就绪时间受限的负荷单机环境下返工工件重调度方法

郭艳东 [1,2] 王 庆 [1] 黄 敏 [1]

摘 要 研究了返工工件的单机重调度问题. 在初始调度中初始工件带有不同的就绪时间, 优化目标为最小化初始工件等待时间和; 重调度时在满足每个初始工件最大等待时间约束情况下安排返工工件的生产, 优化目标为最小化所有工件等待时间和. 文中首先建立了 RRSM (Rescheduling for reworks on single machine) 问题模型, 并证明其为 NP 难问题. 然后, 提出并证明了三个 RRSM 问题性质, 进而根据诸性质设计了求解 RRSM 问题的动态插入启发式 (Dynamic insert heuristic, DIH) 算法. 证明了应用 DIH 算法能在多项式时间内求得两种特殊 RRSM 问题的最优解. 最后, 分析了 DIH 算法解的特点, 给出了最优解的判定方法, 并通过算例说明了 DIH 算法的有效性.

关键词 重调度, 单机, 返工, 启发式算法, 等待时间

引用格式 郭艳东, 王庆, 黄敏. 就绪时间受限的负荷单机环境下返工工件重调度方法. 自动化学报, 2013, **39**(12): 2100−2110

DOI 10.3724/SP.J.1004.2013.02100

Rescheduling with Release Time to Minimize Sum of Waiting Time Considering Waiting Constraint of Original Loads

GUO Yan-Dong[1,2] WANG Qing[1] HUANG Min[1]

Abstract In this paper, we consider the rescheduling problem to minimize the sum of waiting times of rework jobs and the original loads with release time on a single machine, and the waiting time of each original load is constrained by a value. The problem of rescheduling for reworks on a single machine (RRSM) is formulated and proved to be NP-hard. A dynamic insert heuristic (DIH) algorithm of polynomial-time is designed and proved with three properties. With respect to two special cases of the identical processing time of rework jobs or the machine without idle times in the original schedule, the DIH algorithm can obtain an optimal solution. A discrimination condition is proved for the optimal solution and effectiveness of the DIH algorithm is explained by cases with regard to general RRSM problems.

Key words Rescheduling, single machine, rework, heuristic algorithm, waiting time

Citation Guo Yan-Dong, Wang Qing, Huang Min. Rescheduling with release time to minimize sum of waiting time considering waiting constraint of original loads. *Acta Automatica Sinica*, 2013, **39**(12): 2100−2110

多元化、多样化产品的不断涌出使经典的单机调度问题一直以来成为理论研究和生产实践的热点, 目前已有大量相关的研究成果[1−6]. 精细化的管理促使在生产的每一个环节都成为系统优化的研究对象. 在生产实践中, 经常出现各种复杂情况干扰已经安排好的调度 (称之为初始调度). 这时需对初始调度进行适当地调整, 以便获得适应实际状况的最优或者近优的新调度方案 (称之为重调度). 在生产中, 通常成品工件中会出现不合格工件, 但有一类不合格工件可通过简单的加工处理修复为合格工件 (称之为返工工件). 目前, 很多企业在制定最初生产计划时, 不考虑可返工工件的生产调度或以成品率方面加以考虑; 一是因为通常制定计划考虑的都是无废品的理想情况, 即使有不合格工件也没考虑返工, 或者考虑返工也没上升到计划层面进行管理; 二是在制定最初的生产计划时, 可返工工件的数量和返工处理时间未知, 难以参与计划制定. 例如, 某石英玻璃制品加工企业, 在检测环节经常有部分不合格产品需要返回到车间的焊接工位进行返工处理, 通

收稿日期 2013-03-08 录用日期 2013-06-14
Manuscript received March 8, 2013; accepted June 14, 2013
国家杰出青年科学基金资助项目 (71325002,61225012), 国家自然科学基金资助项目 (71071028,70931001, 71021061,71171040), 高等学校博士学科点专项科研基金优先发展领域资助课题 (20120042130003), 高等学校博士学科点专项科研基金资助课题 (20110042110024), 中央高校基本科研业务费专项资金 (N110204003, N120104001) 资助
Supported by the National Science Foundation for Distinguished Young Scholars of China (71325002, 61225012), National Natural Science Foundation of China (71071028, 70931001, 71021061, 71171040), the Specialized Research Fund of the Doctoral Program of Higher Education for the Priority Development Areas (20120042130003), Specialized Research Fund for the Doctoral Program of Higher Education (20110042110024), the Fundamental Research Funds for the Central Universities (N110204003, N120104001)
本文责任编委 王红卫
Recommended by Associate Editor WANG Hong-Wei
1. 东北大学信息科学与工程学院流程工业综合自动化国家重点实验室 沈阳 110819 2. 渤海大学数理学院 锦州 121000
1. State Key Laboratory of Synthetical Automation for Process Industries (Northeastern University), College of Information Science and Engineering, Northeastern University, Shenyang 110819 2. School of Mathematics and Physics, Bohai University, Jinzhou 121000

常经过简单处理即可达到合格要求. 然而, 由于工艺的要求 (焊接前的零件需加温, 焊接工作必须在不低于一定温度下操作), 到达焊接工位上需操作的正常工件具有不同的就绪时间, 同时要求必须在一定时间内开始加工, 即: 工件等待加工的时间有限制. 再如, 半导体制造业的烧结工位也有此类特征. 在实际生产中, 还有许多诸如此类特征的问题, 因此, 本文对这一类当有可返工工件到达时, 在初始工件满足就绪时间、工艺等约束条件下, 根据生产优化目标对初始工件和返工工件进行重调度的单机问题进行研究. 该研究无论是从环保、节约能源角度, 还是从降低企业成本、提高企业利润的角度都十分有意义.

参照重调度优化问题定义框架[7], 本文研究的问题属于信息确定环境下, 事件驱动策略的重调度问题. 目前针对返工工件重调度问题的研究主要集中在组批调度方面[8−10], 本文将从加工排序角度研究返工工件的重调度问题. 而排序方面和本文研究相关的文献也为数不多, 其中, Wu 等[11] 考虑了目标为最小化完工时间和单中断的加工车间重调度问题; Unal 等[12] 研究了单机环境下顺序依赖族、考虑装设时间的重调度问题. Hall 等[13] 为本领域做了开创性的工作, 他们研究了几种单机重调度问题, 考虑的重调度约束包括: 初始工件的最大次序差异、次序差异和、最长延迟时间、延迟时间和小于某一定值, 优化目标考虑了最小化完工时间和最小化最大完工时间, 同时也考虑了这些目标和打断花费组合的情况, 并给出了相应问题的多项式时间算法或者证明了其为 NP 难问题. 值得一提的是, 几乎所有的研究问题都假设初始调度是最优调度且不存在机器空闲. 进而 Hall 等[14] 又研究了单机环境下的多组新工件打断的重调度问题, 提出了一个分支定界算法, 解决了规模为 1 000 个工件的重调度问题.

基于 Hall 等的约束条件, Yuan 等[15] 研究了单机环境下假设初始调度是以最小化最长等待时间为目标的最优调度, 并且初始工件带有就绪时间的重调度问题. 说明了约束条件为初始工件最长延迟时间或延迟时间和小于某一定值, 目标为最小化最大完工时间的重调度问题为 NP 难问题, 并在多项式时间内解决了约束为初始工件最大延迟数小于某一定值, 目标为最小化最大完工时间的重调度问题. Yang 等[16] 研究了单机环境下新工件的处理时间视为额外花费并反映为价格的情况, 该花费随着新工件处理时间的恶化而增加. 借鉴 Hall 文献提到的约束条件, 优化目标是通过重调度确定新工件的处理时间, 以最小化总花费的凸组合. 他们在多项式时间内解决了最小化工件完工时间和的重调度问题, 并说明了约束条件为延迟数或延迟时间和的重调度问题为 NP 难问题, 提出了相应启发式算法. Zhao 等[17] 研究了 Yuan 等问题的变形问题, 针对初始工件和新工件都考虑工件恶化, 即工件的处理时间依赖于它们的开始时间, 越晚开始其处理时间越长. 约束条件为最大位置或者延迟完工时间小于某一定值, 目标为最小化所有工件的完工时间和, 并提供了多项式时间算法. Hoogeveen 等[18] 研究了单机环境下新工件依赖族、不同族之间工件生产切换需要装设时间的 5 种情况, 约束条件与 Hall 等的约束条件相同, 优化目标是找到最小化最大完工时间和最小化初始工件的最大延迟数的帕累托最优点. 在多项式时间内解决了约束条件为最大延迟数、最长延迟时间小于某一定值, 不同族装设时间相同的重调度问题, 并证明了约束条件为最长延迟时间、延迟时间和小于某一定值, 装设时间差异化的重调度问题为 NP 难问题[18]. 以上文献没有就车间生产中返工工件重调度这一类比较重要的问题进行研究, 而且现有文献中的方法对解决本文问题亦不适用, 因此, 本文就 RRSM (Rescheduling for reworks on single machine) 问题展开研究.

