V;\ \forall m\in M,b,b_0\in S_m,b\in FL\_B \tag{11}$$

$$n_{sp}\leqslant P;\forall sp\in SP \tag{12}$$

$$f_{j_1}=f_{j_2};\forall b\in B,\ j_1,\ j_2\in J^b \tag{13}$$

$$|MB_j|\leqslant 1;\forall j\in J,b\in B^j,MB_j=\{b\mid q_{jb}\leqslant th_j\} \tag{14}$$

$$q_{jb}=q_j^h;\forall j\in J,b\in B^j,h_{jb}=1 \tag{15}$$

$$q_j^f+\sum_{b\in B}q_{jb}=q_j;\forall j\in J,b\in B \tag{16}$$

$$q_{jb},s_b,e_b\in\mathbf{N} \tag{17}$$

式(6)定义了调度目标，是染缸切换成本、洗缸成本和延误代价三项成本的线性和。

式(7)表示染缸容量约束，批次加工量(各产品在该批次中加工量之和)应在染缸容量上限和下限之间；式(8)和式(9)表示染缸维护时间约束，任意批次的加工时间都不与染缸维护时间重叠；式(10)表示加工时间约束，产品的任意批次开始加工时间都晚于释放时间；式(11)表示禁荧光约束，荧光批次 b 与其之后的第一个禁荧光批次之间至少间隔 V 个批次；式(12)表示进出缸并发约束，任意时间段 sp 执行进出缸操作的数量少于 P；式(13)为产品的拼缸约束，批次中加工的产品都属于同一产品组；式(14)为产品的拆缸约束，产品最多有一个批次加工数量小于拆缸阈值；式(15)为头缸拆缸约束，若在批次中加工头缸，则加工量为全部头缸数量，即头缸不允许拆分；式(16)为产品加工数量约束，产品总量等于已完成数量与各个批次加工数量之和；式(17)为变量的整数约束，即时间变量、批次数量均为整数。

问题模型中已知染缸数据、产品数据、现有生产计划、排产参数等数据，求解产品的分批方案和新的排产计划，其基本问题是并行批处理调度问题，已被证明是一个 NP-难问题[15]，因此其衍生问题也是一个 NP-难问题，采用传统的数学规划方法无法有效求解。染缸排产调度问题约束条件复杂，批次划分和调度时间都是离散的，时间跨度较长，染缸和产品的数据规模较大，因此依据问题特点和模型约束，设计构造型启发式方法快速求解是实际生产的迫切要求。

2 染缸排产调度优化算法

为简化问题，作出如下假设：图 1 中工序 C1(向客户确认颜色)、C2(原料备库)、C3(生产复样并确定复样配方卡)的状态随调度日期更新，产品的释放时间也随之更新；批次一旦开始加工，就不能被中断；当一个批次加工完成后，就从产品的原计划生产列表中删除，不再对下次调度产生影响；由于产品的头缸不能拆缸，因此假设所有头缸均存在单独加工的缸型；缸型相同的染缸，其容量下限也相同；批次加工时间精确到分钟，释放日期、交货日期精确到天。

基于以上假设和生产约束条件，本文设计了滑动时间窗启发式调度(Slide Time Window Scheduling heuristic, STWS)算法。图 2 为算法流程，该算法以天为单位推进时间，获取释放日期在时间窗 $workT$ 内的产品(满足约束(10))，根据产品的优先级顺序(交货日期升序、产品优先级权重降序、释放日期升序排序)，依次对产品进行调度(产品调度算法)；每个产品调度完成后，更新染缸状态(最早可用时间 a_m、连续染非荧光批次数量 n_m^f、任务序列 S_m)、产品状态(完成数量 q_j^f、头缸状态 h_j，若完成头缸加工，更新释放日期 r_j 至 Z 天后)、SP 时间段集合。

