考虑工序序列动态时间紧迫度的逆序贪婪综合调度算法

曹望成[①②] 谢志强[*①] 裴莉榕[①]

[①](哈尔滨理工大学计算机科学与技术学院 哈尔滨 150080)

[②](牡丹江师范学院计算机科学与技术系 牡丹江 157011)

摘 要：针对树状结构复杂单产品加工和装配的一般综合调度问题，该文提出考虑工序序列动态时间紧迫度(TUD)的逆序贪婪综合调度算法。提出工序排序策略，定义工序序列的时间紧迫度，将工序树逆置，采用叶对齐的方式，按照由叶到根的顺序，逐层根据叶结点所属工序序列动态时间紧迫度值由大到小的顺序确定其调度顺序，将各层排序后的叶结点依次入队列保存，最后将队列中元素逆置。提出逆序贪婪调度策略，每次以一道工序为单位，安排它在所需设备上的准调度时间点进行试调度，得到该工序的准调度方案集，选择准调度方案结束时间最小的方案，若不唯一，选择使该工序尽早加工的方案。实例表明所提算法优化了一般综合调度的结果且效率较高。

关键词：综合调度；加工和装配；时间紧迫度；逆序；贪婪

中图分类号：TP278 **文献标识码**：A **文章编号**：1009-5896(2022)05-1572-09

DOI: 10.11999/JEIT211455

Reverse Order and Greedy Integrated Scheduling Algorithm Considering Dynamic Time Urgency Degree of the Process Sequences

CAO Wangcheng[①②] XIE Zhiqiang[①] PEI Lirong[①]

[①](*College of Computer Science and Technology, Harbin University of Science and Technology, Harbin* 150080, *China*)

[②](*Department of Computer Science and Technology, Mudanjiang Normal University, Mudanjiang* 157011, *China*)

Abstract: For the general integrated scheduling problem of tree structured complex single product machining and assembling, a reverse order and greedy integrated scheduling algorithm is proposed by considering dynamic TUD (Time Urgency Degree) of the process sequences. The strategy of process sorting is put forward, and the TUD of process sequence is defined. The process tree is reversed using leaf alignment, according to the order from leaf to root, the scheduling order of leaf nodes in the same layer is determined layer by layer from large to small according to the dynamic TUD values of the process sequences to which the leaf nodes belong. The sorted leaf nodes are put into the queue in turn. Finally, the elements in the queue are reversed. A reverse order and greedy scheduling strategy is proposed. Each time, a single process is taken as a unit to conduct trial scheduling at the quasi-scheduling time point in the required equipment. Quasi-scheduling scheme set of the process is obtained, and the quasi-scheduling scheme with the minimum end time is selected, and if it is not unique, the scheme is selected to machine the process as early as possible. A case shows that the proposed algorithm optimizes the general integrated scheduling results and has high efficiency.

Key words: Integrated scheduling; Machining and assembling; Time Urgency Degree (TUD); Reverse order; Greedy

收稿日期：2021-12-08；改回日期：2022-03-26；网络出版：2022-04-22

*通信作者： 谢志强 xiezhiqiang@hrbust.edu.cn

基金项目：黑龙江省教育厅重点科技项目(1355ZD004)，国家自然科学基金(61772160)

Foundation Items: The Key Science and Technology Projects of Heilongjiang Provincial Education Department (1355ZD004), The National Natural Science Foundation of China (61772160)

1　引言

调度优化问题的典型实例有制造车间中工序[1]的排序优化问题，网格计算中工作流[2–4]的费用优化问题和云计算中云服务工作流应用程序[5,6]或MapReduce任务[7]的排序优化问题。

智能制造已作为核心专项工程纳入《中国制造2025》实施战略。全国“两会”政府工作报告[8]、《中国制造2025》和国务院关于积极推进“互联网＋”行动的指导意见中均明确指出我国将大力发展产品个性化定制。消费者对产品多元化及个性化需求的不断增加，促使产品类型趋向多品种、小批量。当制造多品种、小批量产品，特别是单件复杂结构产品时，如果按照传统的先分步加工调度最后整体装配调度的生产模式排产，必将割裂加工工序和装配工序可并行处理的关系，降低复杂单产品生产效率。

