文章编号: 1001-0920(2017)11-1921-12 DOI: 10.13195/j.kzyjc.2016.1018

基于剪枝分层的柔性加工车间调度算法

桂忠艳[1,2], 杨 静[1†], 谢志强[3]

(1. 哈尔滨工程大学 计算机科学与技术学院，哈尔滨 150001；2. 黑龙江中医药大学 医学信息工程学院，哈尔滨 150040；3.哈尔滨理工大学 计算机科学与技术学院，哈尔滨 150080)

摘 要: 针对柔性作业车间调度中工序间存在的冗余调度次序约束关系问题和工序-设备间存在的多加工模式情况,提出基于剪枝分层的柔性加工车间调度算法. 该算法首先用有向无环图表示工序及工序间的调度次序关系,采用剪枝法消除图中的冗余弧,采用分层法对图中结点分层;其次对加工模式进行分类,制定工序-设备预约策略和工序-设备预分配策略;最后,采用事件驱动策略,驱动时刻按所提出的柔性加工策略调度工序加工. 理论分析和实例表明,所提出的算法具有较好的调度效果.

关键词: 剪枝分层；工序-设备预约；工序-设备预分配；柔性加工策略

中图分类号: TP278 **文献标志码:** A

Scheduling algorithm for flexible job shop based on pruning and layering

GUI Zhong-yan[1,2], *YANG Jing*[1†], *XIE Zhi-qiang*[3]

(1. College of Computer Science and Technology，Harbin Engineering University，Harbin 150001，China；2. College of Medical Information Engineering，Heilongjiang University of Chinese Medicine，Harbin 150040，China；3. College of Computer Science and Technology，Harbin University of Science and Technology，Harbin 150080，China)

Abstract: In flexible job shop scheduling, two problems are studied,which are the redundancy scheduling sequence constraint among procedures and the multi-mode processing between procedures and devices. Therefore, a scheduling algorithm for flexible job shop based on pruning and layering is proposed. The algorithm first uses the directed acyclic graph(DAG) to represent the procedures and the scheduling sequence constraints among procedures, the pruning strategy to eliminate the redundant in the DAG, and the layering strategy to layer the nodes in the DAG. Then the processing mode of procedures is classified, and the procedure-device booking strategy and procedure-device pre-allocation strategy are designed. Finally, by using the event-driven scheduling strategy, at the driver moment. The procedure is scheduled according to the proposed flexible processing strategy. Theoretical analysis and practical examples proves that the proposed algorithm has a better result.

Keywords: pruning and layering；procedure-device booking；procedure-device pre-allocation；flexible processing strategy

0 引 言

柔性车间调度问题(FJSP)[1-2]是对传统作业车间调度问题(FJSP)[3-4]的扩展,它突破了机器约束和作业加工路线固定的限制,每道作业可以在一台或多台机器上以多种加工模式执行加工,更加符合实际生产情况.

近年来,人们对柔性车间调度问题的研究越来越多,相关专家学者已经取得了一定的成果. Sun等[5]针对现有遗传算法求解FJSP时存在的问题(如模型描述不一致、 编码译码方法复杂等问题),提出了一种基于遗传算法求解FJSP的方法,将带有部分柔性和准时生产要求的JSP转换为一般FJSP,并给出了统一的数学模型. 通过对编码规则、译码算法以及交叉和变异操作的完善,提高了改进遗传算法的收敛性和搜索效率,最后用实验分析表明了所提出算法可以使FJSP稳定、准确和有效地收敛到最优解. Xing等[6]针对FJSP提出了基于知识的蚁群优化算法(KBACO), KBACO对蚁群优化算法模型和知识

收稿日期: 2016-08-10；**修回日期:** 2016-12-12.

基金项目: 国家自然科学基金项目(61672179, 61370083, 61402126, 61370086)；高等学校博士学科点专项科研基金项目(20122304110012)；黑龙江省博士后基金项目(LBH-Z14071).

作者简介: 杜忠艳(1987−),女,博士生,从事企业智能计算、数据挖掘的研究；杨静(1962−),女,教授,博士生导师,从事数据挖掘、企业智能计算等研究.

†通讯作者. E-mail: yangjing@hrbeu.edu.cn

模型提供有效的集成,知识模型从优化的蚁群算法中学习可用的知识并用已有的可用知识指导当前的启发式搜索,最后通过实验数据验证了所提出的KBACO算法的性能超过现有的调度算法. Gao等[7]针对以最小化完成时间、最小化最大机器负载和最小化总负载为目标的FJSP进行研究,提出了一种使用先进的交叉和变异算子来适应特殊染色体结构和问题特征的混合遗传算法. 为了加强搜索能力,通过变邻域下降搜索(VND)改进遗传算法的个体,在最早和最晚事件时间概念的基础上提出了一种为删除的工序发现可分配时间段的高效方式,最后通过大量的基准实验数据给出了算法的性能表现. 以上这些算法均获得了较好的调度效果,但主要解决的是纯加工或纯装配的FJSP.

随着社会对产品个性化需求的不断增加,多品种小批量产品的生产越来越多,企业越来越需要研究加工和装配一同处理的调度算法. Xie等[8]提出了加工和装配一同处理的调度算法,但解决的主要是基于树状结构产品的综合调度问题或综合柔性调度问题. 因此,有必要对其他结构产品的加工装配问题进行研究.

本文研究基于网状结构产品的加工和装配一同处理的调度算法,将工序及工序间的调度约束关系用网状结构的有向无环图表示. 首先,研究有向无环图中工序结点间的复杂调度约束关系,设计简化工艺流程的剪枝分层法;其次,根据工序在设备上的不同加工情况,对工序的加工类型进行分类,并根据不同的加工类型制定不同的工序调度方案;最后,通过实例表明了本文算法的优越性.

1 问题描述

柔性作业车间调度问题是一类资源约束项目调度问题[9-10],一般描述为:k个工件$\{J_1, J_2, \cdots, J_k\}$在$m$台机器$\{M_1, M_2, \cdots, M_m\}$上加工,每个工件$J_i(1 \leqslant i \leqslant k)$由一系列具有工艺加工次序约束关系的工序集$\{P_{i1}, P_{i2}, \cdots, P_{ij}, \cdots, P_{in}\}$组成,$P_{ij}$表示第$i$个工件$J_i$的第$j$道工序,$n$为工件$J_i$包含的工序总数;$M_{ij}(1 \leqslant i \leqslant k, 1 \leqslant j \leqslant n)$为工序$P_{ij}$的可加工机器集,$M_{ij} \in \{1, 2, \cdots, m\}$,工序$P_{ij}$可在$M_{ij}$中的任一台或多台机器上加工,且加工时间随着所选择的机床性能不同而不同,整个车间的总加工工序数$N = \sum\limits_{i=1}^{k}\sum\limits_{j=1}^{n} p_{ij}$.

为简化对柔性车间调度问题的描述,将加工工序和装配工序统一定义为加工工序,加工设备和装配设备统一定义为加工设备. 加工必须满足如下要求:不同工件间是并行加工关系,没有工艺加工顺序约束[11],但存在加工设备约束;同一工件的工序存在工艺约束关系,其加工顺序要严格按照工艺约束关系的顺序进行加工;存在某道工序可在多台设备上加工;一台机器某一时刻只能加工一道工序;工序一旦在设备上加工,加工过程不能中断;某一工序必须在其全部紧前工序加工结束后才可以开始加工;不存在相同机器;允许工序之间等待,允许机器在工序到达前闲置. 调度目标是为每道工序分配最合适的加工机器,并确定每台机器上每个工序的加工顺序,在满足工艺时序约束和资源约束的条件下,求出产品加工调度的近优解. 于是,本文研究问题的数学描述为

$$T = \min\{\max(T_{Mn})\}, 1 \leqslant n \leqslant \sum_{i=1}^{k}\sum_{j=1}^{n} p_{ij}; \quad (1)$$

$$S_{xy} \geqslant \max\{S_{Mi} + t_{Mi}\}; \quad (2)$$

$$T_{M(i+1)} \geqslant T_{Mi} + t_{Mi}. \quad (3)$$

其中:S_{Mi}为第i个工序在设备M上的开始加工时间,t_{Mi}为第i个工序在设备M上的加工时间,T_{Mi}为设备M上第i个工序的驱动时刻. 式(1)表示调度的目标函数是最小化最大完工时间;式(2)表示任意工序的开始加工时间必须大于其所有工艺前序工序的加工结束时间;式(3)表示每台设备上后一道工序的驱动时刻要晚于前一道工序的加工结束时刻.