图 2 中所述产品调度算法是根据染缸的可用状态和当前进出缸并发状态，计算产品 j 最佳的分批方案和批次加工时间，基本步骤如下：

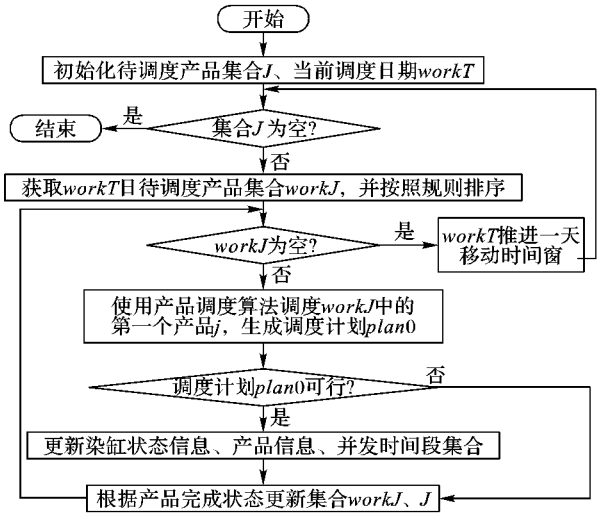


图 2 STWS 算法流程

Fig. 2 Flowchart of STWS algorithm

步骤 1 若产品 j 的原计划批次集合 B^j 不为空，则转至步骤 6；否则生成拼缸调度计划 $PlanC_j$(动态拼缸算法)，即拼缸

批次 b_j。

步骤 2　若 $PlanC_j$ 不存在,则转至步骤 5;否则若 $PlanC_j$ 加工开始时间在 $workT$ 内,$Plan_j \leftarrow PlanC_j$,算法结束。

步骤 3　若 $|J^{bj}|>1$,则生成 $|J^{bj}|\leqslant 1$ 的计划 $PlanC0_j$(动态拼缸算法);否则转至步骤 5。

步骤 4　若 $PlanC0_j$ 存在且不会发生延误,则 $Plan_j \leftarrow PlanC_j$,算法结束;否则若产品当前加工部分不可拆缸,则算法结束。

步骤 5　生成拆缸计划 $PlanS_j$(拆缸算法),令 $Plan_j \leftarrow PlanS_j$。

步骤 6　调整 $PlanS_j$ 批次,选择合适的染缸和加工时间(批次最佳排序算法),优化延误代价、换缸成本、洗缸成本,算法结束。

在使用动态拼缸算法和基于回溯搜索的拆缸算法对产品进行调度时,需要确定产品的待调度数量和产品的最晚加工完成日期。不同状态下的产品调度数量不同,主要和头缸有关,根据约束条件(15)和(16),可得产品 j 的待调度数量 q_j' 的计算方法为:

$$q_j' = \begin{cases} q_j^h, & q_j^h > 0 \text{ 且 } h_j = 1 \\ q_j - q_j^f, & \text{其他} \end{cases} \tag{18}$$

为降低延误代价,产品发生延误时,应该保证延误最短,因此,在产品调度过程中,初次调度最晚加工完成日期(截止日期)均为交货期,在不能成功调度时,最晚加工完成日期每次延后一天,进行迭代调度。

2.1　动态拼缸算法

动态拼缸(Dynamic Combination Batch,DCB)算法首先根据当前调度时间 $workT$ 选择满足并发约束(12)的染缸集合。然后在同一最晚加工时间之前,对于集合中的每个染缸上计算拼缸方案,将该组合问题转化为一维背包问题,使用动态规划算法计算产品 j 的最佳拼缸产品集合,并确定拼缸批次的加工时间,选择第一个可行的拼缸批次返回。若所有拼缸批次都晚于最晚加工时间,则推迟该时间点,重新查找拼缸方案。最后是优化阶段,在可用染缸集合中查找拼缸批次评价指标最小的染缸,确定批次所用染缸及开始加工时间。以下是 DCB 算法伪代码描述。