综上所述, 本文根据生产调度实际调整需要, 抽象出返工工件的单机重调度 (RRSM) 问题, 其特点如下: 初始工件带有不同的就绪时间, 初始调度的优化目标为最小化初始工件总等待时间, 在初始调度执行之前, 有一组只需很短处理时间即可修复的返工工件需要加工. 不失一般性, 如果已经有一部分初始工件已经处理, 则把剩余未处理的工件看成是可调整的初始工件, 组成初始调度. 返工工件的就绪时间为 0, 返工工件类型和初始工件同族 (即不考虑装设时间), 根据工艺要求初始工件在就绪之后必须在一定的时间内进行处理. 重调度的优化目标为最小化所有工件的总等待时间和.

1　问题描述

在单机环境下, 已知优化目标为最小化总等待时间和的最优调度 π^* 中, 包含一组就绪时间为 r_i, 数量为 n_O 的初始工件 $J_O = \{j_1^O, \cdots, j_{n_O}^O\}$, 在调度 π^* 执行前, 有一组数量为 n_R, 就绪时间为 0 的返工工件 $J_R = \{j_{n_O+1}^R, \cdots, j_{n_O+n_R}^R\}$ 需要进行调度, 所涉及的工件数为 $n = n_O + n_R$. 在满足初始工件最长等待时间限值 K 的条件下, 获得优化目标为最小化所有工件总等待时间和的重调度 σ. RRSM 问题示意如图 1.

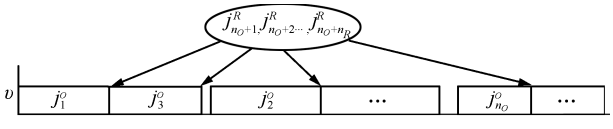


图 1　RRSM 问题

Fig. 1　RRSM problem

1.1 符号定义和问题模型

本文所用符号定义如下: 上角标表示工件类型, 即 O 表示初始工件, R 表示返工工件, 没有标注则表示包括所有初始工件和返工工件. $[\cdot]$ 表示工件在调度中的顺序位置. $(\cdot)$ 内参数表示调度类型, 即 π^* 表示最优初始调度, σ 表示重调度; 例如: $\sum w_i(\sigma)$ 表示重调度中所有工件的等待时间和, $w_i^O(\sigma)$ 表示重调度中初始工件 j_i^O 的等待时间. s_i 表示工件 j_i 的开始时间, p_i^O 表示初始工件 j_i^O 的处理时间, p_j^R 表示返工工件 j_j^R 的处理时间, $p_{\max}^R$ 则表示所有返工工件中最大的处理时间. $\sum F_i(\sigma)$ 表示重调度中所有工件的流水时间和, $\sum C_i(\sigma)$ 表示重调度中所有工件的完成时间和 $\sum F_i(\sigma) = \sum C_i(\sigma) - \sum r_i$. 而 $w_{\max}^O(\pi^*)$ 表示最优初始调度中所有初始工件中最大的等待时间.

RRSM 问题模型:

$$\begin{aligned}
&\min \sum w_i(\sigma), \qquad i \in J\\
&\text{s. t.}\\
&\quad w_i^O(\sigma) \le K, \qquad i \in J^O \qquad (1)\\
&\quad s_i \ge r_i, \qquad i \in J^O \qquad (2)\\
&\quad s_{[i]} + p_{[i]} \le s_{[i+1]}, \quad [i] \in J \qquad (3)
\end{aligned}$$

约束 (1) 说明了每个初始工件的等待时间不超过 K, 约束 (2) 保证初始工件都在就绪时间之后开始加工, 约束 (3) 表示下一个工件只能在当前工件加工完成之后再进行加工. 通过扩展 Graham 等调度优化问题提出的 $\alpha|\beta|\gamma$ 三域表示法[19], RRSM 重调度问题可描述为: $1|r_i : w_i^O(\sigma) \le K|\sum w_i^O(\pi^*) : \sum w_j(\sigma)$, 并满足下列假设条件: 1) $w_{\max}^O(\pi^*) \le K$; 2) 返工工件无等待时间限制, 就绪时间为 0; 3) 返工工件的处理时间不大于初始工件的处理时间, 即 $p_{\max}^R \le p_{\min}^O$; 4) 不考虑工件装设时间.

1.2 问题复杂度分析

RRSM 重调度问题 $1|r_i : w_i^O(\sigma) \le K|\sum w_i^O(\pi^*) : \sum w_j(\sigma)$ 属于典型的组合优化问题. 下面证明其为 NP 难问题.

定理 1. $1|r_i : w_i^O(\sigma) \le K|\sum w_i^O(\pi^*) : \sum C_j(\sigma)$ 重调度问题为 NP 难问题.

证明. 证明可由一个已知的 NP 难问题 — 奇偶分割问题归约得到. 奇偶分割问题描述为: 一组正整数 $\{a_1, \cdots, a_{2t}\}$, 其中, $a_i \ge a_{i+1}$, $1 \le i \le 2t-1$, 并且 $\sum_{i=1}^{2t} a_i = 2A$, 是否存在一个分割将下标集 $S = \{1, \cdots, 2t\}$ 分割成两个子集 S_1 和 S_2, 并且在满足 $\sum_{i \in S_1} a_i = \sum_{i \in S_2} a_i = A$ 的条件下, 集合 S_1 和 S_2 分别包含下标为 $2i-1$ 或者 $2i$ $(i = 1, \cdots, t)$ 的元素之一?

下面考虑 $1|r_i : w_i^O(\sigma) \le K|\sum w_i^O(\pi^*) : \sum C_j(\sigma)$ 重调度问题的一个实例 1:

$$\begin{aligned}
&n_O = 8t,\ n_R = 2t\\
&p_i^O = 2B^{\lceil \frac{i}{2} \rceil} + 2a_i\\
&r_i = \sum_{j=0}^{i-1}(2B^{\lceil \frac{i}{2} \rceil} + 2a_j) + \sum_{j=8t}^{i+8t-1}(B^{\lceil \frac{j-8t}{2} \rceil} + a_{j-8t}),\\
&\qquad\qquad i = 1, \cdots, 2t\\
&p_i^O = 2B^t\\
&r_i = (i-1)B^t + \sum_{j=1}^{t}(24(t-j)+15)B^j +\\
&\qquad \sum_{j=1}^{2t}(12t-6j)a_j + 9A, \qquad i = 2t+1, \cdots, 8t\\
&p_i^R = B^{\lceil \frac{i-8t}{2} \rceil} + a_{i-8t}, \quad i = 8t+1, \cdots, 10t\\
&B = tA^3, \quad K = 0\\
&C = \sum_{j=1}^{t}(60t-24j+15)B^j + (16t^2-3t)B^t +\\
&\qquad \sum_{j=1}^{2t}(30t-9j)a_j + 9A
\end{aligned}$$

下面证明当且仅当奇偶分割问题有解时, 问题实例 1 存在可行解.