谢志强等人[9–11]提出了第3类产品制造调度模式：加工工序和装配工序一同处理的综合调度，简称综合调度。综合调度是智能制造的核心组成部分，更是实现中国制造2025和工业4.0[12]的重要手段，学者针对综合调度问题的算法优化做了大量研究工作。针对单车间一般综合调度问题，谢志强等人[13,14]提出的考虑串行工序紧密度的择时综合调度算法[13]和考虑后续工序的择时综合调度算法[14]，存在以下缺点：

(1)文献[13]的择时综合调度策略为每道工序选择当前完工总用时最小的方案，忽略了该道工序对属于同一工序序列的后续串行工序的影响，导致算法出现局部最优而影响产品调度结果的弊端；

(2)文献[14]的工序排序策略与文献[13]相同，在应用择时综合调度策略调度工序时不能总是优先处理对生产周期有较大影响的工序序列上的工序，必须回溯调整[15]，导致算法复杂度偏高。

综上所述，本文提出串行工序序列的时间紧迫度(TUD)定义，应用所提工序排序策略得到工序队列，再结合所提逆序贪婪调度策略，以工序为单位建立其工序调度方案，最后一道工序的调度方案即为产品调度方案。

2　问题模型描述

为便于综合调度，将加工工序和装配工序统一定义为工序，加工设备和装配设备统一定义为设备。复杂单产品进行综合调度时需满足以下要求。(1)所有工序的加工时间已知而且与其加工顺序无关，产品工序间的约束关系事先已知。(2)每道工序只能由一台设备加工，某时刻每台设备只能加工一道工序，工序一旦开始被加工则直至加工结束不能被间断。(3)允许工序之间等待，允许设备在工序到达之前空闲。(4)当工序无紧前工序或其紧前工序均被加工完毕时，该工序才能被加工。(5)车间内不存在相同设备。

个性化定制复杂结构单产品(如飞行器)的加工工艺拓扑关系结构图呈树状结构，结点间有向边的方向与树结构中相反，称为产品加工工序树，简称工序树。为方便叙述，文中“结点”等价于他所表示的“工序”。

定义1　逆序工序树。将工序树中有向边的方向取反之后所得到的树型结构。

定义2　工序序列(Process Sequence, PS)。逆序工序树中，从根结点或者某叉点孩子结点a开始，沿有向边的方向遍历结点，直到叶结点b为止，彼此之间具有串行偏序关系的结点序列，称为工序a到工序b的工序序列，记为PS_{ab}。特别地，当工序序列只包含单个工序c时，此工序序列记为PS_{cc}。

定义3　工序序列长度。工序序列中全部结点所代表工序的加工时间之和。

定义4　结点的路径。逆序工序树中，从根结点开始沿有向边的方向遍历到当前结点，所得到的有序结点序列称为当前结点的路径。

定义5　结点的路径长度。结点的路径上全部结点的加工时间之和。

定义6　关键路径。当前逆序工序树中，长度最长的叶结点的路径，若不唯一，选择包含工序数多的路径为关键路径，若仍不唯一，任选其一。

定义7　工序队列。用于存放初始逆序工序树中的全部结点按照工序排序策略所得到的排序结果，为方便描述，队列中的结点简写为该结点所代表的工序名称。

定义8　准调度时间点。根据工艺约束关系，某工序的紧前工序加工结束时间点之后其对应的设备上所有空闲时间段的开始时刻都是该工序的准调度时间点。

定义9　准调度时间点队列。存放某工序按由小到大排序的准调度时间点，用QT表示。

定义10　调度方案。记录当前所有设备上已试调度工序的信息：包括方案名称、完工总用时和所有的设备列表，设备列表中的设备记录该设备上按时间先后顺序已试调度工序的信息。

定义11　工序准调度方案。将工序队列中的第k道工序在其设备上的X个准调度时间点进行试调度，X个调度方案都是其工序准调度方案，分别记为$P_k^1, P_k^2, \cdots, P_k^X$。