算法 1　动态拼缸算法。

输入:待调度产品 j;
输出:拼缸批次 b。
计算产品 j 的可拼缸产品集合 $combJ$:
　$combJ=\{j' \mid j'\in J, j'\in J^{fj}, r_{j'}\leqslant workT\}$
计算产品 j 和 $combJ$ 中产品的待调度数量
获得产品 j 可用染缸集合 $workM$,按缸型降序排序:$workM=\{m \mid m\in M, l_m > q_j'\}$(处理约束(11),若产品禁荧光,还应满足 $\forall m\in workM, n_m^f \geqslant V$)
IF($workM$ 为空) RETURN NULL
初始化最晚加工完成日期 $combD \leftarrow d_j$,拼缸标记 $combF \leftarrow$ false,染缸访问序号 $i \leftarrow 0$
WHILE (true)
　WHILE ($i < |workM|$)
　　计算满足容量约束(7)的拼缸产品组合 $combJ_{m_i}$
　　IF ($combJ_{m_i}$ 为空) $i \leftarrow i+1$
　　ELSE
　　计算 b 在 m_i 的加工时间,令 $combF \leftarrow$ true
　　IF ($e_b \leqslant combD$) BREAK
　　ELSE $i \leftarrow i+1$
　END WHILE
　IF($combF$ == false) RETURN NULL
　ELSE
　　IF ($e_b \leqslant combD$) BREAK
　　ELSE 更新 $combD \leftarrow combD+1$, $i \leftarrow 0$
END WHILE
$workM'$ 为 $workM$ 中满足批次 b 容量约束的染缸,计算 b 在 $workM'$ 中各个染缸 m_i 的加工时间,得到新的批次 b_i,则可行批次集合 $comB=\{b_i \mid e_{b_i}\leqslant combD\}$
计算 $b_i \in comB$ 所用利用率 pr_{b_i} 和洗缸代价 cs_{b_i},选择拼缸批次评价指标 C_{b_i} 最小的批次 b;RETURN b

上述染缸利用率和缸批次评价指标的计算方式为:

$$pr_{b_i} = \left(\sum_{j\in J^b} q_{jb}\right)/u_{m_i} \tag{19}$$

$$C_{b_i} = u_{m_i}(1-pr_{b_i}) + cs_{b_i} \tag{20}$$

2.2　拆缸算法

2.2.1　基于回溯搜索的拆缸算法

由于产品拆缸有最小阈值限制,并且产品拆分时尽可能拆分为大的批次,若使用简单贪心算法直接选择最大容量的染缸进行拆分,会产生很多拆缸方案不可行的情况,因此本文提出基于回溯搜索的拆缸算法(Batch split Algorithm based on Backtracking Search, BABS)。

BABS 是一种隐枚举算法,计算满足约束条件最佳拆缸方案,由大到小选择染缸,在每个染缸上尽可能地续缸生产,直至拆缸不可行,此时删除当前方案中已安排的最小染缸,回溯查找可用染缸,然后继续分批,其伪代码描述如下。

算法 2　基于回溯搜索的拆缸算法。

输入:待调度产品 j;
输出:拆缸方案 $PlanS_j$。
计算产品 j 的待调度数量 q_j'
获得可用染缸 $workM$,按缸型降序、可用时间升序排序
初始化最晚加工完成日期 $splitD \leftarrow d_j$
初始化基本变量:可延误拆缸标记 $splitF \leftarrow$ false,拆分批次集合 $PlanS_j \leftarrow$ NULL,所用染缸的可加工最大量 $uC \leftarrow 0$、最小量 $lC \leftarrow 0$,染缸访问序号 $i \leftarrow 0$
WHILE (true)
　从 i 开始顺次查找第一个满足式(21)或(22)的染缸 m_i
　IF (m_i 不存在)
　　WHILE ($PlanS_j$! = NULL)
　　回溯查找新的可用染缸 m_i; BREAK
　　END WHILE
　　IF (m_i 不存在)
　　IF ($splitF$ == true)　　//可延期拆缸
　　$splitD \leftarrow splitD+1$,重新初始化基本变量
　　CONTINUE
　　ELSE 产品拆缸失败,算法异常结束
　WHILE (true)
　　IF (m_i 满足式(21)或(22))
　　在染缸 m_i 上生成新的批次 b
　　ELSE BREAK
　　IF ($e_b > splitD$)　　//可通过延期拆缸
　　更新 $splitF \leftarrow$ true, $i \leftarrow i+1$; BREAK
　　ELSE
　　$lC \leftarrow l_{m_i} + lC$, $uC \leftarrow u_{m_i} + uC$
　　$PlanS_j \leftarrow PlanS_j \cup b$
　END WHILE
　IF ($lC \leqslant q_j' \leqslant u_{m_i}$) BREAK
END WHILE