(充分性) 初始调度为最优调度 π^*, 根据实例 1 有:

$$\begin{aligned}
&r_1 = 0, \quad p_1^O = 2B + 2a_1\\
&r_2 = 2B + 2a_1 + B + a_1\\
&p_2^O = 2B + 2a_2, \cdots,\\
&r_{2t} = \sum_{j=0}^{2t-1}(2B^{\lceil \frac{j}{2} \rceil} + 2a_j) +\\
&\qquad \sum_{j=8t}^{10t-1}(B^{\lceil \frac{j-8t}{2} \rceil} + a_{j-8t})\\
&p_{2t}^O = 2B^t + 2a_{2t}\\
&r_{2t+1} = \sum_{j=1}^{t}(24(t-j)+15)B^j +\\
&\qquad \sum_{j=1}^{2t}(12t-6j)a_j + 9A\\
&p_{2t+1}^O = 2B^t, \cdots, p_{8t}^O = 2B^t\\
&r_{8t} = (8t-1)B^t + \sum_{j=1}^{t}(24(t-j)+15)B^j +
\end{aligned}$$

$$\sum_{j=1}^{2t}(12t-6j)a_j+9A,\cdots$$

所有初始工件都在就绪时间开始处理, 并产生了 $2t$ 个空闲时间, 分别是:

$$[2B+2a_1,2B+2a_1+B+a_1],\cdots,$$
$$[2B^t+2a_{2t},\sum_{j=1}^{t}(24(t-j)+15)B^j+\sum_{j=1}^{2t}(12t-6j)a_j+9A]$$

而 $2t$ 个返工工件的就绪时间为 0, 处理时间为 $p^R_{8t+1}=B+a_1,p^R_{8t+2}=B+a_2,\cdots,p^R_{10t}=B^t+a_{2t}$, 由实例 1 的已知条件 $K=0$, 在得到的重调度中, 初始工件的开始时间保持不变, 而且都在奇数位置上被调度. 而 $2t$ 个返工工件恰好在每个初始工件之后的空闲时间里进行处理, 即都在偶数位置上被调度. 计算前 4 个位置的工件完成时间和为

$$\begin{aligned}&(2B+2a_1)+(3B+3a_1)+\\&(4B+3a_1+a_2)+(6B+3a_1+3a_2)=\\&15B+11a_1+4a_2=\\&15B+\frac{21a_1}{2}+\frac{9a_2}{2}+\frac{a_1-a_2}{2}\end{aligned}\tag{4}$$

类似地得到:

$$\begin{aligned}&\sum C_i(\sigma)=\\&\sum_{j=1}^{t}(60t-24j+15)B^j+(16t^2-3t)B^t+\\&\sum_{j=1}^{2t}\left(30t-6j+\frac{9}{2}\right)a_j+\frac{\sum_{i\in S_1}a_i}{2}-\frac{\sum_{i\in S_2}a_i}{2}\end{aligned}\tag{5}$$

因为奇偶分割有解, 即 $\sum_{i\in S_1}a_i=\sum_{i\in S_2}a_i=A$, 所以 $\sum C_i(\sigma)=C$.

(必要性) 考虑一个满足条件的可行调度, 初始工件都在就绪时间开始处理, 即得到最优调度, 满足 $K=0$, 所以重调度中初始工件开始时间不变, 并且在奇数位置上被调度. 所有的返工工件只能插入初始调度的机器空闲当中, 或者安排在最后一个初始工件之后进行调度. 根据实例 1, 初始调度中初始工件共产生了 $2t$ 个机器空闲. 若有一个返工工件在最后一个初始工件之后调度, 则借助类似式 (4) 和 (5) 的计算, 可以得到 $\sum C_i(\sigma)$ 都一定大于 C, 所以返工工件一定在各机器空闲进行加工, 即安排在偶数位置上, 并且没有机器空闲, 计算得到:

$$\begin{aligned}&\sum C_i(\sigma)=\\&\sum_{j=1}^{t}(60t-24j+15)B^j+(16t^2-3t)B^t+\\&\sum_{j=1}^{2t}(30t-9j)a_j+\frac{9}{2}\sum_{j=1}^{2t}a_j+\\&\frac{\sum_{i\in S_1}a_i}{2}-\frac{\sum_{i\in S_2}a_i}{2}=C=\\&\sum_{j=1}^{t}(60t-24j+15)B^j+(16t^2-3t)B^t+\\&\sum_{j=1}^{2t}(30t-9j)a_j+9A\end{aligned}\tag{6}$$

即得到 $\sum_{i\in S_1}a_i=\sum_{i\in S_2}a_i=A$, 因此奇偶问题有解. □

定理 2. $1|r_i:w_i^O(\sigma)\le K|\sum w_i^O(\pi^*):\sum w_j(\sigma)$ 重调度问题为 NP 难问题.

证明. 因 $\sum C_i(\sigma)=\sum w_i(\sigma)+\sum p_i=\sum F_i(\sigma)+\sum r_i$, $i=1,\cdots,n$, 其中, $\sum p_i$ 与 $\sum r_i$ 均为已知定值, 故优化目标为最小化 $\sum C_i(\sigma)$、$\sum w_i(\sigma)$、$\sum F_i(\sigma)$ 的三个 RRSM 问题为等价问题. 定理 2 得证. □

目前, 针对单机调度问题 $1|r_j|\sum w_j$ 已有大量研究, 假设根据现有算法[20−22] 已得到最优初始调度. 本文重点研究 RRSM 问题: $1|r_i:w_i^O(\sigma)\le K|\sum w_i^O(\pi^*):\sum w_j(\sigma)$ 及其求解算法.

2 RRSM 问题性质

求解 RRSM 问题: $1|r_i:w_i^O(\sigma)\le K|\sum w_i^O(\pi^*):\sum w_j(\sigma)$, 需要考虑: 1) 确定初始调度中可以插入返工工件的位置, 确定插入返工工件的方法; 2) 将由于插入返工工件而导致开始时间改变的初始工件调整为最优排序. 通过问题分析, 可得到并证明如下问题性质.

性质 1. 最优重调度中, 如果存在返工工件紧接着某个初始工件之后, 则该返工工件的处理时间一定大于该初始工件在重调度后仍可延迟的最大时间.

证明. 反证法: 设在最优重调度 σ^* 中, 存在一个返工工件 j_d^R 紧接着初始工件 j_d^O 被调度, 且 $p_d^R\le\Delta_b^O$. 交换 j_d^R 与 j_d^O 的位置初始工件仍然满足 $w_b^O(\sigma^*)\le K$, 得到另一可行调度 σ', 且 $\sum w_j(\sigma')=\sum w_j(\sigma^*)+p_d^R-p_d^O$, 由第 1.1 节中问题假设条件 (3) 可知 $p^R_{\max}\le p^O_{\min}$, 因此, $\sum w_j(\sigma')\le\sum w_j(\sigma^*)$, 故与 σ^* 为最优调度矛盾, 得证. □

在重调度中, 返工工件的插入很可能需要调整部分初始工件的开始时间, 使相关初始工件相对于初始调度而被延迟开工, 导致部分初始工件的排序失去了最优性. 不失一般性, 以在初始调度中一次插入返工工件为例, 设该次插入一个或多个返工工件, 其处理时间和为 a, 多次插入情况类似, 可以得出下面性质. 其中, ψ 表示当前机器的最早可用时间; $R_i\{\psi\} = \max\{\psi, r_i\}$ 表示初始工件 j_i^O 最早可以开始加工的时间.

性质 2. RRSM 重调度问题 $1|r_i : w_i^O(\sigma) \le K|\sum w_i^O(\pi^*) : \sum w_j(\sigma)$, 插入一次 (一组) 处理时间和为 a 的返工工件. 为保持插入后调度的最优化, 需将满足 $s_{i-1}^O(\pi^*) < r_i < s_{i-1}^O(\pi^*) + a$ 条件的初始工件开工顺序调整为最优.