定义12 工序调度方案。设工序队列中第k道工序试调度共产生X个准调度方案，它们构成准调度方案集$P_k(X)$，从$P_k(X)$中选择完工时间最小的方案为工序调度方案，记为P_k，若不唯一，则从中选择使该道工序尽早开始加工的方案。

定义13 产品调度方案。产品的全部n道工序试调度后产生的最佳调度方案，用P表示。

假设复杂单产品包含n道工序，需要m台设备。A_i表示编号为$i(1\le i\le n)$的工序，s_{ij}表示A_i在设备$M_j(1\le j\le m)$上的加工开始时间，t_{ij}表示其在设备M_j上的连续加工时间。工序队列中第1道工序为根结点工序，其加工开始时刻为0，对应的工序调度方案为P_1。设工序队列中第k道工序的工序调度方案为P_k，则P_n即为P，一般综合调度问题的目标函数及约束条件描述为

$$T=\min(P_n.\text{totaltime}) \tag{1}$$

s.t.

$$\min(s_{ij}),\ 1\le i\le n,\ \ 1\le j\le m \tag{2}$$

$$s_{xy}\ge s_{ij}+t_{ij}, 1\le x,i\le n,\ \ 1\le y,j\le m \tag{3}$$

$$s_{i+1j}\ge s_{ij}+t_{ij},\ 1\le i\le n,\ \ 1\le j\le m \tag{4}$$

$$\begin{aligned}&P_k=P_k^y,\\&P_k^y.\text{totaltime}=\min(P_k^z.\text{totaltime}),\\&1\le k\le n,\ 1\le y,z\le X\end{aligned} \tag{5}$$

$$\begin{aligned}&P_k.\text{totaltime}=\max(\text{Mj.finishtime}),\\&1\le k\le n,\ \ 1\le j\le m\end{aligned} \tag{6}$$

$$\begin{aligned}&P_k.\text{totaltime}\le P_{k+1}.\text{totaltime},\\&1\le k\le n-1\end{aligned} \tag{7}$$

$$\left.\begin{aligned}&P_k=P_k^\delta,\\&s_{\alpha\beta}^\delta=\min(s_{\alpha\beta}^x,s_{\alpha\beta}^y,\cdots,s_{\alpha\beta}^z)\\&\bigcap(P_k^x.\text{totaltime}=P_k^y.\text{totaltime}\\&=\cdots=P_k^z.\text{totaltime}=\min(P_k^w.\text{totaltime})),\\&1\le\alpha,k\le n,\ \ 1\le x,y,z,w,\delta\le X,\\&1\le\beta\le m,\ \ \delta\in\{x,y,\cdots,z\},\\&s_{\alpha\beta}^x\in P_k^x,\ \ s_{\alpha\beta}^y\in P_k^y,\ \ \cdots,s_{\alpha\beta}^z\in P_k^z\end{aligned}\right\} \tag{8}$$

式(1)表示P_n的总用时为产品的完工时间；式(2)表示在满足式(3)，式(4)，式(5)的条件下使工序尽早开始加工；式(3)表示某工序必须在其紧前工序加工结束后才可以被加工，Ax为Ai的紧后工序；式(4)表示相同设备上的紧后工序必须要在其紧前工序加工结束后才可以被加工；式(5)表示P_k^y所用时间最小，则P_k^y被确定为P_k；式(6)表示P_k的总用时为当前所有设备完工时间的最大值；式(7)表示某道工序调度方案的总用时一定不大于工序队列中其紧后工序的工序调度方案的总用时；式(8)表示当第k道工序的准调度方案结束时间最小值不唯一时，选择使该道工序尽早开始加工的P_k^δ作为P_k，$s_{\alpha\beta}^\delta$表示工序队列中序号为α的Aα在Mβ上加工开始时间的最小值。