调整 $PlanS_j$ 批次的加工数量,使相同缸型的批次加工数量尽可能均匀;RETURN $PlanS_j$

回溯查找染缸的方法:首先删除 $PlanS_j$ 最后加入的批次 $newb$,然后令 $i \leftarrow 0$,从 i 开始顺次查找第一个比 $newb$ 所用缸型小且满足式(21)或式(22)的染缸 m_i。

染缸可用条件判断式:

$$l_{m_i} + lC \leqslant q_j' \leqslant u_{m_i} + uC \tag{21}$$

$$l_{m_i} \geqslant th_j \text{ 并且 } l_{m_i} + lC \leqslant q_j' \tag{22}$$

2.2.2 拆缸优化算法

上述基于回溯搜索的拆缸算法中的基本准则是优先选择大缸,查找满足染缸容量约束和产品拆缸约束的拆缸组合方案,这种方式得到的拆缸方案大多使用染缸的缸型种类比较多,一个产品需要在多个染缸上生产,染缸切换成本比较高,产品质量稳定性也变差,因此最佳的拆缸方案是在较大的缸中续缸(而不是选择最大的染缸和较小的染缸拆缸)。

拆缸优化(Batch Split Optimization,BSO)算法是在基本拆缸方案的基础上,保证最晚加工完成时间相同、使用拆缸批次数量相同和优先选用大缸的准则,优化洗缸成本和染缸切换成本。每一轮优化都是在同一个最晚加工完成时间前,将生产量均分到较小的染缸中,如果不可行,则保留最大的一个批次,均分剩余加工部分,直到所有批次都被保留或者得到可行方案。以下是算法的伪代码描述。

算法 3 拆缸优化算法。

输入:基础拆缸方案 $PlanS_j$;
输出:优化拆缸方案 $OPlanS_j$。
根据计算产品 j 的待调度数量 q_j'
获得可用染缸 $workM$,可用缸型 $typeM$,按缸型降序、可用时间升序排序
根据 $PlanS_j$ 计算最晚加工完成日期 $splitD = \max\{e_b \in PlanS_j\}$
初始化待拆缸数量 $n \leftarrow |PlanS_j|$,所用染缸的可加工最大量 $uC \leftarrow 0$、最小量 $lC \leftarrow 0$;将 $PlanS_j$ 按照缸型从大到小排序
IF ($n == 1$) RETURN $PlanS_j$
WHILE ($|PlanS_j| > 1$ 且 $n > 1$)
  在 $typeM$ 中查找 n 缸可加工 q_j' 的染缸 m:
    $n * l_m + lC \leqslant q_j' \leqslant n * u_m + uC$
  IF (染缸 m 不存在)
    删除 $PlanS_j$ 中使用染缸缸型最大的批次 $b_{\max}$,
      $OPlanS_j \leftarrow OPlanS_j \cup b_{\max}$
  ELSE
    在 $splitD$ 之前,在所有缸型为 u_m 的可用染缸中优先续缸加工 n 个批次,加入集合 $OPlanS_j$ 中
END WHILE
调整 $OPlanS_j$ 批次的加工数量,使相同缸型的批次加工数量尽可能均匀;
RETURN $OPlanS_j$

2.3 批次最佳排序算法

使用 BABS 计算产品 j 的拆缸方案 $PlanS_j$($|PlanS_j| > 1$)时,仅考虑交货日期之前每个染缸上的最大加工能力,优先选择了开始可用时间最早的染缸。由于染缸有维护计划,可能发生这种情况:可用时间最早的染缸,其维护时段很长且开始时间很早,进而可用时间段相对其他同缸型的染缸较短,批次选择染缸数目较多。一方面会增加洗缸成本,另一方面也会增加切换成本,导致产品质量不稳定。

在原生产计划中已经拆缸或者拼缸的产品,产品的批次划分已经确定,但是具体的染缸未确定,因此需要确定合适的染缸以及加工时间。

以上两种场景都需要进行批次调度,本文设计了批次最佳排序(Batch Optimal Sorting,BOS)算法,调整批次在各个染缸上的生产顺序,使得产品 j 的延误时间最短,染缸切换成本、洗缸成本最小。将产品的各个批次按照缸型分组,对使用同一种染缸缸型的批次,统一在指定缸型的染缸上调度,每一组都搜索一个最长续缸方案。以下是算法步骤描述:

步骤 1 设置最佳方案最晚加工完成日期 $combD$ 为 $\max\{e_b \mid b \in PlanS_j\}$。

步骤 2 将 $PlanS_j$ 中的批次按照批次所使用缸型降序排序,计算 $PlanS_j$ 使用的所有染缸缸型集合 $MType$,设置 $MType$ 访问序号 i 为 0,最佳调整方案为 $OPlanS_j$。

步骤 3 对染缸进行筛选,获得指定缸型染缸 $workM$。其中 $workM = \{m \mid m \in M, l_m = ty, ty \in MType\}$,若是禁荧光产品,还应满足 $\forall m \in workM, n_m^f \geqslant V$。

步骤 4 若访问序号 $i \geqslant |PlanS_j|$,则批次调整完成,算法结束;否则,获得 $PlanS_j$ 中使用染缸缸型为 ty_i 的批次 Bty_i,获得 $workM$ 中缸型为 ty_i 的染缸 Mty_i。

步骤 5 对于每个 $m_k \in Mty_i$,计算 Bty_i 在 m_k 上的最多连续加工数量 CB_k^i、洗缸成本 SB_k^i、延误时间 DB_k^i;选择评价指标 M_k^i 最小的染缸 m_k:

$$M_k^i = \omega_1 DB_k^i + \omega_2 CB_k^i + \omega_3 SB_k^i;\ \omega_1 > \omega_2 > \omega_3 \tag{23}$$

步骤 6 更新 Bty_i 中的 CB_k^i 各批次的所用染缸、加工时间,加入集合 $OPlanS_j$ 中,从 Bty_i 中移除,记录最新的染缸状态、并发状态,从 Mty_i 中移除 m_k;若 Bty_i 为空,则缸型为 ty_i 的批次调度结束,令 $i \leftarrow i + 1$,返回步骤 4,否则返回步骤 5。

2.4 进出缸并发控制策略

由于整个调度过程是按照时间先后顺序进行的,并发时间段的占用不可逆,调度时每个批次可用时间段仅和当前并发状态和染缸状态相关。

上述 DCB 算法、BABS、BSO 算法和 BOS 算法在查找批次在染缸上的可用时间段时,需要严格满足染缸的维护时间约束和并发时间段约束(进出缸时间段同时满足并发约束);产品生成可行计划后,按照批次的时间先后顺序,依次分配到指定染缸上,每分配一个批次,都重新划分时间段,更新集合 SP 以及各时间段的并发数量 n_{sp}。

3 实验评测

本章首先给出了实验所用的参数和数据集,然后将 STWS 算法与人工排产方案进行对比,最后对算法的增量调度进行评测。

本文所使用的产品和染缸数据均从真实数据集中随机抽取,调度日期为 2018 年 4 月 1 日,染色类型均匀分布。

根据产品生产量将产品分为 4 类:小($1 \leqslant q_j \leqslant 100$)、中($100 < q_j \leqslant 700$)、大($700 < q_j \leqslant 1\,200$)、超大($1\,200 < q_j$)。数据集中产品大小描述可由 4 类产品的占比表示,例如(0.42,0.19,0.16,0.23)。同理,根据染缸的缸型将染缸分为 4 类:小($1 \leqslant u_m \leqslant 80$)、中($80 < u_m \leqslant 300$)、大($300 < u_m \leqslant 800$)、超大($800 < u_m \leqslant 1\,200$)。数据集中染缸大小描述由 4 类染缸的占比表示,例如(0.20,0.30,0.35,0.15)。

表 1 列出了算法需要的参数取值列表。

表 2 列出了实验中使用的测试数据集,其初始调度计划均为空。其中第 6 组数据是第 3 组数据的各个产品的交货期整体提前 1 ~5 天所得,其他数据与 3 号相同,第 7、8、9 组数据是对第 4 组数据部分处理得到的:第 7 组数据延迟了产品的释放日期;第 8 组数据减小了可拼缸数目,即在产品数量相同的情况下增加了分组数目;第 9 组数据增加了荧光产品所占比例。

表 1　参数取值列表

Tab. 1　List of parameter values

符号	含义	值
Z	头缸未确认的延迟排产时长	7 d
U	批次任务的出缸时长	30 min
L	批次任务的进缸时长	30 min
V	荧光批次和禁荧光批次间隔序列长度	2
P	进出缸最大并发数目	5