证明. 因初始调度为最优调度 π^*, 根据 Chu[20] 的性质 1 知 $R_i\{\psi(\pi^*)\} < R_{i+1}\{\psi(\pi^*)\}$, 或者 $\mathrm{PRTF}\{i, \psi(\pi^*)\} \le \mathrm{PRTF}\{i+1, \psi(\pi^*)\}$, 其中, PRTF (Priority rule for total flow time) 定义为: 工件 j_i 在时间 ψ 的优先级度量函数, $\mathrm{PRTF}\{j_i, \psi\} = 2R_i\{\psi\} + p_i$. 因为 $\psi(\sigma) = \psi(\pi^*) + a \ge \psi(\pi^*)$, 所以初始调度中的初始工件可以分成三种情况考虑. 第一种情况为初始工件满足 $r_i < \psi(\pi^*)$, 即 $R_i\{\psi(\pi^*)\} = \psi(\pi^*)$. 因为初始调度是最优调度, 所以工件遵循最短生产时间 (Shortest processing time, SPT) 顺序, 而 $\psi(\sigma) = \psi(\pi^*) + a$ 该部分工件排序不受影响, 所以仍保持最优性. 第二种情况为初始工件满足 $r_i > \psi(\sigma)$, 即 $R_i\{\psi(\sigma)\} = R_i\{\psi(\pi^*)\} = r_i$. 该部分初始工件排序也保持最优性. 只有第三种情况: 满足 $\psi(\pi^*) < r_i < \psi(\sigma)$ 的初始工件排序在重调度中可能失去最优性. 而 $\psi(\sigma) = \psi(\pi^*) + a$, 因此, 只要将满足 $\psi(\pi^*) < r_i < \psi(\pi^*) + a$, 即 $s_{i-1}^O(\pi^*) \le r_i \le s_{i-1}^O(\pi^*) + a$ 的初始工件的排序调整到最优, 即可获得重调度中初始工件的最优调度. □

性质 3. 插入返工工件之后, 为保证重调度最优性而调整的子调度中的工件之间没有机器空闲.

证明. 因插入的返工工件的就绪时间为 0, 故在返工工件前一定没有机器空闲, 机器空闲只能出现在初始工件之前, 根据性质 2 在插入返工工件之后只有满足 $\psi(\pi^*) \le r_i \le \psi(\sigma)$ 条件的初始工件的排序需要调整, 因此, 需要调整的子调度中工件之间没有机器空闲. □

3 DIH 算法

基于上述 RRSM 问题的性质, 设计动态插入调整 (Dynamic insert heuristic, DIH) 算法, 其基本思路如下: 1) 将返工工件按照处理时间的非降序排序; 2) 从初始调度第一个空闲开始, 整合该空闲和空闲之前的初始工件可以延迟的最长时间, 在初始工件前尽可能多地依次插入已排好序的返工工件, 根据最优条件对被延迟的初始工件中符合性质 2 的初始工件进行调整, 确定包含被插入的返工工件及不符合延迟条件的初始工件的子调度; 3) 以此类推, 如果确定的子调度已包含了所有的返工工件, 则剩余没被确定的初始工件保持初始调度中的排序不变, 如果确定的子调度已包含所有的初始工件, 但仍有返工工件没有被调度, 则将剩余的返工工件依次追加到已经确定的子调度之后进行调度. $\alpha|\beta$ 表示子调度 β 紧接着子调度 α 后执行的调度. Δ_i^O 表示初始工件 j_i^O 可以延迟的最长时间. I_i 表示工件 j_i^O 之前的机器空闲时间, $\tilde{J}_B^O$ 表示下一阶段仍能继续延迟的初始工作集. 图 2 是 DIH 算法的流程图.

不失一般性, 为方便陈述, 假设 J^R 中的返工工件顺序符合 SPT 规则, 记为调度 ω, 即: $p_{n_O+1}^R \le p_{n_O+2}^R \le \cdots \le p_{n_O+n_R}^R$.

DIH 算法的具体步骤如下:

设初始值 $\sigma = \emptyset$, $i = 1$, $q = 1$, $t = 1$, $\tilde{I} = I_{[1]}$, $\tilde{J}_B^O = \emptyset$, 按下列步骤执行:

步骤 1. 如果 $\pi^* \ne \emptyset$, 并且 $\omega \ne \emptyset$, 转步骤 1.1; 如果 $\pi^* \ne \emptyset$, $\omega = \emptyset$, 则 $\sigma = \sigma|\pi^*$, 算法结束. 如果 $\pi^* = \emptyset$, $\omega \ne \emptyset$, $\sigma = \sigma|\omega$.

步骤 1.1. 如果 $C_{[i-1]}^O(\pi^*) < s_{[i]}(\pi^*)$ 且 $s_{[i]}^O(\pi^*) = r_{[i]}$, 则计算 $\Delta_{\min}^O(\pi^*) = \min_{q=1,\cdots,i-1}\{\Delta_{[q]}^O(\pi^*)\}$, $\Delta = \min\{\Delta_{\min}^O(\pi^*), r_{[i]} - C_{[i-1]}^O(\pi^*)\}$. 选择初始工件 $j_{[m]}^O$, $m = \arg\max_{i=1,\cdots,n_O}\{\Delta_{[i]}^O(\pi^*) = \Delta_{\min}^O(\pi^*)\}$, $\tilde{J}^O = \{j_{[1]}^O(\pi^*), \cdots, j_{[m]}^O(\pi^*)\}$, 转步骤 1.2; 否则 $i = i + 1$; 如果 $i \le n_O - m$, 则转步骤 1; 如果 $i > n_O - m$, 则令 $\Delta = \Delta_{\min}^O(\pi^*)$, 并转步骤 1.2.

步骤 1.2. 令 $P_l = \sum_{j=1}^{l} p_{[j]}^R, j_j \in \omega$, 如果 $\exists P_l \le \Delta + \tilde{I}$, $P_{l+1} > \Delta + \tilde{I}$, 确定返工工件集 $\tilde{J}^R = \{j_1^R(\omega), \cdots, j_l^R(\omega)\}$, $\omega = \omega - \tilde{J}^R$; 如果 $\tilde{J}^R = \emptyset$, 当 $\Delta = \Delta_{\min}^O(\pi^*)$ 时, $\sigma = \sigma|\tilde{J}^O$, $\pi^* = \pi^* - \tilde{J}^O$, $\tilde{I} = r_{[i]} - C_{[i-1]}^O(\pi^*)$; 当 $\Delta = r_{[i]} - C_{[i-1]}^O(\pi^*)$ 时, $\sigma = \sigma$, 更新 π^* 中工件的所有开始时间, 使 $j_1^O(\pi^*)$ 的开始时间等于 $s_1^O(\pi^*) = s_1^O(\pi^*) + \Delta$, $\tilde{I} = \tilde{I} + \Delta$, 转步骤 1; 否则, $\tilde{J}^R \ne \emptyset$, 则更新 π^* 中工件的所有开始时间, 使 $j_1^O(\pi^*)$ 的开始时间等于 $s_1^O(\pi^*) = \max\{s_1^O(\pi^*) + P_l - \tilde{I}, s_1^O(\pi^*)\}$, 转步骤 2.

步骤 2. 当 $q > m$ 时, 转步骤 2.1; 当 $q \le m$ 时, 如果 $r_{[q]} < s_{[q-1]}^O(\pi^*)$ 或者 $r_{[q]} > s_{[q-1]}^O(\pi^*) + P_l$, 则 $q = q + 1$, 转步骤 2; 否则执行步骤 2.2.