3 策略设计

本文中工序、设备和方案的结构描述如下。

(1)用链表存储产品初始逆序工序树，工序队列存储经排序后的全部结点，链表和队列中元素的结构为：Node- Aid :{int} /Mid:int /T:int /L:int /F:Node*/Q:Node*[]/Tb:int /Te:int。Aid是结点表示的工序名，Mid是工序对应的设备名，T是工序在机器Mid上的加工时间，L是结点的路径长度，F是指向其紧前工序的指针，Q是指向结点紧后工序的指针集合，Tb是工序实际开始加工时刻，Te是工序的加工结束时刻。

(2)调度方案的设备列表中设备结构为：$M-$Mid : int/NodeList : Node $*$ /finishtime : int。

Mid是设备名；NodeList是设备上已试调度的工序链表，工序按加工开始时刻由小到大排序；finishtime是当前设备完工时间的最大值。

(3)调度方案结构为：$P-$Pid : int/MachineList : $M*$ /totaltime : int。

Pid是方案名；MachineList是方案对应的设备列表；totaltime是方案中所有设备完工时间的最大值。

3.1 工序排序策略

3.1.1 工序排序策略分析

定义14 工序序列的时间紧迫度(Time Urgency Degree, TUD)。Ai表示当前逆序工序树关键路径上的叉点，$L_i(j)$表示以Ai的第j个孩子为起点的唯一最长工序序列的长度，将$L_i(j)$减去以Ai为紧前工序的最长工序序列的长度$\max(L_i(x)),(x=1,2,\cdots,j,\cdots)$的差值定义为以Ai为紧前工序的第$j$个工序序列的时间紧迫度，记为$\text{TP}_i(j)=L_i(j)-\max(L_i(x))$。

将$\text{TP}_i(j)$大小作为判断以第j个工序序列为关键路径的子树的完工时间对当前逆序工序树完工时间影响程度的基本依据。工序序列时间紧迫度越大表明其所属子树对当前逆序工序树的完工总时间影响越大，子树上工序对设备的需求紧迫度越大，为缩短产品生产周期，优先调度该子树上的工序。工序序列由若干加工时间不等的工序组成，随着时间紧迫度值大的工序序列上的某道或某些工序加工完毕，由该工序序列剩余工序所组成的工序序列的时间紧迫度值将会发生变化，即工序序列的时间紧迫度是动态变化的。

3.1.2 工序排序策略的算法设计

设置一个工序队列Queue、一个整型变量N、两个链表List0和List1。Queue存放工序排序策略的

结果；N是初始逆序工序树的层数；List0存储初始逆序工序树，List1保存排序前的工序树List0。

算法1：工序排序策略算法

(1) 初始化List0，List1为空；

(2) 创建逆序工序树List0：根据输入产品工序的信息、工序间偏序关系和设备信息创建工序结点，输入工序属性值，插入逆序工序树，属性L，Tb和Te初值为0；

(3) 计算各结点的路径长度和N，确定关键路径的长度T'，更新各结点L属性值：根结点的路径长度为根结点的加工时间，其余结点的路径长度为该结点的加工时间加上其紧前工序的路径长度；

(4) 初始化Queue；

(5) i=0；

(6) 用List1备份List0；

(7) 判断工序树中叶子结点个数是否为1，是转步骤(8)，否则转步骤(9)；

(8) $i++$，将叶子结点入Queue，转(11)；

(9) 调用算法2，对当前逆序工序树中叶子结点排序并依次入Queue；

(10)List0复制List1，在工序树中删除所有叶子结点，转步骤(6)；

(11)判断入队列结点的指针F是否为空，是转步骤(13)，否则转步骤(12)；

(12)从工序树中删除所有叶子结点，转步骤(7)；

(13)将Queue中的元素逆置；

(14)退出：返回Queue，N。

3.1.3 当前逆序工序树中叶子结点排序算法设计

设置1个1维指针数组$a[k](k=0,1,\cdots,n-2)$，1个1维整型数组$b[k](k=0,1,\cdots,n-2)$，堆栈Stack1和Stack2。排序过程中，$a[k]$存放指向以关键路径上各叉点的孩子结点作为起点的唯一最长工序序列的指针，$b[k]$存放$a[k]$元素所指向的各工序序列的时间紧迫度的数值，将$a[k]$中元素按$b[k]$中对应数值由小到大排序；Stack1存放某工序序列所包含的全部叉点；Stack2存放数组$a[k]$从前到后的各个元素。