表 2　实验中使用的数据集

Tab. 2　Dataset used in the experiment

数据编号	产品数量	产品分组	交货期	释放日期	产品大小	禁荧光产品数量	荧光产品数	头缸产品数	染缸数量	染缸大小
0	52	47	04-04—04-21	04-04—04-07	(0.42,0.19,0.16,0.23)	5	6	6	20	(0.20,0.30,0.35,0.15)
1	115	63	04-05—04-26	04-01—04-09	(0.23,0.30,0.23,0.24)	27	10	14	71	(0.32,0.28,0.27,0.13)
2	127	63	04-05—04-26	04-01—04-09	(0.21,0.35,0.22,0.22)	28	11	14	71	(0.32,0.28,0.27,0.13)
3	254	63	04-05—04-26	04-01—04-09	(0.21,0.35,0.22,0.22)	56	22	28	71	(0.32,0.28,0.27,0.13)
4	508	63	04-05—04-26	04-01—04-09	(0.21,0.35,0.22,0.22)	112	44	56	71	(0.32,0.28,0.27,0.13)
5	762	92	04-06—05-01	04-01—04-09	(0.21,0.35,0.22,0.22)	168	66	84	71	(0.32,0.28,0.27,0.13)
6	254	63	04-04—04-21	04-01—04-09	(0.21,0.35,0.22,0.22)	56	22	28	71	(0.32,0.28,0.27,0.13)
7	508	63	04-05—04-26	04-03—04-13	(0.21,0.35,0.22,0.22)	112	44	56	71	(0.32,0.28,0.27,0.13)
8	508	100	04-05—04-26	04-01—04-09	(0.21,0.35,0.22,0.22)	112	44	56	71	(0.32,0.28,0.27,0.13)
9	508	63	04-05—04-26	04-01—04-09	(0.21,0.35,0.22,0.22)	112	70	56	71	(0.32,0.28,0.27,0.13)

人工排缸方案一般是由经验丰富的计划员完成,约束执行往往不是很严格,当实际排产发生约束冲突时,只能通过延迟染缸的后续批次或者取消某些批次来缓解,使得排产不能按照原计划实施。这里使用普通规则(General Rule,GR)算法来模拟人工排缸过程,并与 STWS 算法对比。与 STWS 类似,GR 算法是按照产品的优先级顺序(交货日期升序、产品优先级权重降序、释放日期升序排序),依次对产品进行调度;但是 GR 算法在产品调度时没有使用最优调度策略,而是使用贪心方法对产品进行拆缸和拼缸;批次调度时,选择洗缸时间最短的染缸依次调度,不再进行迭代优化。表 3 为 GR 算法的调度结果。

表 3　GR 算法调度结果

Tab. 3　Scheduling results using GR algorithm

数据编号	延误代价	切换成本	洗缸成本/min	批次数量	拼缸数量	染缸利用率/%
0	213	63	1 370	135	1	90
1	2 240	142	1 050	191	8	91
2	2 240	148	1 310	194	12	91
3	4 480	312	2 180	376	30	91
4	15 393	691	9 850	798	64	92
5	46 858	1 113	16 090	1 312	99	92
6	6 793	316	1 760	385	29	91
7	20 096	669	8 800	810	62	92
8	14 353	737	9 410	863	65	92
9	15 817	655	9 370	797	68	92

使用 STWS 进行调度,可得到如表 4 所示的调度结果,其中,延误成本优化比例 = (GR 算法延误成 - STWS 算法延误成本)/GR 算法延误成本,同理可得切换成本优化比例、洗缸成本优化比例。从调度结果可以看到,在数据规模适中(产品数量在 100 到 500 左右)的情况下,该算法可在 10 s 内给出可行方案。相对于 GR 算法,STWS 算法的染缸切换成本、洗缸成本明显减小,批次数量有所减少、拼缸批次有所增加,染缸利用率明显提高,以上结果说明动态拼缸算法和拆缸算法有利于更好地利用染缸的生产能力;在延误代价方面,当数据规模较大时,STWS 算法优化效果显著,最高可优化 50% 以上,数据规模较小时,优化不明显,这是因为产品数据较少时,可优化空间也有限;除第 0 组数据外,洗缸成本均有大幅度降低,说明批次最佳排序算法能有效安排同一产品的不同批次进行续缸生产,从而节约了洗缸成本;对于第 0 组数据,洗缸成本上升的主要原因是批次调度时为增加续缸而选择了首次洗缸成本较高的染缸。综合以上几个方面,STWS 算法性能优于 GR 算法。