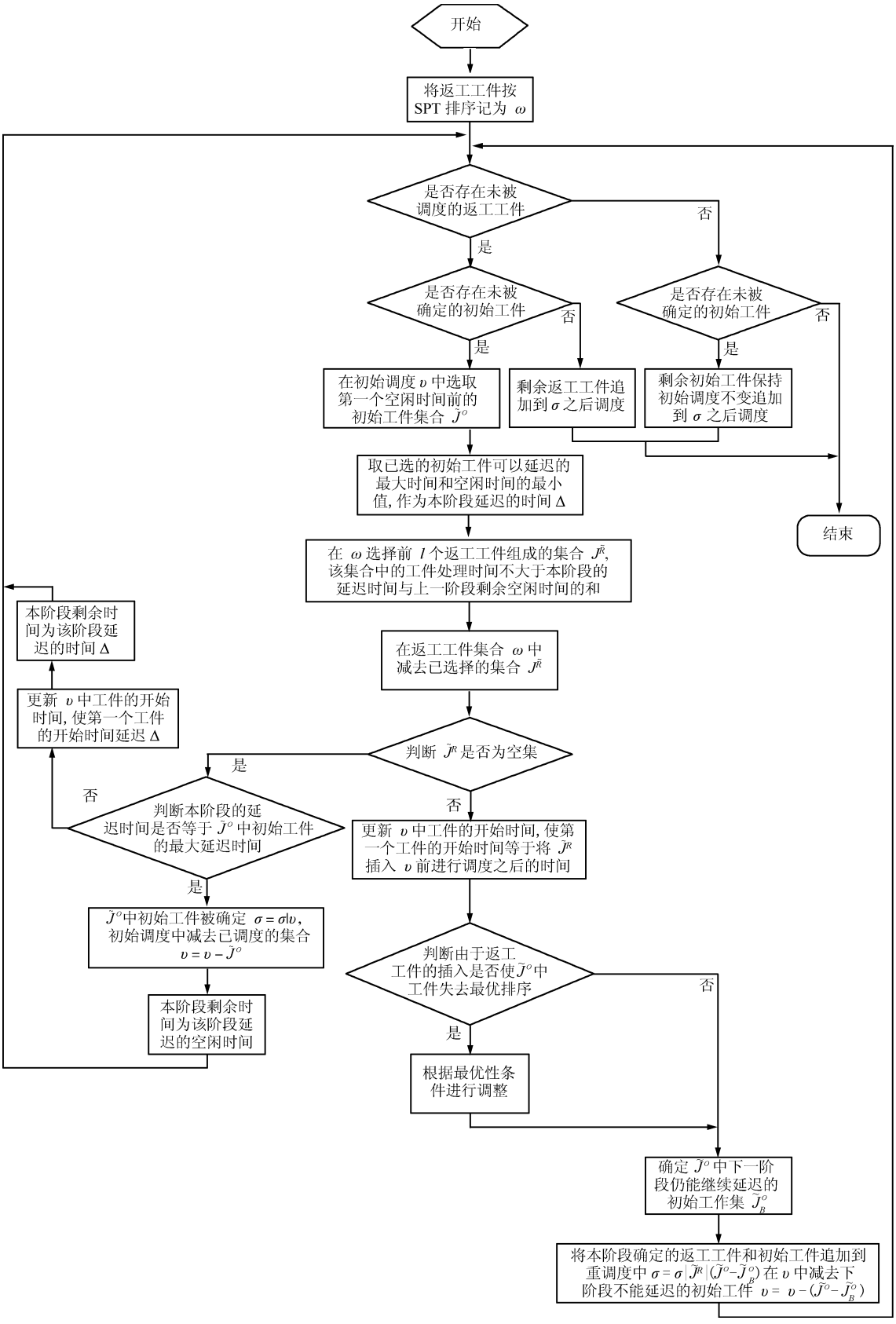


图 2　DIH 算法流程图

Fig. 2　Flow chart of DIH algorithm

步骤 2.1. 如果 $\Delta^O_{[m]}(\pi^*) \geq p^R_{[1]}(\omega)$, $\tilde{J}^O_B = \tilde{J}^O_B \bigcup \{j^O_{[m]}\}$, $m = m-1$, 当 $m > 0$, 转步骤 2.1; 当 $m \leq 0$, $\sigma = \sigma|\tilde{J}^R|(\tilde{J}^O - \tilde{J}^O_B)$, $\pi^* = \pi^* - (\tilde{J}^O - \tilde{J}^O_B)$, 转步骤 1.

步骤 2.2. 如果 $p^O_{[q+t]} < p^O_{[q]}$, $w^O_{[q]}(\pi^*) + p^O_{[q+t]} \leq K$, $r_{[q+t]} \leq s^O_{[q+t-1]}(\pi^*) + P_l$, 则更新 $\tilde{J}^O$, 使 $j^O_{[q]} = j^O_{[q+t]}$, $j^O_{[q+t]} = j^O_{[q]}$, $t = t+1$; 否则, $t = t+1$. 如果 $q + t \leq m$, 转步骤 2.2; 否则 $q = q+1$, 转步骤 2.

DIH 算法的复杂度:

步骤 1 的时间复杂度为 $\mathrm{O}(n_O n_R)$, 步骤 1.1 的时间复杂度为 $\mathrm{O}(2n_O)$, 步骤 1.2 的时间复杂度为 $\mathrm{O}(2n_R+n_O)$, 步骤 2.1 的时间复杂度为 $\mathrm{O}(n^2_O+n_O)$, 步骤 2.2 的时间复杂度为 $\mathrm{O}(2n_O + n_R)$; DIH 算法整体时间复杂度为: $\mathrm{O}(n^3_O n_R + 6n^2_O n_R + 3n_O n^2_R)$.

4 DIH 算法应用及有效性

本节对 DIH 算法应用于 RRSM 重调度问题的有效性, 从特殊问题和一般性问题两方面进行分析.

4.1 两种特殊 RRSM 问题

在此讨论两种特殊的 RRSM 问题: 一种是返工工件具有相同的处理时间; 另一种是初始调度中没有机器空闲时间. 下面通过两个定理说明 DIH 算法求解这两种特殊 RRSM 问题的有效性.

定理 3. DIH 算法能够在多项式时间内求得 $1|r_i : w^O_i(\sigma) \leq K, p^R_i = b|\sum w^O_i(\pi^*) : \sum w_j(\sigma)$ 重调度问题的最优解.

证明. 因返工工件的初始时间 $p^R_i = b, r^R_i = 0$, 故返工工件之间不必考虑排序问题. 又因问题假设条件 $p^R_{\max} \leq p^O_{\min}$, 故若证明该定理命题, 只需证明在满足约束条件下, 将返工工件尽可能多地插入初始调度后, 并保持初始工件排序的最优性即可. 下面用数学归纳法证明.

先验证 DIH 算法进行 1 次循环得到重调度 σ_1 的最优性, 根据 DIH 算法可分 2 种情况讨论.

1) 在满足约束条件下, 没有一个返工工件能插入初始调度进行加工. 因此, 得到的重调度 σ_1 中的初始工件排序保持与初始调度一致, 又因已知初始调度 π^* 为最优调度, 故仍保持初始工件的最优排序, 即得到最优重调度 σ^*_1.

2) 部分返工工件在第一个初始工件前调度. DIH 算法步骤 2 对第一个空闲结束时间之前所包含的 m 个初始工件中, 满足 $C^O_{q-1}(\pi^*) \leq r_q \leq C^O_{q-1}(\pi^*) + \sum p^R_i, q \leq m$ 条件的初始工件排序进行检验、调整. 根据性质 2 知只需证明此时重调度中, 前 m 个初始工件的排序为最优子调度即可. 因为被调整的初始工件满足 $r_q < r_{q+1} < \psi(\sigma_1)$, 所以:

$$\begin{aligned}&\mathrm{PRTF}\{q, \psi(\sigma_1)\} = \\ &\quad 2R_q\{\psi(\sigma_1)\} + p^O_q = 2\psi(\sigma_1) + p^O_q \\ &\mathrm{PRTF}\{q+1, \psi(\sigma_1)\} = \\ &\quad 2R_{q+1}\{\psi(\sigma_1)\} + p^O_{q+1} = 2\psi(\sigma_1) + p^O_{q+1}\end{aligned}$$

又因 $p_{q+1} < p_q$, 所以 $\mathrm{PRTF}\{q+1, \psi(\sigma_1)\} < \mathrm{PRTF}\{q+1, \psi(\sigma_1)\}$. 根据 Chu[20] 的定理 1 可知, 在重调度中初始工件 j^O_{q+1} 一定安排在工件 j^O_q 之前. 再根据性质 3 知重调度中前 m 个初始工件之间一定没有机器空闲, 所以根据 Chu[20] 的定理 13 可知, σ_1 中前 m 个初始工件的顺序为此时重调度中的最优子调度, 所以保证了此时初始工件排序的最优性, 即得到最优重调度 σ^*_1.

假设 DIH 算法进行 h 次循环, 得到最优重调度 σ^*_h. 对 DIH 算法中进行 $h+1$ 次循环, 得到重调度 σ_{h+1} 的最优性证明, 可分 2 种情况.

1) $\sigma_{h+1} = \sigma^*_h|\tilde{J}^R_{h+1}$. 因为第 h 次循环得到的重调度 σ^*_h 为最优调度, 因此, 在重调度 σ_{h+1} 中, 所有初始工件的开始时间与 σ^*_h 相同, 即得到最优重调度 σ^*_{h+1}.