算法2：当前逆序工序树中叶子结点排序算法

(1) 初始化$a[k]$，$b[k]$，Stack1，Stack2为空，$j=0$；

(2) i++；

(3) 确定关键路径，将其作为初始逆序工序树中时间紧迫度最大的工序序列；

(4) $j++$，将关键路径最后一个结点存入Queue，q_j为指向该工序序列首结点的指针；

(5) 判断指针q_j指向的结点是否为叉点，是转步骤(6)，否则转步骤(7)；

(6) 叉点入Stack1；

(7) 指针后移指向下一结点；

(8) 判断指针否为空，是转步骤(9)，否则转步骤(5)；

(9) 从工序树中删除此时间紧迫度最大的工序序列中的工序；

(10)判断Stack1是否为空，是转步骤(16)，否则转步骤(11)；

(11)Stack1出栈，在逆序工序树森林中，为出栈结点的各紧后工序找到以其为起点的一个长度最长的工序序列，若不唯一，选择工序数多的，用$a[k++]$保存指向各最长工序序列的首结点的指针；

(12)依次计算各工序序列的时间紧迫度并存入$b[k++]$：为便于计算，方法是用各工序序列叶结点的路径长度减去初始工序树关键路径长度；

(13)判断Stack1是否为空，是转步骤(14)，否则转步骤(11)；

(14)将$a[k]$中元素按$b[k]$中对应数值由小到大排序，数值相等时按工序序列所包含的工序数由小到大排序；

(15)将$a[k]$元素从前到后依次入Stack2保存，清空$a[k]$和$b[k]$；

(16)判断Stack2是否为空，是转步骤(18)，否则转步骤(17)；

(17)Stack2出栈，出栈指针所指工序序列作为其所属子树的时间紧迫度最大的工序序列，转步骤(4)；

(18)退出。

3.2 逆序贪婪调度策略

3.2.1 逆序贪婪调度策略分析

工序队列中第1个元素为根结点，安排其在所需设备上的“0”时刻试调度形成唯一的P_1。试调度工序队列中序号为$i(i\geq 2)$的工序时，其紧前工序已被试调度完毕，因此序号为i的工序的准调度时间点已知，设QT中元素个数为X，可形成X个工序准调度方案，从中选择结束时间最小的准调度方案，若不唯一，选择使该工序最早开始加工的方案作为P_i，依次类推，直到序号为n的工序试调度结束，P_n即为P。

逆序贪婪调度的优点有3个：(1)因为是按由根到叶的顺序试调度各工序，在试调度Queue中序号为$i(i\geq 2)$的工序时，约束关系被破坏的已试调度工序数量少，只需考虑序号小于等于$i-1$的同设备工序；(2)当序号为$i-1$的工序试调度结束，序号为i的工序的“准调度时间点”便随之确定，算法效率高；(3)逆序贪婪调度以单个工序为单位进行试调度，形成P_i时即实现了前i道工序的充分并行处理。

3.2.2 逆序贪婪调度策略算法设计

工序调度方案$P_i(2 \le i \le n)$的建立以P_{i-1}为基础，首先确定工序队列中序号为i的工序的QT，依次安排它在各准调度时间点进行试调度，当该工序的加工完成时间大于同设备上紧后工序的加工开始时间时，后移紧后工序并依次调整约束关系被破坏的工序的位置，通过对准调度方案的选取得到P_i。

算法3：建立工序调度方案$P_i(2 \le i \le n)$的逆序贪婪调度策略算法

(1) 序号为i的结点出队列Queue：$\mathrm{A}\alpha = (\mathrm{front} \to \mathrm{data}).\mathrm{Aid}$；

(2) 确定Aα紧前工序的加工结束时间点：$t(((\mathrm{front} \to \mathrm{data}).F) \to \mathrm{data}).\mathrm{Te}$；