表 4　STWS 算法相比 GR 算法的优化结果

Tab. 4　Optimization results of STWS algorithm compared to GR algorithm

数据编号	时间/s	延误成本优化比例/%	切换成本优化比例/%	洗缸成本优化比例/%	批次数量	拼缸数量	染缸利用率/%
0	0.611	0.00	14.29	-16.06	134	1	94
1	1.183	0.00	34.27	60.00	180	12	89
2	1.484	0.00	33.11	67.94	186	15	94
3	3.035	-2.23	28.53	56.42	346	41	94
4	8.387	18.90	22.87	47.31	713	75	95
5	23.232	50.29	12.67	11.19	1 307	84	94
6	3.358	2.51	28.80	31.25	343	41	94
7	8.659	31.19	13.75	32.27	758	76	95
8	8.802	15.84	22.12	37.41	782	69	95
9	8.145	32.01	17.26	40.77	735	74	95

从表 4 的第 4、7、8、9 组数据的调度结果可以看到,在数据规模适中时,无论什么特点的数据,STWS 算法性能都比较稳定,延误代价优化程度稳定在 20% ~30%,切换成本优化程度在 10% ~20%,洗缸成本优化程度在 30% ~50%,总体优化效果都较为明显。

第 4 组数据的产品数据是在第 3 组数据的产品数据的基础上加入了一些新的产品得到的，因此将第 3 组数据的调度计划作为第 4 组数据初始计划用于增量调度测试，表 5 为增量调度结果。相对于第 4 组数据的无初始调度计划的调度结果，增量调度的消耗时间短，这是由于增量调度时调度计划中已经分批的产品不再经过分批过程，从而节约了调度时间。不过，由于已调度过的产品的批次划分不能改变，减小了优化空间，因此各项成本值均有所增加，拼缸批次数量减少，染缸利用率有所降低。

表 5 STWS 算法在第 4 组数据上的增量调度比较

Tab. 5 Incremental scheduling of STWS algorithm on the 4th data

数据编号	时间/s	延误代价	切换成本	洗缸成本/min	批次数量	拼缸数量	染缸利用率/%
4	8.387	12484	533	5190	713	75	95
4*	6.703	16377	620	4827	786	65	93

4 结语

根据印染车间的生产需求，考虑产品的多样化、异构并行机、批次划分约束、头缸约束、禁荧光约束、染缸维护计划等实际场景，建立了染缸排产增量调度模型。本文提出了构造型启发式算法——滑动时间窗调度(STWS)算法，按照一定的规则和逻辑对产品进行调度，最小化延误代价、洗缸成本、染缸切换成本。产品调度过程中，设计了动态拼缸(DCB)算法和拆缸算法(BABS、BSO)进行批次划分，使用批次最佳排序算法调度批次。通过实验对比，检验了 STWS 算法增量调度的性能，在数据规模适中时，STWS 算法能够在 10 s 内给出可行方案，相比人工排产方案，该算法显著地降低了生产成本和延误代价，目前已在某企业的生产系统中上线运营。

由于算法大多数场景都是增量调度模式，批次划分仅限于第一次调度的产品，批次划分确定后，批次的调度是影响生产代价的重要因素，因此批次调度的全局优化是接下来的研究重点。

参考文献 (References)

[1] QI J, HE L. Extremal optimization for a dye vat scheduling problem [C]// Proceedings of the 7th International Conference on Natural Computation. Piscataway: IEEE, 2011: 2065 – 2069.

[2] 戴智杰, 宋执环, 宋春跃. 基于遗传算法的浸染生产排缸策略[J]. 运筹与管理, 2016, 15(2): 149 – 153. (DAI Z J, SONG Z H, SONG C Y. Strategy in dip-dye production scheduling based on genetic algorithm[J]. Operations Research and Management Science, 2016, 15(2): 149 – 153.)