2) $\sigma_{h+1} = \sigma^*_h|\tilde{J}^R_{h+1}|\tilde{J}^O_{h+1}$. 根据 DIH 算法, σ^*_h 中不能再插入 $\tilde{J}^R_{h+1}$ 中任何一个返工工件, 且 $p^R_{\max} \leq p^O_{\min}$, 所以与第 1) 种情况相同, $\sigma_{h+1} = \sigma^*_h|\tilde{J}^R_{h+1}$ 为最优子调度. 只需证明 $\tilde{J}^O_{h+1}$ 中初始工件顺序为最优的即可, 证明过程与证明第 1 次循环最优性的第 2) 种情况相同. 因此, 得到初始工件的最优排序, 即得到最优重调度 σ^*_{h+1}.

综上所述, 对一切 $h\,(h \geq 1)$ 次循环均得到最优重调度 σ^*_h, 所以定理 3 成立. □

定理 4. DIH 算法能够在多项式时间内求得 $1|r_i, I(\pi^*) = 0 : w^O_i(\sigma) \leq K|\sum w^O_i(\pi^*) : \sum w_j(\sigma)$ 重调度问题的最优解.

证明. 该命题分为 2 种情况进行证明.

1) 所有返工工件只能追加到初始调度之后进行调度.

$\sum w_i(\sigma) = \sum w^O_i(\sigma) + \sum w^R_i(\sigma)$, $\sum w^O_i(\sigma) = \sum w^O_i(\pi^*)$, 因为初始调度为最优调度 π^*, 所以等式右侧第一项为最优值; 又根据算法可知, 返工工件按照 SPT 规则进行调度, 且返工工件就绪时间为 0, 所以第二项也为最优值, 故 $\sum w_i(\sigma)$ 为最优解.

2) 所有返工工件都插入到初始调度之中进行调度.

用反证法证明, 假设将 DIH 算法得到的调度 σ 中任意前后衔接的两个返工工件 $j_{[d]}$ 与 $j_{[d']}$ 交换顺序, $d' = d+1$, 得到最优调度 σ'.

a) 如果 $j_{[d]}$ 与 $j_{[d']}$ 均为返工工件, 因初始调度没有机器空闲, 返工工件的就绪时间为 0, 所以, 交换前重调度 σ 的目标值为

$$\sum w_i(\sigma)=\sum w_{[j]}(\sigma)+w_{[d]}+w_{[d']}=\sum w_{[j]}(\sigma)+2C_{[d-1]}+p_{[d]},\quad i\in\{1,\cdots,n\},j\in\{1,\cdots,n\}\setminus\{d,d'\}$$

交换 $j_{[d]}$ 与 $j_{[d']}$ 后得到重调度 σ' 的目标值为

$$\sum w_i(\sigma')=\sum w_{[j]}(\sigma)+w_{[d']}+w_{[d]}=\sum w_{[j]}(\sigma)+2C_{[d-1]}+p_{[d]},\quad i\in\{1,\cdots,n\},j\in\{1,\cdots,n\}\setminus\{d,d'\}$$

由于算法是按照返工工件的 SPT 顺序插入的, 所以 $p_{[d]}\le p_{[d']}$, 所以 $\sum w_i(\sigma)\le\sum w_i(\sigma')$, 与 σ' 是最优调度矛盾.

b) 如果 $j_{[d]}$ 与 $j_{[d']}$ 均为初始工件, 若根据算法可知 $\mathrm{PRTF}(C_{[d-1]},r_{[d]})\le\mathrm{PRTF}(C_{[d-1]},r_{[d']})$, 即 $2\max(C_{[d-1]},r_{[d]})+p_{[d]}\le 2\max(C_{[d-1]},r_{[d']})+p_{[d']}$, 因初始调度没有空闲调度, 所以 $r_{[d]}\le C_{[d-1]}$. 当满足 $r_{[d']}\le C_{[d-1]}$ 时, 有 $p_{[d]}\le p_{[d']}$, 与 a) 均为返工工件情况相同. 当 $r_{[d']}>C_{[d-1]}$ 时, $\sum w_i(\sigma')\ge\sum w_{[j]}(\sigma)+2C_{[d-1]}+p_{[d']}+2(r_{[d']}-C_{[d-1]})\ge\sum w_{[j]}(\sigma)+2C_{[d-1]}+p_{[d']}+2r_{[d']}-2C_{[d-1]}$, 其中: $i\in\{1,\cdots,n\}$, $j\in\{1,\cdots,n\}\setminus\{d,d'\}$, 根据算法可知 $2C_{[d-1]}+p_{[d]}\le 2r_{[d']}+p_{[d']}$, 即 $2r_{[d']}+p_{[d']}-2C_{[d-1]}\ge p_{[d]}$, 导致 $\sum w_i(\sigma')\ge\sum_{[j]}(\sigma)+2C_{[d-1]}+p_{[d]}\ge\sum w_i(\sigma)$ 与 σ' 是最优调度矛盾.

c) 如果 $j_{[d]}$ 为返工工件, $j_{[d']}$ 为初始工件, 有 $\sum w_i(\sigma')\ge\sum w_{[j]}(\sigma)+2C_{[d-1]}+p_{[d']}$ 根据问题假设条件 $p^R_{\max}\le p^O_{\min}$, 有 $p_{[d]}\le p_{[d']}$, 导致 $\sum w_i(\sigma)\le\sum w_i(\sigma')$, 与 σ' 是最优调度矛盾.

d) 如果 $j_{[d]}$ 为初始工件, $j_{[d']}$ 为返工工件, 根据算法可知 $w_{[d]}(\sigma')=w_{[d]}(\sigma)+p_{[d']}>K$, 则 σ' 为不可行解.

综上, DIH 算法求得的重调度 σ 为 $1|r_i,I(\pi^*)=0:w_i^O(\sigma)\le K|\sum w_i^O(\pi^*):\sum w_j(\sigma)$ 问题的最优解. □

4.2 一般的 RRSM 问题

针对一般的 RRSM 问题, DIH 算法同样具有很好的求解效果. 本节先根据 DIH 算法得到解的特征, 给出最优解的判定定理; 然后, 用 RRSM 问题算例 1 说明该判定定理; 最后, 给出一般的 RRSM 问题算例 2, 详细描述 DIH 算法的流程, 并针对实际问题加以说明.

定理 5. 应用 DIH 算法求解 $1|r_i:w_i^O(\sigma)\le K|\sum w_i^O(\pi^*):\sum w_j(\sigma)$ 问题, 如果得到的重调度 σ 中, 最后一个返工工件前没有任何机器空闲, 则得到的重调度为 RRSM 问题的最优重调度 σ^*.

证明. 如果 DIH 算法得到的重调度 σ 中, 最后一个返工工件前没有机器空闲, 则从第一个工件开始连续调度没有机器空闲的所有工件组成的子调度为最优调度. 证明过程与定理 4 的第 2) 种情况相同. 其余工件必均为初始工件, 且其首个初始工件之前有空闲时间说明从该初始工件起之后, 所有的初始工件没有受到插入的返工工件的影响, 子调度仍然与 π^* 中相同, 因此 $\sigma=\sigma^*$. □

RRSM 问题算例 1.

下面以具有如下特征的一类 RRSM 问题为例, 说明 DIH 算法得到的解为最优解:

$$p_1^O=p_2^O=p_4^O=X,p_3^O=X+x$$
$$r_1^O=0,r_2^O=X-\varepsilon_1,r_3^O=2X+x,r_4^O=3X+3x$$
$$\pi^*=\{j_1^O,j_2^O,j_3^O,j_4^O\}$$
$$p_1^R=x,\quad p_2^R=x-\varepsilon_2$$
$$K=x,\quad X>x,\quad \forall\varepsilon_i>0,\quad i=1,2,\quad \varepsilon_1\le\varepsilon_2$$

图 3 中, π^* 表示已知的初始最优调度, σ 表示 DIH 算法获得的重调度. 重调度过程中, 首先, 将返工工件按 SPT 规则排序为 $\{j_2^R,j_1^R\}$; 然后, 计算 j_1^O,j_2^O 可以延后加工的最大时间为 $x-\varepsilon_1$, 因为 $\varepsilon_1\le\varepsilon_2$, 得 $x-\varepsilon_2\le x-\varepsilon_1$, 所以 j_2^R 可以插入到 j_1^O 之前进行调度; 再计算 j_3^O 可以延后加工的最大时间为 x, 加上 j_3^O 之前剩余的空闲时间 ε_2, 得 $x+\varepsilon_2>x$, 所以可以将 j_1^R 插入到 j_3^O 进行调度, 因此, 得到重调度 $\sigma=\{j_2^R,j_1^O,j_2^O,j_1^R,j_3^O,j_4^O\}$, $\sum w_j(\sigma)=2X+4x+\varepsilon_1-4\varepsilon_2$. 同时根据定理 5 的判定条件可以确定 $\sigma=\sigma^*$, $\sum w_j(\sigma)=\sum w_j(\sigma^*)$, 即 DIH 算法得到了 RRSM 问题算例的最优解.