(3) 在P_{i-1}基础上为Aα寻找设备Mβ上t之后的k个“准调度时间点”，按从小到大的顺序入QT，$j = 1$；

(4) 判断QT是否为空，若不为空，QT出队列，得第j个T，转步骤(5)，否则转步骤(8)；

(5) 对Aα在T时刻进行试调度：即$(\mathrm{front} \to \mathrm{data}).\mathrm{Tb} = T$，$(\mathrm{front} \to \mathrm{data}).\mathrm{Te} = T + (\mathrm{front} \to \mathrm{data}).T$；

(6) 利用文献[13]的择时调整策略对试调度Aα后约束关系被破坏的工序进行调整；

(7) 形成P_i^k，$k = j++$，转步骤(3)；

(8) 对$P_i^1, P_i^2, \cdots, P_i^k$的完工总用时进行比较，选择完工总用时最小的方案；

(9) 判断完工总用时最小的方案是否唯一，不唯一转步骤(10)，否则转步骤(11)；

(10)选择使Aα加工开始时间$s_{\alpha\beta}$最小的方案作为P_i；

(11)刷新Aα的各属性值、设备Mβ信息和P_i的信息；

(12)退出。

4 算法设计与复杂度分析

4.1 算法设计

首先应用算法1对全部工序进行排序，形成工序队列；其次对工序队列中序号为1的根结点进行调度，形成P_1，再对序号为$i(2 \le i \le n)$的工序应用算法3建立P_i；最后形成甘特图并输出。

算法4：考虑工序序列动态时间紧迫度的逆序贪婪综合调度算法

(1) 应用算法1对工序排序，得到包含$n(n \ge 1)$个工序的Queue；

(2) 试调度根结点工序，形成P_1；

(3) i=2；

(4) 判断$i \le n$是否成立，是转步骤(5)，否则转步骤(7)；

(5) 应用算法3建立P_i；

(6) $i++$，转步骤步骤(4)；

(7) 形成甘特图并输出；

(8) 退出。

4.2 算法复杂度分析

假设产品的工序数为n，有m台设备，逆序工序树的层数为N。

文中算法4是总算法，在算法4中调用1次算法1和循环$n-1$次调用算法3。算法1调用了算法2。因而算法4的时间复杂度为总的时间复杂度，即算法1和循环$n-1$次调用算法3的时间复杂度之和。

4.2.1 算法1的时间复杂度

算法1主要包括以下4个操作：

(1)建立逆序工序树

建立逆序工序树是根据输入的产品工序信息、工序间的偏序关系及设备信息建立链表的过程，为链表中的n个结点的属性赋初值，时间复杂度为n的整数倍，时间复杂度为$O(n)$。

(2)计算工序路径长度和工序树的层数

计算工序路径长度和工序树的层数只需要对逆序工序树由根开始进行1次遍历，时间复杂度为$O(n)$。

(3)调用算法2

影响算法2时间复杂度的核心操作可简单地描述为以下4个步骤：

(a)获取当前逆序工序树中的叶子结点；

(b)对步骤1中得到的叶子结点进行排序，将排序后的结点追加到工序队列的末端；

(c)删除当前逆序工序树的叶子结点，得到新的当前逆序工序树；

(d)重复步骤(a)，步骤(b)，步骤(c)，直到所有的结点都进入工序队列。

4个步骤中对前3个步骤的循环次数为初始逆序工序树的层数N，分析前3个步骤的时间复杂度。

步骤(a)只需要对当前逆序工序树进行1次遍历，设工序树中结点个数为$n_i(i = 1, 2, \cdots, N)$，显然有$n_i < n$，步骤1循环N次访问结点的总次数$n_1 + n_2 + \cdots + n_N < n \times N \le n^2$，时间复杂度为$O(n^2)$。

步骤(b)每次进行排序的叶子结点数平均为n/N，进行排序的时间复杂度为$O((n/N)^2)$，循环N次的时间复杂度为$O(N \times (n/N)^2) = O(n^2/N)$。