2 问题分析

柔性作业车间调度中,工序及工序之间的调度次序关系可以用扩展有向无环图$G = (\{V\}, \{E\})$[12]来表达.$\{V\}$称为工序结点集,表示所有作业工序的集合;$\{E\}$称为有向弧集,表示工序结点间活动次序的约束关系的集合. 各集合的定义如下.

1) 工序结点V:用一个5元组$V =$(工件号,工序名,加工类型,加工设备集,加工时间集)表示. 每个工序名都有1个紧前工序个数属性与1个紧后工序集属性与之对应,这两个属性需要根据工艺调度关系计算得到.

2) 有向弧E:有向弧连接两个具有加工次序约束关系的工序结点,用有序对$\langle V_i, V_j\rangle$表示,V_i表示弧的开始结点,V_j表示弧的终止结点,次序约束要求工序结点V_j的开始时间必须晚于工序结点V_i的结束时间.

有向无环图G又称为工艺调度关系图,工艺调度关系图中包含一个标识活动开始的虚拟源点和一个标识活动结束的虚拟汇点. 虚拟源点和虚拟汇点把

整个车间多工件同时加工的并行生产线组合成一个总过程. 工艺调度关系图中不能出现有向环,即任何工序都不应当以自己为先决条件,否则对应的加工过程将无法进行下去. 一个简单的表示一个工件加工过程的工艺调度关系如图1所示.

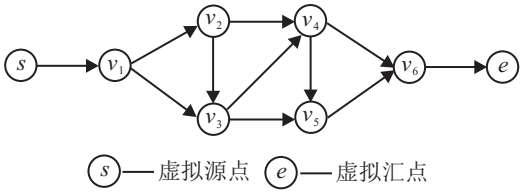


图 1　1个工件的工艺调度关系

2.1　工序间冗余调度约束关系问题分析

定义 1 (冗余弧)　工艺调度关系图中如果某结点与其后继结点间有路径长度大于1的间接路径,则该结点与其后继结点间直接相连的弧为冗余弧.

定义 2 (冗余调度约束关系)　冗余弧连接的两个工序结点间的加工次序约束关系为冗余约束关系.

定义 3 (剪枝)　在工艺调度关系图中,消除某一条冗余弧,即消除了以该弧的开始结点为父结点经过该弧的所有路径组成的分支,所以消除冗余弧的过程为对该分支的剪枝过程.

2.1.1　工序间冗余调度约束关系的存在性

在工艺调度关系图中,任意两个工序结点之间均可能存在调度次序约束关系,工序间可能存在冗余的调度次序约束关系. 例如,某产品的生产过程中存在V_1、V_2、V_3三道工序,由于工艺要求工序V_3需要等待工序V_1和工序V_2全部加工完之后才能开始加工,而工序V_2又要等待工序V_1加工完毕后才能开始加工,工序V_1、V_2、V_3及其相互间的调度约束关系可用图1表示. 由图1可以看出,工序V_3的最早开始时间晚于工序V_2的最晚结束时间,工序V_2最早开始时间晚于工序V_1的最晚结束时间,所以工序V_3最早开始时间肯定晚于工序V_1最晚结束时间. 因此,有向弧$\langle V_1, V_3\rangle$中的调度次序约束关系已被路径(V_1, V_2, V_3)中的调度次序约束关系包含,有向弧$\langle V_1, V_3\rangle$为冗余调度约束关系.

判断存在冗余弧的方法很多,本文采用如果某结点的两个后继结点在一条路径上就存在冗余弧的方法来判断是否存在冗余弧. 判断存在冗余弧的过程即为搜索冗余弧的过程.

2.1.2　剪枝分层法处理工序间冗余调度约束关系的方案设计分析

为清晰地表达工序间的调度次序约束关系设计了剪枝分层法,剪枝分层法是一种先采用剪枝法再采用分层法的方法[13]. 首先,以最小化工序间的调度次序约束关系为目标,采用剪枝法消除工艺调度关系图中的冗余弧,减少不必要的约束关系,经过剪枝处理后得到工艺调度剪枝图;其次,为体现工艺调度过程中相邻工序结点间具有加工先后次序约束关系的工艺调度流程,采用分层法对工艺调度剪枝图中的工序结点进行分层.

1) 剪枝法判断存在冗余弧的设计过程分析.

在工艺调度关系图中,如果某个工序结点V_i与其后继工序结点之间有路径长度大于1的间接路径,则该工序结点V_i至少有两个以上的后继工序结点在该间接路径上. 以V_i为起点,遍历该路径上V_i的所有后继工序结点,记遍历到的后继工序结点顺序依次为$\{V_x, V_{x+1}, \cdots, V_{x+k}\}$,则除$V_x$外,$V_i$和其余后继结点$\{V_{x+1}, \cdots, V_{x+k}\}$之间均存在冗余弧,$V_i$与后继结点$V_z(V_z \in \{V_{x+1}, \cdots, V_{x+k}\})$直接相连的弧为冗余弧.

由于V_x可以是V_i的紧后工序集中的任一工序结点,$\{V_{x+1}, \cdots, V_{x+k}\}$由$V_i$的紧后工序集$\{N_i\}$中不包含$V_x$的其余工序结点组成,即对于$\forall x \in \{N_i\}$,有$\{V_{x+1}, \cdots, V_{x+k}\} \subseteq \{\{N_i\} - V_x\}$. 当工序结点$V_i$和$\{V_x, V_{x+1}, \cdots, V_{x+k}\}$在一条路径上时,由于$V_i$与$V_x$之间肯定有路径,只需要判断$V_x$和$\{V_{x+1}, \cdots, V_{x+k}\}$中的工序结点间是否有路径,即对于$\forall x \in \{N_i\}$,如果$V_x$与$\{\{N_i\} - V_x\}$中的某个结点$V_y$之间有路径,则存在$V_i$到$V_y$的冗余弧,将冗余弧删掉即完成$\langle V_i, V_y\rangle$的剪枝.

通过以上对冗余弧搜索过程的分析,有如下定理.

定理 1　在有向无环图中,如果某结点的多个后继结点之间存在路径,则除了路径的起点之外,该结点与路径上的其余结点之间均存在冗余弧.

证明　设结点V_i是有向无环图G中的某一结点,$\{V_x, V_{x+1}, \cdots, V_{x+k}\}$是$V_i$的后继结点,且$\{V_x, V_{x+1}, \cdots, V_{x+k}\}$之间存在路径Path,设$V_x$为路径的起点,$V_y$为路径的终点. V_i与每一个后继结点间均有一条直接可达的路径,所以V_i与V_x之间有直接路径,V_i通过V_x可间接抵达路径path上V_i的所有后继结点. 因此,V_i和path上除V_x以外的其余后继结点之间有路径长度大于1的间接路径,V_i与其余结点之间均存在冗余弧. □

定理1的充分性成立,但必要性不成立,即如果已知某结点与其多个后继结点之间存在冗余路径,则不能得出这些后继结点在一条路径上.

推论 1　在有向无环图中,如果某结点与其x个后继结点在一条路径上,则该有向无环图至少存在

$x-1$条冗余弧.

由定理1可知,搜索冗余弧的过程就是判定某个结点的多个后继结点之间是否存在路径的过程,由于多个后继结点之间是否有路径的求解问题复杂度较高,本文将求解多个后继结点之间是否有路径的问题转换为求解任意两个后继结点之间是否有路径的问题,即如果某结点的任意两个后继结点之间有路径(路径具有方向性,$V_x \to V_y$和$V_y \to V_x$是两条不同的路径),则存在冗余弧.冗余弧搜索策略主要思想如下:

冗余弧搜索策略从有向无环图的源点开始,按照深度优先顺序遍历有向无环图中的每一个结点,对于当前正在访问的结点,建立该结点的后继结点集,判断后继结点集内任意两个结点之间是否有路径,如果有路径,则存在冗余弧.显然,当紧后工序集中工序个数小于2时不存在冗余弧.