RRSM 问题算例 2.

下面用一个一般的 RRSM 问题的算例来详细阐述 DIH 算法的调度机制.

已知 $K=8$,

$\pi^*=\{j_1^O,j_2^O,j_3^O,j_4^O,j_5^O,j_6^O,j_7^O,j_8^O,j_9^O,j_{10}^O\}$

$\omega=\{j_1^R,j_2^R,j_3^R,j_4^R,j_5^R,j_6^R,j_7^R\}$

表 1 是初始工件和返工工件的具体数据.

表 1　RRSM 问题算例 2

Table 1　Case 2 of RRSM problem

i	1	2	3	4	5	6	7	8	9	10
p_i^O	10	7	7	6	6	6	9	10	9	16
r_i	0	7	20	22	28	33	51	52	78	79
p_i^R	1	2	2	3	4	5	6			

如图 4 (a) 中已知 π^* 和 ω, $\sigma=\emptyset$. 第 1 步

先找到 π^* 中第一个机器空闲之前的初始工件集 $\tilde{J}^O$, 通过计算 $\Delta^O_{\min}(\pi^*)=\Delta^O_{[2]}(\pi^*)=5$, 因此, 在 π^* 前最多可以插入的返工工件的总处理时间不大于 $\Delta=\min\{\Delta^O_{[2]}(\pi^*), r_{[3]}-C^O_{[2]}(\pi^*)\}=r_{[3]}-C^O_{[2]}(\pi^*)=3$.

因此, 在图 4 (b) 中 $\tilde{J}^R=\{j^R_1, j^R_2\}$ 被调度. 因为 $r_{[2]}=r_2=7$, $s^O_{[1]}(\pi^*)=0$, $P_l=3$, $r_{[2]}>s^O_{[1]}(\pi^*)+P_l$, 所以 j^O_1, j^O_2 顺序不变. 但 j^O_1, j^O_2 仍然可以延迟, 所以第 2 步 π^* 中第一个机器空闲之前的初始工件集 $\tilde{J}^O=\{j^O_1,\cdots,j^O_6\}$. 通过计算 $\Delta^O_{\min}(\pi^*)=\Delta^O_{[2]}(\pi^*)=2$, 而在 π^* 前最多可以插入的返工工件的总处理时间不大于 $\Delta=\min\{\Delta^O_{[2]}(\pi^*), r_{[7]}-C^O_{[6]}(\pi^*)\}=\Delta^O_{[2]}(\pi^*)=2$.

因此, 在图 4 (c) 中 $\tilde{J}^R=\{j^R_3\}$ 被调度, j^O_1, j^O_2 被调度. 第 3 步 π^* 中第一个机器空闲之前的初始工件集 $\tilde{J}^O$ 中, $\Delta^O_{\min}(\pi^*)=\Delta^O_{[6]}(\pi^*)=0$, 因此, 在这一步不能调度任何返工工件.

在图 4 (d) 中, $r_{[2]}=r_4=22$, $s^O_{[1]}(\pi^*)=s^O_3(\pi^*)=20$, $P_l=5$, 所以 $s^O_{[1]}(\pi^*)\le r_{[2]}\le s^O_{[1]}(\pi^*)+P_l$, 同时又满足 $p^O_4<p^O_3$, $w_3(\pi^*)+p^O_4\le K$, $r_4\le s^O_3(\pi^*)+P_l$, 因此, 需要交换 j^O_3 和 j^O_4 的位置, 其他初始工件不满足交换条件, 所以 j^O_5, j^O_6 保持原顺序被调度. 接着进行第 4 步, π^* 中第一个机器空闲之前的初始工件集 $\tilde{J}^O$ 中, 虽然 $\Delta^O_{\min}(\pi^*)=\Delta^O_8(\pi^*)=0$, $\Delta=0$, 但是 $\tilde{I}=r_7-C^O_6(\pi^*)=4$, 因此, π^* 前可以插入总处理时间不大于 $\Delta+\tilde{I}=4$ 的返工工件.

因此, 在图 4 (e) 中 $\tilde{J}^R=\{j^R_4\}$ 被调度, j^O_4, j^O_3, j^O_7 和 j^O_8 被调度. 第 5 步与第 4 步情况类似.

因此, 在图 4 (f) 中 $\tilde{J}^R=\{j^R_5\}$ 被调度, j^O_9 和 j^O_{10} 被调度, 至此所有初始工件均被调度, 即 $\pi^*=\emptyset$. 但仍有返工工件 $\omega=\{j^R_6, j^R_7\}$ 没有被调度.

根据 DIH 算法在图 4 (g) 中, 依次将 j^R_6 和 j^R_7 放在 j^O_{10} 之后被调度. 最终获得 RRSM 问题算例 2 的解 $\sigma=\{j^R_1, j^R_2, j^R_3, j^O_1, j^O_2, j^O_4, j^O_3, j^O_5, j^O_6, j^R_4, j^R_7, j^O_8, j^R_5, j^O_9, j^O_{10}, j^R_6, j^R_7\}$.

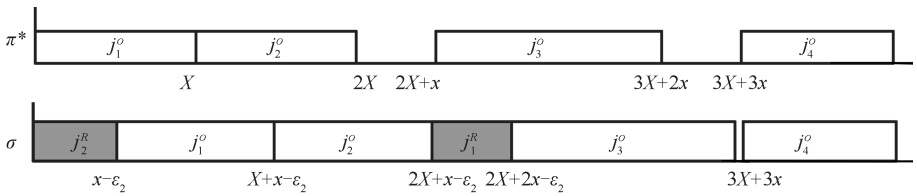


图 3 算例 1 的求解过程图

Fig. 3 Procedures of solving Case 1

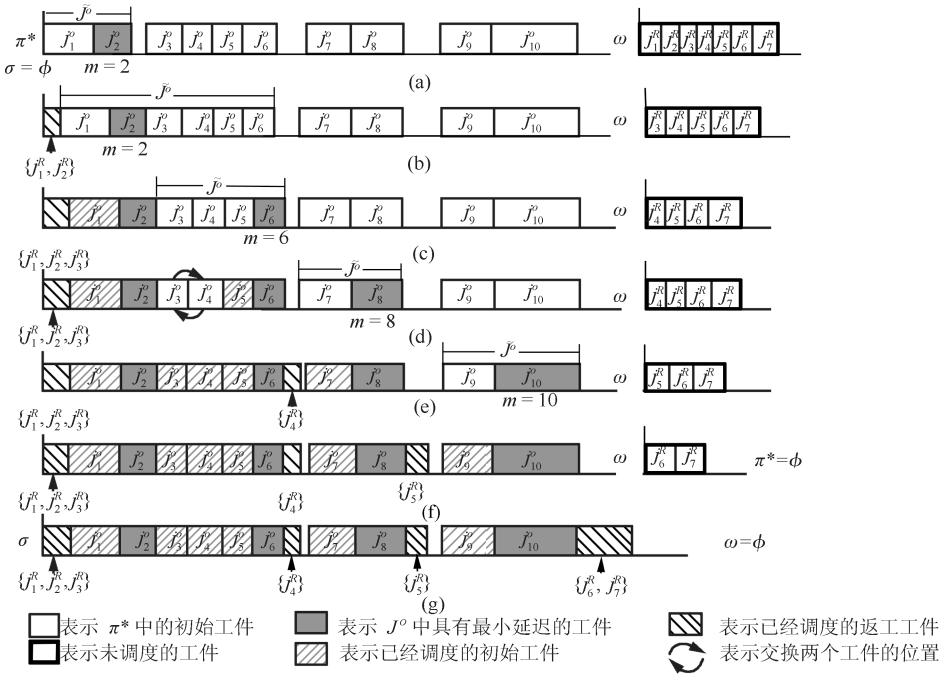


图 4 算例 2 的求解过程图

Fig. 4 Procedures of solving Case 2

值得注意的是, 算例 2 是 RRSM 问题的一般情况, 如果算例 2 中 p_i^R, $i = 1, \cdots, 6$ 的值相等, 则该算例对应的是第 4.1 节中返工工件具有相同处理时间的一类特殊的 RRSM 问题, 根据定理 3 可知, DIH 算法得到的解 σ 为最优解 σ^*; 如果算例 2 中已知的初始调度 π^* 中没有机器空闲时间, 则对应的是第 4.1 节中初始调度中没有机器空闲时间的另一类特殊的 RRSM 问题, 根据定理 4 可知, DIH 算法仍可以得到这类 RRSM 问题的最优解 σ^*.