[3] 金锋, 宋士吉, 杨建华, 等. 染整车间染缸优化调度算法研究[J]. 计算机集成制造系统, 2008, 14(3): 543 – 547. (JIN F, SONG S J, YANG J H, et al. Dyeing machine scheduling problem in dyeing and finishing workshop[J]. Computer Integrated Manufacturing System, 2008, 14(3): 543 – 547.)

[4] ZHANG R, CHANG P C, SONG S, et al. A multi-objective artificial bee colony algorithm for parallel batch-processing machine scheduling in fabric dyeing processes[J]. Knowledge-Based Systems, 2017, 116: 114 – 129.

[5] HUYNH N T, CHIEN C F. A hybrid multi-subpopulation genetic algorithm for textile batch dyeing scheduling and an empirical study [J]. Computers and Industrial Engineering, 2018, 125: 615 – 627.

[6] 蒋佳颖, 王万良, 徐新黎, 等. 基于量子遗传算法的染缸排产问题研究[J]. 计算机工程, 2011, 37(21): 159 – 161, 164. (JIANG J Y, WANG W L, XU X L, et al. Study on dye vat scheduling problem based on quantum genetic algorithm[J]. Computer Engineering, 2011, 37(21): 159 – 161, 164.)

[7] 王万良, 范丽霞, 徐新黎, 等. 多目标差分进化算法求解柔性作业车间批量调度问题[J]. 计算机集成制造系统, 2013, 19(10): 2481 – 2492. (WANG W L, FAN L X, XU X L, et al. Multi-objective differential evolution algorithm for flexible Job-Shop batch scheduling problem[J]. Computer Integrated Manufacturing Systems, 2013, 19(10): 2481 – 2492.)

[8] GOKHALE R, MATHIRAJAN M. Minimizing total weighted tardiness on heterogeneous batch processors with incompatible job families[J]. The International Journal of Advanced Manufacturing Technology, 2014, 70(9/10/11/12): 1563 – 1578.

[9] ABEDI M, SEIDGAR H, FAZLOLLAHTABAR H, et al. Bi-objective optimisation for scheduling the identical parallel batch-processing machines with arbitrary job sizes, unequal job release times and capacity limits[J]. International Journal of Production Research, 2015, 53(6): 1680 – 1711.

[10] ZHANG R, CHANG P C, SONG S, et al. Local search enhanced multi-objective PSO algorithm for scheduling textile production processes with environmental considerations[J]. Applied Soft Computing, 2017, 61: 447 – 467.

[11] WANG I L, WANG Y C, CHEN C W. Scheduling unrelated parallel machines in semiconductor manufacturing by problem reduction and local search heuristics[J]. Flexible Services and Manufacturing Journal, 2013, 25(3): 343 – 366.

[12] PAN Q, WANG L, MAO K, et al. An effective artificial bee colony algorithm for a real-world hybrid flowshop problem in steelmaking process[J]. IEEE Transactions on Automation Science and Engineering, 2013, 10(2): 307 – 322.

[13] GOKHALE R, MATHIRAJAN M. Scheduling identical parallel machines with machine eligibility restrictions to minimize total weighted flowtime in automobile gear manufacturing[J]. The International Journal of Advanced Manufacturing Technology, 2012, 60(9/10/11/12): 1099 – 1110.

[14] BILYK A, MÖNCH L. A variable neighborhood search approach for planning and scheduling of jobs on unrelated parallel machines [J]. Journal of Intelligent Manufacturing, 2012, 23(5): 1621 – 1635.

[15] KOVALYOV M Y, SHAFRANSKY Y M. Batch scheduling with deadlines on parallel machines: an NP-hard case[J]. Information Processing Letters, 1997, 64(2): 69 – 74.

WEI Qianqian, born in 1995, M. S. candidate. Her research interests include scheduling theory and algorithm, operation optimization, intelligent optimization algorithm.

DONG Xingye, born in 1974, Ph. D., associate professor. His research interests include scheduling theory and algorithm, operation optimization, intelligent optimization algorithm.

WANG Huanzheng, born in 1995, M. S. candidate. His research interests include scheduling theory and algorithm, operation optimization, intelligent optimization algorithm.