从冗余弧策略可以看出,假设当前结点为V_i,后继结点集中存在$V_x \to V_y$的路径,则$V_i \to V_y$的弧为冗余弧.

经过以上分析,剪枝法的主要思想是采用冗余弧搜索策略搜索冗余弧,对搜索到的冗余弧进行剪枝.

2) 分层法对工序结点分层的必要性分析.

车间中多工序的并行加工过程是一个随时间变化不断前进的过程,工序结点间的关系应该具有很强的方向性,方向性要求后一道工序总是在前一道工序的后方,比如一个普通零件的加工工艺流程是粗加工$\to$精加工$\to$装配$\to$检验$\to$包装,这是一个从开始到结束向着产品完成逐步前进的流程.同时,分层还可以清楚地定位每个结点所在的位置,便于描绘出工序加工过程的进度状况.因此,有必要对工艺调度关系图中的工序结点进行分层,使所有节点边的指向相同,以体现出工序并行加工不断前进的工艺流程.

2.2 工序间多模式加工问题分析

定义4 (协同设备) 某工序的协同设备是指可同时相互协作地完成该道工序加工的多个相关设备.

定义5 (预分配工序集) 指由多个预分配到某空闲设备上且可在该空闲设备上尽早加工结束的工序组成的集合.

定义6 (预调度工序) 预分配工序集中加工结束时间最早的工序为预调度工序.

定义7 (紧后工序集) 某结点的紧后工序集由该结点的全部直接后继工序构成.

定义8 (预约工序) 指预约协同设备在某个特定时刻开始加工的工序.

2.2.1 工序-设备间加工类型的划分

柔性车间调度中的加工设备具有柔性,柔性加工设备能以多种模式对工序执行加工.从完成一道工序加工的角度而言,主要有3种加工模式:一道工序只能由一台设备加工,一道工序可由多台设备中的一台设备加工,一道工序需要在多台设备上同时加工[14].每一种加工模式对应一种加工类型,工序在设备上主要有3种加工类型,分别为:1)加工类型I:一个工序在一台设备上加工;2)加工类型II:一个工序在多个可选设备中的一台设备上加工;3)加工类型III:一个工序在多台设备上同时加工,即多个设备协同地完成一个工序的加工处理.

2.2.2 根据加工类型设计工序的分配策略

实际调度中会出现这3种加工类型的工序,所以需要制定相应的策略使每种加工类型的工序在满足时序约束的前提下尽早调度到相应的设备上加工.在这3种加工类型中,加工类型为III的工序需要在多个设备上同时加工,既有加工设备要求又有加工时间要求,应予以优先处理,因此根据加工类型对工序设置处理顺序优先级,优先处理加工类型III的工序.对加工类型III的工序制定工序-设备预约策略,使加工类型III的工序的加工要求最先得到满足.

对加工类型为I和II的工序采用工序-设备预分配策略,将这两种加工类型的工序分配到尽早加工结束设备的预分配工序集中.考虑到驱动时刻某空闲设备的预分配工序集中可能会被分配到多个加工类型为I和II的可调度工序,这些可调度工序在该驱动时刻的该空闲设备上加工均可以满足各自最早加工结束的要求.但是,由于该设备在该时刻只能加工一道工序,系统必须作出决策确定优先调度的工序,为达到产品尽早加工结束的要求,系统应优先调度预分配工序集中的预调度工序.

2.2.3 柔性加工策略设计

工序-设备预约策略和工序-设备预分配策略都是为获得较好的调度目标而在可调度工序调度前对其做的预处理操作.为真正调度工序到相应的设备上开始加工设计了工序-设备间的柔性加工策略.柔性加工策略的主要思想是在每个设备驱动时刻,系统先判断可调度工序集中是否有预约在当前驱动时刻开始加工的预约工序.如果有,则系统调度在当前驱动时刻开始加工的预约工序到各自的协同设备上加工;如果没有,则在当前时刻开始加工的预约工序或者在当前时刻加工的预约工序已全部调度到协同设

备上加工. 系统再判断此时是否还有设备空闲,如果有设备空闲,则对该空闲设备的预分配工序集中的预调度工序进行调度判断处理;如果没有,则设备空闲,当前时刻的空闲设备全部调度完毕.

2.3　根据加工类型设置工序处理顺序优先级

2.3.1　按加工类型设置工序处理顺序优先级的原因

由于加工类型为III的工序要在协同设备上同时加工,如果不作特殊的控制处理(如让某协同设备等待另一协同设备加工处理完),则某工序的协同设备很难在同一驱动时刻均处于空闲状态,可能经常会出现一部分协同设备处于空闲状态而另一部分协同设备处于加工状态. 当处于加工状态的协同设备加工结束后变为空闲状态时,原本处于空闲状态的协同设备可能早已开始了新的工序的加工而变为加工状态,这样,反复下去,会出现加工类型为III的工序迟迟得不到加工的情况,因此,需要根据工序的加工类型对工序的处理顺序设置优先级.

2.3.2　处理顺序优先级的具体规则

加工类型的可调度工序具有最高的处理顺序优先级,可调度工序集一旦出现加工类型III的工序应优先予以处理,加工类型为I和II的可调度工序优先级相同. 当存在多个加工类型为III的可调度工序时,其协同设备个数越多加工优先级越高. 当协同设备个数相同时,按尽早加工结束要求加工时间短的可调度工序优先级高. 如果协同设备个数相同且加工用时相同,则优先级相同,优先处理哪个工序都可以.

3　调度策略设计与实现

3.1　剪枝分层法

剪枝分层法由剪枝法和分层法两部分组成,首先用剪枝法消除冗余的工艺调度约束关系得到工艺调度剪枝图,再用分层法对工艺调度剪枝图中的工序结点进行分层得到工艺调度剪枝分层图.有关剪枝法和分层法的具体实现过程如下.

3.1.1　剪 枝 法

1) 剪枝法的设计思想.

对于有向无环图中的每个工序结点V_i,建立结点V_iSub_OpSet$[V_i][]$,设置紧后工序集中每个工序的状态属性为0(0表示每个紧后工序节点都未被剪枝处理),计算紧后工序集中的工序个数k. 如果$k = 0$或$k = 1$,则表示结点V_i没有紧后工序或只有1个紧后工序,此时结点V_i与紧后工序之间最多只有1条弧,不存在冗余弧,不做剪枝处理;如果紧后工序集中的工序个数$k > 1$,则依次对紧后工序集中的每个工序结点$V_x(1 \geqslant x \geqslant k)$,建立$V_x$的补集$\overline{\text{Sub_OpSet}[V_i][V_x]}$. Sub_OpSet$[V_i][\,]$中有$k$个工序,$\overline{\text{Sub_OpSet}[V_i][V_x]}$中有$k - 1$个工序. 对于$\overline{\text{Sub_OpSet}[V_i][V_x]}$中的每个结点$V_y(1 \leqslant y \leqslant k-1)$,当$y < x$时,$V_y$与Sub_OpSet$[V_i][V_y]$相对应;当$y > x$时,$V_y$与Sub_OpSet$[V_i][V_{y+1}]$相对应,其对应关系如图2所示, Sub_OpSet$[V_i][\,]$和$\overline{\text{Sub_OpSet}[V_i][V_x]}$中相对应的工序的工序状态属性一致. 对于$\overline{\text{Sub_OpSet}[V_i][V_x]}$中每个未被访问的结点$V_y$,判断结点$V_x$与结点$V_y$之间是否有路径,如果没有路径,则不作任何处理;如果有路径,则保留V_i与结点V_x之间的弧,将V_i与结点V_y之间的弧删除掉,并将紧后工序集Sub_OpSet$[V_i][\,]$中与结点V_y对应的工序的状态属性标记为1(表示该节点已剪枝处理完).