在实际生产中, 以某石英玻璃厂焊接工位的一类工件为例, 算例 2 中 K 的取值范围约为 0.17 小时至 0.25 小时, 初始工件处理时间的取值范围约为 0.2 小时至 0.75 小时, 返工工件的处理时间范围约为 0.1 小时至 0.2 小时, π^* 为车间已知的调度计划. 当有返工工件需要重调度时, 将模型中的变量附值为实际数据, 即可应用 DIH 算法求解.

5 总结

本文研究了工件排序中复杂的 RRSM 问题, 首先讨论了该问题的复杂度, 并证明了 RRSM 问题为 NP 难问题. 通过对 RRSM 问题的分析, 得到了 3 个性质, 并给出证明. 然后, 在问题性质的基础上设计了 DIH 算法, 描述了算法具体步骤和流程图. 应用 DIH 算法对两个特殊的 RRSM 问题进行求解, 证明了该算法的最优性. 最后, 针对一般的 RRSM 问题, 根据 DIH 算法得到解的特征, 给出了最优解的判定证明, 同时给出相应的算例分别阐述 DIH 算法获得最优解和一般情况下 DIH 算法的详细求解过程. 然而, 本文所研究的返工重调度问题仅针对确定性单机环境, 一方面今后本文研究可以扩展到其他生产环境中, 如并行机、流水车间、加工车间; 另一方面可以延伸到考虑随机性环境, 在实际生产环境中, 有很多计划因素是随机的, 因此, 下一步在考虑随机的情况下开展本研究将会更有实际意义.

References

1 Liu Min. A survey of data-based production scheduling methods. *Acta Automatica Sinica*, 2009, **35**(6): 785−806
(刘民. 基于数据的生产过程调度方法研究综述. 自动化学报, 2009, **35**(6): 785−806)

2 Daniels R L, Kouvelis P. Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science*, 1995, **41**(2): 363−376

3 Gupta S K, Kyparisis J. Single machine scheduling research. *Omega*, 1987, **15**(3): 207−227

4 Webster S, Baker K R. Scheduling groups of jobs on a single machine. *Operations Research*, 1995, **43**(4): 692−703

5 Yan Yang, Wang Da-Zhi, Wang Ding-Wei, Ip W H, Wang Hong-Feng. Single machine group scheduling problems with the effects of deterioration and learning. *Acta Automatica Sinica*, 2009, **35**(10): 1290−1295

6 Zhao Yu-Fang, Tang Li-Xin. Scheduling with agreeable release times and due dates on a single continuous batch processing machine. *Acta Automatica Sinica*, 2008, **34**(8): 957−963
(赵玉芳, 唐立新. 释放时间和工期同序的单机连续型批调度问题. 自动化学报, 2008, **34**(8): 957−963)

7 Herrmann J W. Rescheduling strategies, policies, and methods. *Handbook of Production Scheduling*, 2006, **89**: 135−148

8 Inderfurth K, Kovalyov M Y, Ng C T, Werner F. Cost minimizing scheduling of work and rework processes on a single facility under deterioration of reworkables. *International Journal of Production Economics*, 2007, **105**(2): 345−356

9 Sarker B R, Jamal A M M, Mondal S. Optimal batch sizing in a multi-stage production system with rework consideration. *European Journal of Operational Research*, 2008, **184**(3): 915−929

10 Haji R, Haji B. Optimal batch production for a single machine system with accumulated defectives and random rate of rework. *Journal of Industrial and Systems Engineering*, 2012, **3**(4): 243−256

11 Wu S D, Storer R H, Chang P C. A rescheduling procedure for manufacturing systems under random disruptions. In: New Directions for Operations Research in Manufacturing. Berlin, Germany: Springer, 1992. 292−308

12 Unal A T, Uzsoy R, Kiran A S. Rescheduling on a single machine with part-type depend setup times and deadlines. *Annals of Operations Research*, 1997, **70**: 93−113

13 Hall N G, Potts C N. Rescheduling for new orders. *Operations Research*, 2004, **52**(3): 440−453

14 Hall N G, Liu Z X, Potts C N. Rescheduling for multiple new orders. *INFORMS Journal on Computing*, 2007, **19**(4): 633−645

15 Yuan J J, Mu Y D. Rescheduling with release dates to minimize makespan under a limit on the maximum sequence disruption. *European Journal of Operational Research*, 2007, **182**(2): 936−944

16 Yang B B. Single machine rescheduling with new jobs arrivals and processing time compression. *The International Journal of Advanced Manufacturing Technology*, 2007, **34**(3): 378−384

17 Zhao C L, Tang H Y. Rescheduling problems with deteriorating jobs under disruptions. *Applied Mathematical Modelling*, 2010, **34**(1): 238−243

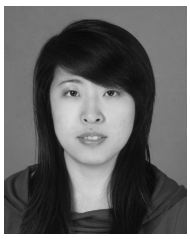
18 Hoogeveen H, Lenté C, T'kindt V. Rescheduling for new orders on a single machine with setup times. *European Journal of Operational Research*, 2012, **223**(1): 40−46

19 Graham R L, Lawler E L, Lenstra J K, Rinnooy Kan A H G. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 1979, **5**: 287−326

20 Chu C B. A branch-and-bound algorithm to minimize total flow time with unequal release dates. *Naval Research Logistics*, 1992, **39**(6): 859−875

21 Kellerer H, Tautenhahn T, Woeginger G J. Approximability and nonapproximability results for minimizing total flow time on a single machine. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing. New York, NY, USA: ACM, 1996. 418−426

22 Chu C. Efficient heuristics to minimize total flow time with release dates. *Operations Research letters*, 1992, **12**(5): 321−330



郭艳东 东北大学信息科学与工程学院系统工程研究所博士研究生, 渤海大学副教授. 主要研究方向为生产计划调度, 优化与决策.
E-mail: gyd1030@163.com
(**GUO Yan-Dong** Ph. D. candidate in the Department of Systems Engineering, College of Information and Engineering, Northeastern University. She is also a associate professor in Bohai University. Her research interest covers production planning and scheduling, optimization and decision.)

王 庆 东北大学信息科学与工程学院系统工程研究所副教授. 主要研究方向为系统建模与优化, 生产运作与计划, 供应链, 企业资源计划, 电子商务, 智能优化算法, 软件工程. 本文通信作者.
E-mail: qwang@mail.neu.edu.cn
(**WANG Qing** Associate professor in the Department of Systems Engineering, College of Information and Engineering, Northeastern University. His research interest covers production planning and scheduling, operations management, supple chain, mathematical modeling and optimization theories, e-commerce, and intelligent algorithm. Corresponding author of this paper.)

黄 敏 东北大学信息科学与工程学院教授, 博士. 主要研究方向为生产计划、调度与存储控制, 物流与供应链管理, 风险管理和软计算.
E-mail: mhuang@mail.neu.edu.cn
(**HUANG Min** Professor in the Department of Systems Engineering, College of Information and Engineering, Northeastern University. Her research interest covers production planning, scheduling and inventory control, the management of logistics and supply chain, and risk management and soft computing.)