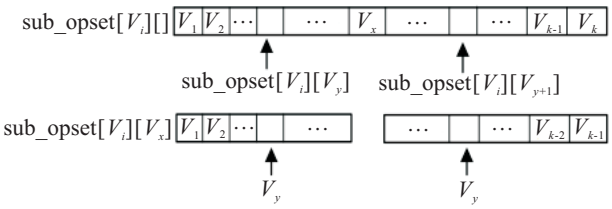


图 2　Sub_OpSet$[V_i][\,]$和V_y的对应关系图

2) 剪枝法的具体过程如算法1所示.

算法1　剪枝法.

输入:具有冗余边的有向无环图;

输出:无冗余边的有向无环图.

count( )函数用来计算结点个数;

Path(V_x, V_y)用来判断结点V_x与结点V_y之间是否有路径存在,如果Path$(V_x, V_y) = 1$,则表示V_x与V_y之间有路径,如果Path$(V_x, V_y) = 0$,则表示V_x与V_y之间没有路径,V_x为路径起点,V_y为路径终点.

Begin

For(有向无环图中的每个结点V_i, $1 \leqslant i \leqslant n$)

{

建立结点V_i的紧后工序集Sub_OpSet$[V_i][\,]$;

k = count(Sub_OpSet$[V_i][\,]$);

For(Sub_OpSet$[V_i][\,]$的每个结点V_x,$1 \leqslant x \leqslant k$)

Sub_OpSet$[V_i][V_x]$.status=0;

If($k \geqslant 0$)

{

If($k == 0 || k == 1$) Then end;

If($k > 1$)

For(each $V_x(1 \leqslant x \leqslant k)$ in Sub_OpSet$[V_i][\,]$)

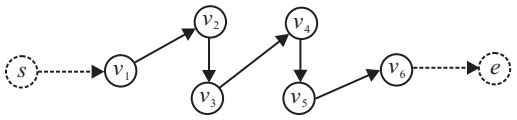
{$\overline{\text{Sub\_OpSet}[V_i][V_x]} = \{\text{Sub\_OpSet}[V_i][\,] - V_x\}$;}

For ($\overline{\text{Sub_OpSet}[V_i][V_x]}$中的每个节点$V_y$, $1 \leqslant y \leqslant k-1$)

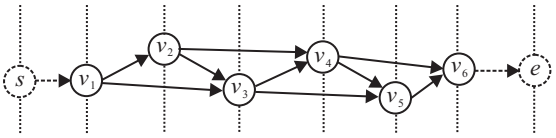
If (V_y.status== 0)

{ If(Path(V_x, V_y) == 0)
Then 不作任何处理;
If(Path(V_x, V_y) == 1) Then
delete$E(V_i, V_y)$;
If($y < x$)
Sub_OpSet$[V_i][y]$.status = 1;
Else
Sub_OpSet$[V_i][y+1]$.status= 1;
}
}
}
End

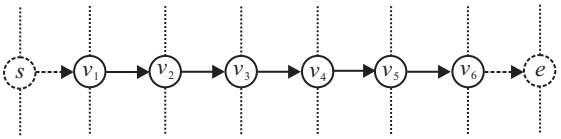
经过剪枝处理的工艺调度剪枝图中,如果某工件的两个工序结点之间有直接路径,则不会有间接路径;如果某工件的两个工序结点之间有间接路径,则不会有直接路径. 对图1中的工序结点采用剪枝法得到的工艺调度剪枝图如图3(a)所示.



(a) 工艺调度剪枝图



(b) 工艺调度分层图



(c) 工艺调度剪枝分层图

图3 工艺调度关系

3.1.2 分 层 法

分层法主要是计算工艺调度关系图中每个工序结点所在的层次并对其分层. 分层法主要步骤如下:

1) 将虚拟源点放在第$L(0)$层;

2) 对于工艺调度关系图中的每个工序结点V_i(虚拟源点和虚拟汇点除外),设其所在的层次为$L(V_i)$,则$L(V_i) = \max\{L(K_1), L(K_2), \cdots, L(K_n)\} + 1$,式中$K_1, K_2, \cdots, K_n$为$V_i$的直接前驱结点;

3) 将虚拟汇点放在最右侧,即$\max\{L(V_i)\}+1$层$(1 \leqslant i \leqslant n)$.

对图1中的工序结点采用分层法得到的工艺调度分层如图3(b)所示.

由图3(a)可看出,经过剪枝处理,工序结点间加工次序约束关系达到最小化. 由图3(b)可以看出,经过分层处理,有加工先后次序关系的工序结点位于不同的层,工艺流程层次清晰. 剪枝分层法兼顾了两者的优点,对图1中的工序结点采用剪枝分层法得到工艺调度剪枝分层图,如图3(c)所示. 由图3(c)可看出,经过剪枝分层处理产品的生产工艺流程一目了然.

虽然剪枝分层法并未能从本质上缩短产品制造时间,但经过剪枝分层处理工艺调度中的冗余约束关系被消除,冗余约束关系连接的两个工序结点不再是直接相连的工序结点,降低了计算结点的紧前工序个数和建立紧后工序集时寻找紧后工序的次数;该方法还可以使工艺流程清晰地表达,便于管理者理顺工艺流程,编制生产作业计划,定位每个结点所在的位置,掌握工序加工过程的进度状况等.

3.2 工序-设备预分配策略

对于加工类型的可调度工序P_{ij},将可调度工序P_{ij}分配到其唯一加工设备的预分配工序集中. 在该加工设备的某个驱动时刻T_k,工序P_{ij}为预分配工序集中的预调度工序时,调度工序P_{ij}在该设备上加工.

对于加工类型的可调度工序P_{ij},将可调度工序P_{ij}分配到可尽早加工结束的柔性加工设备的预分配工序集中. 具体过程如下:

1) 假设可调度工序P_{ij}的柔性加工设备为$\{M_1, M_2, \cdots, M_m\}$,工序$P_{ij}$在柔性加工设备上的加工时间分别为$\{t_1, t_2, \cdots, t_m\}$,柔性加工设备的最早驱动时刻分别为$\{T_1, T_2, \cdots, T_m\}$,计算工序$P_{ij}$在柔性设备上的加工结束时间$\{T_1+t_1, T_2+t_2, \cdots, T_m+t_m\}$,采用快速排序法对加工结束时间排序.

2) 对工序P_{ij}做试探分配操作:将工序P_{ij}预分配到最早加工结束设备M_i的预分配工序集中. 在设备M_i的最早驱动时刻到来时,如果工序P_{ij}为预分配工序集中的预调度工序,则调度P_{ij}在设备M_i上加工;如果P_{ij}不是设备M_i的预调度工序,则将P_{ij}预分配到加工结束时间次早的柔性加工设备M_j的预分配工序集中,再判断P_{ij}在设备M_j的驱动时刻是否为设备M_j上的预调度工序. 依此类推,这样可将工序P_{ij}分配到加工结束时间尽可能早的柔性设备上加工.

3.3 工序-设备预约策略

Step 1: 计算加工类型III的可调度工序个数g,如果$g = 1$,则对这唯一的工序按Step 2执行T_k时刻的

工序-设备预约操作,执行完 Step 2 后,转 Step 6; 如果 $g > 1$,则转 Step 3.

Step 2: 执行 T_k 时刻的工序-设备预约操作,具体实现步骤如下:

Step 2.1: 判断 T_k 时刻该工序的协同设备是否全部处于空闲状态,若是,则转 Step 2.2,否则,转 Step 2.3;

Step 2.2: 该工序预约所有协同设备在 T_k 开始加工,转 Step 2.4;

Step 2.3: 计算并比较此时处于加工状态的协同设备的加工结束时刻,令该工序预约所有协同设备在最晚加工结束时刻开始加工;

Step 2.4: 预约加工时长为该工序的加工时间,预约结束时间=预约开始时间+该工序的加工时间,标记已被预约的设备在预约的时间段内为预约加工状态.

Step 3: 对 g 个加工类型为 III 的可调度工序按优先级规则确定处理顺序.

Step 4: 对第 1 个具有最高处理顺序的工序按 Step 2 中 T_k 时刻的工序-设备预约操作过程执行该工序对设备的特定时刻预约.

Step 5: $r = g - 1$,系统处理其余 r 个优先级依次非递增的工序,具体步骤如下:

Step 5.1: 对优先级依次非递增的每一个工序 P_{ij},判断 P_{ij} 的某些协同设备是否已被优先级不低于 P_{ij} 的某些工序在特定时刻预约,如果没有在特定时刻预约,则转 Step 5.2,否则,转 Step 5.3;

Step 5.2: 对工序 P_{ij} 按 Step 2 执行 T_k 时刻的工序-设备预约操作,转 Step 5.4;

Step 5.3: 设被预约的协同设备中最晚的预约结束时间为 T_y,则对工序 P_{ij} 按 Step 2 执行 T_y 时刻的工序-设备预约操作;

Step 5.4: $r = r - 1$,如果 $r = 0$,则转 Step 6,否则,转 Step 5.1.

Step 6: 结束.

3.4　柔性加工策略

Step 1: 系统首先检查该驱动时刻 T_k 的可调度工序集中是否有加工类型 III 的预约工序在当前时刻开始加工,如果有,则转 Step 2,如果没有,则转 Step 6.

Step 2: 系统计算在当前时刻 T_k 开始加工的预约工序个数 Δn.

Step 3: 系统按顺序调度每一个预约在当前时刻开始加工的预约工序到相应的协同设备上加工.

Step 4: $\Delta n = \Delta n - 1$,如果 $\Delta n = 0$,则转 Step 5,否则,转 Step 3.

Step 5: 系统判断该驱动时刻 T_k 是否还有空闲设备,如果有,则转 Step 6,如果没有,则转 Step 12.

Step 6: 系统计算当前时刻的空闲设备个数 Δm.

Step 7: 对于每一个空闲设备 $M_i (1 \leqslant i \leqslant \Delta m)$,系统检查 M_i 在 T_k 时刻以后的某个时间段内是否被预约,如果没有被预约,则转 Step 8,否则,转 Step 9.

Step 8: 系统调度该空闲设备 M_i,寻找 M_i 的预分配工序集中的预调度工序,如果找到,则转 Step 10,如果没找到,则转 Step 11.

Step 9: 系统判断从 T_k 时刻起到最早开始预约时刻的时间段内,空闲设备 M_i 的预分配工序集中的预调度工序是否可在这段时间内加工完毕,是,则转 Step 10,否则,转 Step 11.

Step 10: 调度该工序加工.

Step 11: $\Delta m = \Delta m - 1$,如果 $\Delta m = 0$,则转 Step 12,否则,转 Step 7.

Step 12: 结束.

3.5　事件驱动策略

事件驱动策略[15]是指调度系统在驱动时刻根据事件队列中出现的各种事件对设备集和工序集进行处理. 调度系统由设备集、工序集和事件队列构成,事件队列由加工开始事件、加工完毕事件和设备空闲事件构成.

4　算法设计

算法设计思想如下:

Step 1: 设置工序属性. J_i 表示工件名;P_{ij} 表示工件 J_i 的第 j 个工序的工序号; Type 表示工序 P_{ij} 的加工类型; $\{M_1, M_2, \cdots, M_m\}$ 表示工序 P_{ij} 的加工设备集,每个加工设备均有一个预分配工序集; $\{t_1, t_2, \cdots, t_m\}$ 表示工序 P_{ij} 的加工时间集; C_{ij} 表示 P_{ij} 的紧前工序个数,当 $C_{ij} = 0$ 时,表示该工序为可调度工序; $\{N_1, N_2, \cdots, N_x\}$ 表示 P_{ij} 的紧后工序集. 于是,工序的属性表示为 J_i/P_{ij}/Type/$\{M_1, M_2, \cdots, M_m\}/\{t_1, t_2, \cdots, t_m\}/C_{ij}/\{N_1, N_2, \cdots, N_x\}$.

Step 2: 用剪枝分层法对工艺调度关系图进行处理,得到工艺调度剪枝分层图.

Step 3: 计算工艺调度剪枝分层图中每个工序结点(虚拟源点和虚拟汇点除外)的紧前工序个数.

For ($i = 1; i < n; i++$){ $C_{ij} = 0$; for($x = 1; x < n; x++$){if ($N_x = P_{ij}$) $C_{ij}++$; } }

Step 4: 根据工艺调度剪枝分层图为每个工序结点(虚拟源点和虚拟汇点除外)建立紧后工序集.

Step 5: 建立初始时刻由可调度工序集、空闲设备

集和事件队列构成的调度系统.

Step 5.1: 建立初始时刻的可调度工序集.

初始时刻,系统根据工序P_{ij}的紧前工序个数属性C_{ij},将$C_{ij}=0$的工序加入可调度工序集中.

Step 5.2: 建立初始时刻空闲设备集.

初始时刻,所有设备均处于空闲状态,系统将所有设备加入到空闲设备集.

Step 5.3: 建立初始时刻的事件队列.

初始时刻,系统将设备空闲事件加入事件队.

Step 5.4: 驱动时刻的计算.

初始时刻为$T_0=0$.非初始时刻,计算当前驱动时刻T_k所有正在加工工序的加工结束时间,将最早加工结束时间$\min\{T_k+t_i\}(1\leqslant i\leqslant m)$确定为下一设备驱动时刻$T_{k+1}$.

Step 6: 系统处理初始时刻的可调度工序集.

系统检查可调度工序集中工序的加工类型.如果有加工类型III的可调度工序,则采用工序-设备预约策略; 如果有加工类型I或II的工序,则采用工序-设备预分配策略.

Step 7: 系统检查事件队列,并处理可调度工序集和空闲设备集.

1) 若为加工完毕事件,则将加工完毕的工序从可调度工序集中删除;若此时可调度工序集中的工序全部调度完毕,则转Step 9, 否则,对于加工完毕的工序,将其紧后工序集中的工序的C_{ij}值减1.如果某些紧后工序的紧前工序个数$C_{ij}=0$,则将这些$C_{ij}=0$的紧后工序加入可调度工序集.系统检查所有新加入的可调度工序的加工类型,首先,判断是否存在加工类型的可调度工序.如果存在,则采用工序-设备预约策略,对该驱动时刻加工类型的可调度工序按优先级由高到低的顺序逐个预约各自的协同设备,在特定的时刻开始加工;如果存在加工类型I或II的工序,则按照工序-设备预分配策略将该可调度工序预分配到尽早加工结束设备的预分配工序集中. 同时,完成加工任务的设备处于空闲状态,产生设备空闲事件,将该事件加入事件队列.

2) 若为设备空闲事件,系统则按照柔性加工策略确定该驱动时刻各空闲设备上的加工工序,工序集触发加工开始事件,将该事件加入事件队列,设备空闲事件移出事件队列.

3) 若为加工开始事件,则将开始加工的设备从空闲设备集中删除.

Step 8: 系统根据驱动时刻的计算方法,计算下一设备驱动时刻T_{k+1},在T_{k+1}时刻,工序加工完毕,触发加工完毕事件,将该事件加入事件队列,转Step7.

Step9: 该驱动时刻即为产品总加工结束时间,结束.

5 算法复杂度分析

设产品的总工件数为k,总工序为n,总加工设备数为m,本文算法主要有如下操作:

1) 剪枝法: 由剪枝算法的详细算法描述可知,最坏情况下,剪枝算法的时间复杂度为$O(n^3)$.

2) 分层法: 由于每个工件至少有1道工序,存在某个工件J_i最多有$n-k+1$个工序,该工件J_i最多有$n-k+1$层,工件J_i中存在某个结点最多有$n-k$个紧前工序,最多需要比较$n-k-1$次操作确定该结点所在的层.分层法的最多操作次数取决于最大层数$\times$每层的最多比较次数,所以分层法最多需要$(n-k+1)(n-k-1)$次操作,时间复杂度为$O(n^2)$.

3) 计算紧前工序个数: 由算法设计中的Step3可知,计算所有结点的紧前工序个数的时间复杂度为$O(n^2)$.

4) 计算驱动时刻: 当前驱动时刻T_k,最多有m个设备均有工序加工,需比较$m-1$次确定T_{k+1}时刻.由于整个产品调度最多有n个驱动时刻,计算驱动时刻最多需要比较$n(m-1)$次,因$m\ll n$,所以时间复杂度为$O(n)$.

5) 工序-设备预分配策略:

i) 对于加工类型I的可调度工序,将其分配到加工设备的预分配工序集中需1次操作,最多有n个加工类型为I的可调度工序,所以最多需要n次分配操作,时间复杂度为$O(n)$.

ii) 对于加工类型II的可调度工序P_{ij},最多有m个柔性加工设备,计算柔性加工设备的加工结束时间最多需要m次操作;采用快速排序法对加工结束时间排序的时间复杂度为$O(m\lg m)$;对工序P_{ij}做预分配试探操作,最坏情况下,工序P_{ij}要按加工结束时间从小到大的顺序依次预分配到m个柔性加工设备上各1次,共分配m次,每一次预分配都需要判断P_{ij}是否为当前分配设备的预分配工序集中的预调度工序,因预分配工序集中最多有$n-1$个可调度工序,最多需要$n-2$次比较来判断P_{ij}是否为预调度工序,所以对工序P_{ij}的预分配试探操作需要$m(n-2)$次操作. 综上,对每个加工类型II的工序预分配一次的时间复杂度数量级为$O(m+m\lg m+m+m(n-2))$,最多有n个加工类型II的可调度工序,且$m\ll n$,所以时间复杂度为$O(n^2)$.

工序-设备预分配策略的时间复杂度取决于i)和

ii)的最大值,所以时间复杂度为$O(n^2)$.

6) 工序-设备预约策略:对于每一个加工类型III的工序,如果该工序的协同设备在驱动时刻都没有被预约,则该工序直接预约协同设备,最多需要1次操作,这是最好的情况. 最坏情况下,该工序的所有协同设备均已被预约,该工序最多有m个协同设备,最多需要比较$m-1$次以找出最晚加工结束的设备,执行预约需1次操作,计算预约结束时间需1次操作,标记预约加工状态需1次操作,总共需要$m+2$次操作. 因为最多有n个加工类型III的工序,且$m \ll n$,所以时间复杂度为$O(n)$.

7) 柔性加工策略:驱动时刻T_k,该算法的复杂度数量级为O(预约工序个数Δn+剩余空闲设备个数Δm),由于预约工序个数$\Delta n < n$,剩余空闲设备个数$\Delta m < m$,且$m \ll n$,T_k时刻的时间复杂度为$O(n)$. 由于最多有n个驱动时刻,时间复杂度为$O(n^2)$.

6　调度实例对比与分析

设某车间有一个产品A由3个工件组成,这3个工件需要在4台柔性加工设备上同时加工处理,每个工件具有不同的工艺加工路线,具体的工艺调度关系如图4所示. 每个工件包含的工序数不同,总共有24个工序,每个工序结点V_i对应一个五元组:工序号/工序名/加工类型/加工设备集/加工时间集,具体对应值参见表1.

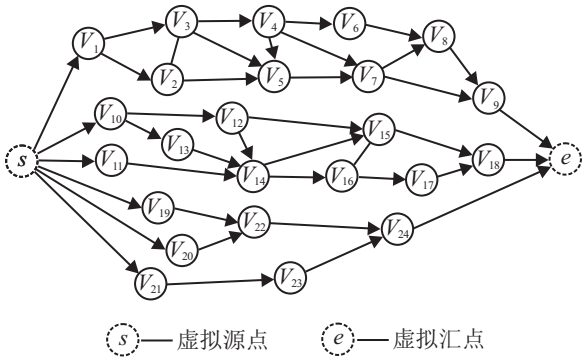


图4　产品A的工艺调度关系

采用本文提出的调度算法对车间中的3个工件进行调度处理. 采用剪枝分层法对工艺调度关系图进行处理,得到工艺调度剪枝分层如图5所示.

建立初始时刻由空闲设备集、可调度工序集和事件队列构成的调度系统,按本文提出的驱动时刻的调度算法对图5中的工序结点进行调度,得到工序调度次序为: P_{21}, P_{32}, P_{11}, P_{31}, P_{23}, P_{33}, P_{34}, P_{12}, P_{22}, P_{13}, P_{14}, P_{24}, P_{16}, P_{15}, P_{25}, P_{35}, P_{17}, P_{26}, P_{18}, P_{27}, P_{28}, P_{19}, P_{36}, P_{29}. 调度甘特图如图6(a)所示.

表1　不同风险厌恶水平下的最优投资和对冲组合的权重

工序结点	工件号	工序号	加工类型	加工设备	加工设备/工时
V_1	J_1	P_{11}	III	M_1, M_4	20
V_2	J_1	P_{12}	II	M_1, M_4	10, 5
V_3	J_1	P_{13}	I	M_1	5
V_4	J_1	P_{14}	III	M_1, M_3	5
V_5	J_1	P_{15}	II	M_1, M_3	10, 5
V_6	J_1	P_{16}	III	M_1, M_4	10
V_7	J_1	P_{17}	I	M_3	5
V_8	J_1	P_{18}	II	M_3, M_4	5, 10
V_9	J_1	P_{19}	III	M_2, M_4	20
V_{10}	J_2	P_{21}	III	M_2, M_3, M_4	15
V_{11}	J_2	P_{22}	II	M_3, M_4	20, 10
V_{12}	J_2	P_{23}	II	M_1, M_3	5, 15
V_{13}	J_2	P_{24}	II	M_1, M_2	20, 20
V_{14}	J_2	P_{25}	II	M_1, M_4	10, 10
V_{15}	J_2	P_{26}	II	M_1, M_4	25, 25
V_{16}	J_2	P_{27}	III	M_1, M_3	5
V_{17}	J_2	P_{28}	III	M_3, M_4	5
V_{18}	J_2	P_{29}	III	M_1, M_3	10
V_{19}	J_3	P_{31}	I	M_2	10
V_{20}	J_3	P_{32}	II	M_1, M_3	10, 5
V_{21}	J_3	P_{33}	III	M_2, M_4	25
V_{22}	J_3	P_{34}	III	M_1, M_3	10
V_{23}	J_3	P_{35}	II	M_2, M_3	25, 15
V_{24}	J_3	P_{36}	II	M_2, M_3, M_4	15, 5, 20

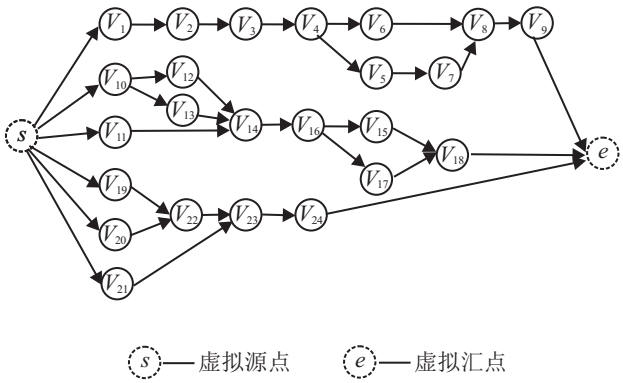
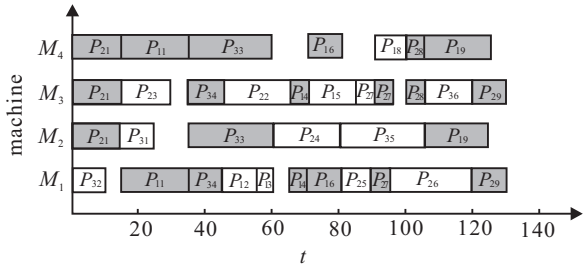
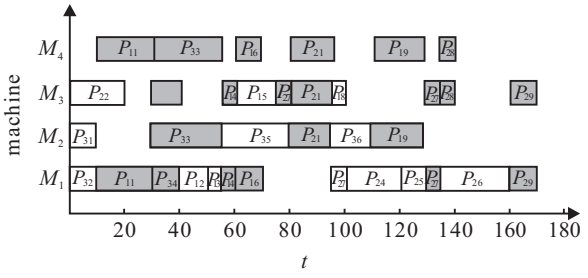


图5　采用剪枝分层法化简产品A得到的流程

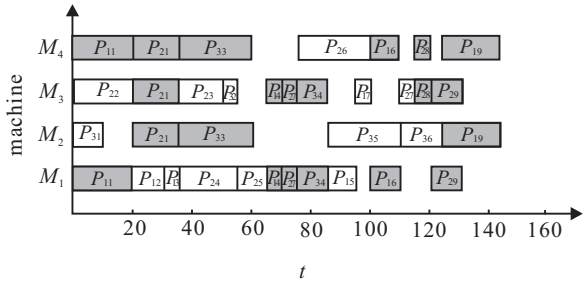
本文研究的柔性车间调度中存在多设备工序的情况,目前在综合调度中有相关的算法研究[14],但只解决了树状结构产品的多设备工序调度问题,在非综合调度中还未见相关研究. 本文的调度工序是在采用事件驱动策略时按提出的工序-设备柔性加工调度算法选择工序调度加工,所以本文对比了采用事件驱动策略时不按照本文提出的算法调度工序加工的调度算法. 由于事件驱动策略对所调度的产品结构没有限制要求,将综合柔性调度中与事件驱动策略相关的调度算法应用到传统的柔性车间调度中,综合柔性调度中基于事件驱动的调度算法在驱动时刻选择



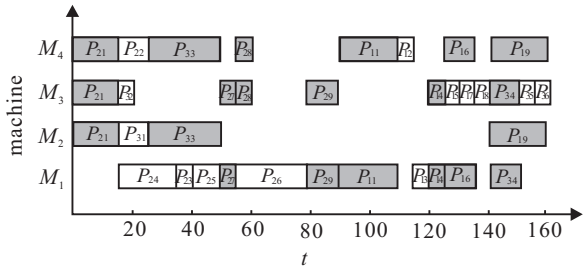
(a) 本文算法调度甘特图



(b) 采用方案1调度的甘特图



(c) 采用方案2调度的甘特图



(d) 采用修改的文献[4]调度的甘特图

图6 调度甘特图

可调度工序时主要有两种方案: 一种是选择加工用时短的工序[16-17]; 另一种是选择实质路径上的工序[16-17]. 现将综合柔性调度中的这两种工序选择方案设计成适用于解决存在多设备加工工序的柔性车间调度问题的两种调度算法, 分别称为方案1和方案2; 同时, 将文献[14]中解决树状结构产品的调度算法进行适当修改, 以适应于解决本文研究的DAG结构的产品生产调度问题. 用方案1、方案2和修改后的文献[14]中的调度算法(记作方案3)对图4中的实例进行调度, 并与本文提出的调度算法进行对比以说明本文所提出算法的优越性.

由于综合柔性调度问题的工艺流程是树状结构, 本文研究的柔性车间调度问题的工艺流程是网状结构, 两种调度问题的工艺流程结构不同, 要将基于事件驱动的树状结构产品综合柔性调度算法转换成适用于解决网状结构产品的柔性车间调度问题的算法, 需要解决计算每个结点的路径长度问题. 本文采用与综合柔性调度相似的方式计算结点的路径长度, 按短用时策略将传统柔性车间调度问题转换为一般车间调度问题, 并计算工序结点的路径长度, 路径长度的计算方法是每个结点的路径长度等于从汇点到当前结点的最长路径上的所有工序的加工用时的总和. 路径长度的计算问题解决后, 即可设计本文的两种对比调度方案和将文献[14]中的调度方案进行适当修改以适应解决本文产品的调度问题. 本文设计的两种对比调度方案(方案1和方案2)和对文献[14]修改后形成的方案3分别如下.

方案1 优先调度加工结束时间早的工序(设备驱动时刻, 加工用时短的工序即为加工结束时间早的工序).

具体步骤如下:

Step 1: 设备驱动时刻T_k, 计算空闲设备的个数Δm.

Step 2: 按设备号递增的顺序, 对空闲设备M_i建立加工工序集(空闲设备M_i的加工工序集由T_k时刻可调度工序集中可在该空闲设备上加工的工序组成).

Step 3: 从空闲设备M_i的加工工序集中选择加工用时最短的工序作为设备M_i上的预调度工序, 如果加工工序集中只有1个工序, 则该工序即为设备M_i上加工用时最短的工序; 如果存在多个最短加工用时相同的工序, 则选择等待时间长的可调度工序作为预调度工序.

Step 4: 判断预调度工序的加工类型. 如果为加工类型I或II, 则转Step 5; 如果为加工类型III, 则转Step 6.

Step 5: 直接调度该工序在空闲设备M_i上加工, 加工设备数$x = 1$, 转Step 9.

Step 6: 判断该预调度工序的其余协同加工设备是否全部处于空闲状态, 如果是, 转Step 7, 否则, 转Step 8.

Step 7: 计算协同设备个数γ, $x = \gamma$, 调度该工序到x个协同设备上加工, 转Step 9.

Step 8: 将该工序从该设备的加工工序集中删除, 判断删除后空闲设备M_i的加工工序集中是否还有工序, 如果有, 则转Step 3, 否则, 转Step 9.

Step 9: $\Delta m = \Delta m - x$, 如果$\Delta m = 0$, 则转

Step 10, 否则, 转 Step 2.

Step 10: 结束.

方案 1 的调度甘特图如图 6(b) 所示.

方案 2　优先调度实质路径上的工序.

具体步骤如下:

Step 1: 设备驱动时刻 T_k, 计算空闲设备的个数 Δm.

Step 2: 按设备号递增的顺序, 对空闲设备 M_i 建立加工工序集 (空闲设备 M_i 的加工工序集由可调度工序集中可在该空闲设备上加工的工序组成).

Step 3: 从加工工序集中选择路径最长的工序作为 T_k 时刻空闲设备 M_i 上的计划调度工序, 若路径长度最长值存在多个, 则按短用时优先策略选取加工用时较短的工序为计划调度工序; 若加工工序集中只有一个可调度工序, 则该工序即为预调度工序.

Step 4: 判断该计划调度工序的加工类型, 如果加工类型为 I 或 II, 则转 Step 5; 如果加工类型为 III, 则转 Step 6.

Step 5: 判断空闲设备 M_i 的加工工序集中是否还有其他加工类型为 I 或 II 的可调度工序, 如果有, 则转 Step 5.1, 否则, 转 Step 5.2.

Step 5.1: 计算并比较该计划调度工序与加工工序集中其他加工类型为 I 或 II 的工序形成的实质路径长度, 选择实质短路径上的可调度工序为计划调度工序, 若实质短路径存在多条, 则选择加工用时最长的可调度工序为计划调度工序.

Step 5.2: 调度该计划调度工序在当前设备上加工, 加工设备数 $x = 1$, 转 Step 7.

Step 6: 判断该计划调度工序的协同设备是否全部处于空闲状态, 若是, 则转 Step 6.1, 否则, 转 Step 6.2.

Step 6.1: 计算协同设备个数 $\gamma, x = \gamma$, 调度该计划调度工序到 x 个协同设备上加工, 转 Step 7.

Step 6.2: 不予调度该计划调度工序, 将该计划调度工序从空闲设备 M_i 的加工工序集中删除, 判断删除后的加工工序集中是否还有工序, 如果有, 则转 Step 3, 如果没有, 则转 Step 7.

Step 7: $\Delta m = \Delta m - x$, 如果 $\Delta m = 0$, 则转 Step 8, 否则, 转 Step 2.

Step 8: 结束.

方案 2 的调度甘特图如图 6(c) 所示.

方案 3　对于文献 [14] 的改进调度算法, 主要过程如下:

Step 1: 将多设备同时加工的工序虚拟成工序组, 使工艺调度关系图变为虚拟工艺调度关系图.

Step 2: 将虚拟工艺调度关系图分解为以虚拟工序组为汇点的子图和剩余标准工序组成的子图.

Step 3: 采用优先调度虚拟工序组中虚拟工序的策略, 确定以虚拟工序组为汇点的子图的调度顺序和各子图中工序的调度顺序.

Step 3.1: 子图的调度顺序是, 优先调度包含虚拟工序组个数较多的子图, 当虚拟工序组个数相同时, 子图中包含的工序结点加工用时之和越多 (路径长度越长), 优先级越高.

Step 3.2: 子图中工序的调度顺序是, 对于每个以虚拟工序组为汇点的子图, 优先调度虚拟工序组个数较多的路径上的工序. 路径不唯一时, 则优先调度加工用时长的路径上的工序.

Step 4: 重复 Step 3, 确定所有工序的调度顺序.

按照方案 3 确定的子图及子图中工序的调度顺序为 $\{P_{21}, P_{24}, P_{22}, P_{23}, P_{25}, P_{27}, P_{28}, P_{26}, P_{29}\}$, $\{P_{11}, P_{12}, P_{13}, P_{14}, P_{16}, P_{15}, P_{17}, P_{18}, P_{19}\}$, $\{P_{33}\}$, $\{P_{31}, P_{32}, P_{34}, P_{36}\}$. 调度甘特图如图 6(d) 所示.

通过对比图 6(a)、图 6(b)、图 6(c) 和图 6(d) 可以看出, 本文提出的算法调度结果更优. 虽然在方案 1 和方案 2 中, 每个驱动时刻的空闲设备都分配到了加工用时最短的工序或实质短路径上的工序, 但是方案 1 和方案 2 没有对加工类型的工序予以优先调度处理, 使得加工类型为 I 或 II 的工序较早地被加工而占用加工设备, 导致出现加工类型 III 的工序的加工设备被占用而迟迟得不到加工的情况. 方案 3 中的调度算法虽然优先调度了加工类型为 III 的工序和其所在的子图 (即工序组) 中的工序, 以工序组为单位进行调度, 必然会在设备上产生大量的不可用的空闲时间段, 因而影响产品的总完工时间. 而本文提出的算法优先调度加工类型 III 的工序, 使该类型的工序预约到一个较早的时刻开始加工, 尽量避免出现一部分协同设备长时间等待另一部分被占用的协同设备, 加工结束后才能同时开始加工的情况, 因而缩短了产品的制造时间.

7　结　论

本文研究了柔性车间调度中工序间的冗余调度约束关系问题和工序-设备间的加工方式问题, 提出了基于剪枝分层的柔性加工车间调度算法. 主要结论如下:

1) 所提出的剪枝分层法既可以最小化工序间的调度次序约束关系, 又可以使工序间的工艺加工流程层次清晰.

2) 所提出的根据加工类型设置工序处理顺序优

先级可以使加工类型III的工序的加工要求优先得到满足.

3) 所提出的工序-设备预约策略可以使加工类型III的工序预约协同设备到一个较早的时刻开始加工.

4) 所提出的工序-设备预分配策略可以使加工类型I和II的工序预分配到尽早加工结束的空闲设备上.

5) 所提出的柔性加工策略可以实现工序-设备预约策略中预约工序的加工和工序-设备预分配策略中预分配工序的加工.

本文提出的调度算法对深入研究柔性车间调度问题具有一定的理论价值和实际意义,可以为进一步研究多品种小批量产品的生产调度问题提供一定的参考借鉴作用.

参考文献(References)

[1] Zhou W, Bu Y P, Zhou Y Q. Combining CA and PSO to solve flexible job shop scheduling problem[C]. The 26th Chinese Control and Decision Conf. Changsha, 2014: 1031-1036.

[2] Rez M A, Raupp F M. A newton-based heuristic algorithm for multi-objective flexible job-shop scheduling problem[J]. J of Intell Manuf, 2016, 27(2): 409-416.

[3] Amirghasemi M, Zanani R. An effective asexual genetic algorithm for solving the job shop scheduling problem[J]. Computer & Industrial Engineering, 2015, 83(C): 123-138.

[4] Taisch M. Multi-objective genetic algorithm for energy-efficient job shop scheduling[J]. Int J of Production Research, 2015, 53(23): 7071-7089.

[5] Sun W, Pan Y, Lu X H, et al. Research on flexible job-shop scheduling problem based on a modified genetic algorithm[J]. J of Mechanical Science and Technology, 2010, 21(10): 2119-2125.

[6] Xing L N, Chen Y W, Wang P, et al. A knowledge-based ant colony optimization for flexible job shop scheduling problems[J]. Applied Soft Computing, 2010, 10(3): 888-896.

[7] Gao J, Sun L, Gen M. A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems[J]. Computers & Operations Research, 2008, 35(9): 2892-2907.

[8] Xie Z Q, Hao S Z, Ye G J, et al. A new algorithm for complex product flexible scheduling with constraint between jobs[J]. Computers & Industrial Engineering, 2009, 57(3): 766-772.

[9] 方晨, 王凌. 资源约束项目调度研究综述[J]. 控制与决策, 2010, 25(5): 641-650.
(Fang C, Wang L. Survey on resource-constrained project scheduling[J]. Control and Decision, 2010, 25(5): 641-650.)

[10] 王军强, 张松飞, 陈剑, 等. 一种求解资源受限多项目调度问题的分解算法[J]. 计算机集成制造, 2013, 19(1): 83-96.
(Wang J Q, Zhang S F, Chen J, et al. Decomposition algorithm for resource-constrained multi-project scheduling problem[J]. Computer Integrated Manufacturing Systems, 2013, 19(1): 83-96.)

[11] 黄学文, 马雪丽, 曹德弼. 工序顺序柔性的作业车间调度问题的改进遗传算法求解[J]. 运筹与管理, 2013, 22(1): 65-70.
(Huang X W, Ma X L, Cao D B. Improved genetic algorithm for job-shop scheduling problem with process sequence flexible[J]. Operations Research and Management Science, 2013, 22(1): 65-70.)

[12] 贾艳, 李世其, 王峻峰. 一类资源受限项目调度问题的仿真方法[J]. 系统仿真学报, 2012, 24(11): 2243-2248.
(Jia Y, Li S Q, Wang J F. Simulation method for a class of RCPSP[J]. J of System Simulation, 2012, 24(11): 2243-2248.)

[13] 唐海. 个性化概念图在网络自主学习中的应用研究[D]. 武汉: 武汉大学, 2010: 14-35.
(Tang H. Research on Individualized concept map applied in web-based learning[D]. Wuhan: Wuhan University, 2010: 14-35.)

[14] 谢志强, 齐永红, 杨静. 存在多设备工序的综合调度算法[J]. 机械工程学报, 2014, 50(24): 191-200.
(Xie Z Q, Qi Y H, Yang J. Integrated scheduling algorithm with multiple-devices-operation[J]. J of Mechanical Engineering, 2014, 50(24): 191-200.)

[15] 谢志强, 辛宇, 杨静. 可回退抢占的设备驱动综合调度算法[J]. 自动化学报, 2011, 37(11): 1332-1343.
(Xie Z Q, Xin Y, Yang J. Machine-driven integrated scheduling algorithm with rollback-preemptive[J]. Acta Automatica Sinica, 2011, 37(11): 1332-1343.)

[16] 谢志强, 周含笑, 于洁, 等. 基于设备驱动的综合柔性调度冲突调解算法[J]. 北京理工大学学报. 2014, 34(11): 1150-1156.
(Xie Z Q, Zhou H X, Yu J, et al. Conflict medication algorithm of integrated flexible scheduling based on device driver[J]. Trans of Beijing Institute of Technology, 2014, 34(11): 1150-1156.)

[17] 谢志强, 桂忠艳, 杨静. 基于设备驱动和实质路径的动态并行综合柔性调度算法[J]. 机械工程学报, 2014, 50(18): 203-212.
(Xie Z Q, Gui Z Y, Yang J. Dynamic parallel integrated flexible scheduling algorithm based on device driver and essential path[J]. J of Mechanical Engineering, 2014, 50(18): 203-212.)

（责任编辑：孙艺红）