步骤(c)删除当前逆序工序树的叶子结点只需要对当前工序树进行1次遍历，循环N次总的时间复杂度与步骤(a)相同，最大为$O(n^2)$。

因此调用算法2的时间复杂度为$O(n^2)$。

(4)将工序队列Queue中的元素逆置

借助一个堆栈即可实现工序队列Queue中n个元素的逆置，时间复杂度为$O(n)$。

综合分析，算法1的时间复杂度为以上4个操作的时间复杂度之和，即为$O(n^2)$。

4.2.2 循环调用算法3的时间复杂度

算法3中建立P_i的过程是在P_{i-1}的基础上对Queue中的第i个工序Aα按贪婪调度策略进行调度的过程。这个过程的关键操作是在P_{i-1}中Aα的准调度时间点之后，Aα所需设备上求出各个准调度方案，选择结束时间最小的准调度方案作为Aα的P_i。设Aα的QT中元素个数为c_i，则c_i是影响时间效率的决定因素。因此循环调用算法3的时间复杂度由$\sum_{i=2}^{n} c_i$决定，显然有$c_i \le i-1 (i=2,3,\cdots,n)$，QT中元素个数$c_i$按最大值计算，因为$\sum_{i=2}^{n} c_i \le 1+2+\cdots+(n-1)=n(n-1)/2$，所以循环调用算法3的时间复杂度为$O(n^2)$。

综合以上分析，本文算法的时间复杂度为$O(n^2)$。

5 算法实例与对比分析

本文提出的算法是基于理论分析，不针对任何具体实例，具有普遍意义，为了方便读者了解该算法，下面借助产品调度实例进一步说明。设制造企业的单件订单产品A包含37道工序，需要4台设备，产品的逆序工序树如图1所示。逆序工序树中结点用长方框表示，长方框内的信息是对应工序4个属性的简写：工序名的编号i|设备名的编号j|工序的加工时间|结点的路径长度，其中加工时间为单位时间(h)。

应用文献[11]的策略确定产品A中工序的调度次序为：A37，A34，A36，A35，A32，A29，A33，A31，A30，A25，A26，A27，A24，A21，A20，A28，A22，A23，A19，A16，A15，A17，A18，A12，A9，A13，A14，A10，A11，A8，A7，A6，A2，A4，A5，A3，A1。同设备工序A17与A18的路径长度分别为9和8，加工时间都是1 h，A18所在的路径为关键路径，文献[11]只关注工序的路径长度，忽略了工序所在分支对产品周期的整体影响，优先处理工序A17造成调度结果欠佳，产品生产周期为26 h。

应用文献[13]的工序序列排序策略将产品A的10个工序序列按路径长度由大到小排序。调度过程中，以关键路径形成的初始调度方案为基础，每调度1道工序形成多个准调度方案，从中选择当前最佳调度方案，容易使算法陷入局部最优。比如在调度工序A4时，A4的4个准调度时间点分别为2，6，11，17，文献[13]选择在时间点11调度A4，虽然当前方案总用时最少，但使得设备M3上A1与A7，A7与A23之间出现了空闲时间段，且A4所在工序序列的完成时间滞后，整体调度结果欠佳，产品生产周期为32 h。应用文献[11]和文献[13]，调度甘特图如图2和图3所示。

应用本文所提算法调度产品A。应用算法1对

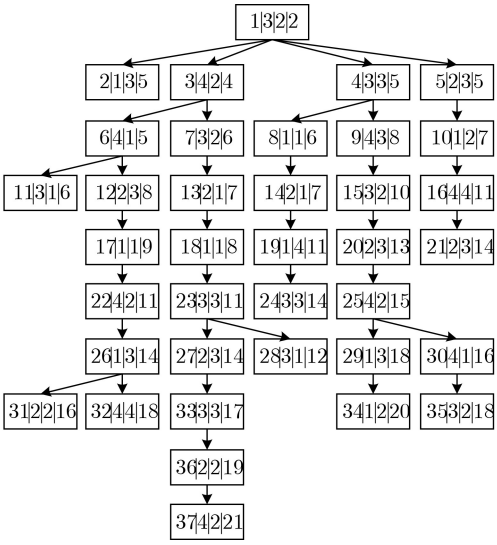

图 1 产品A的逆序工序树

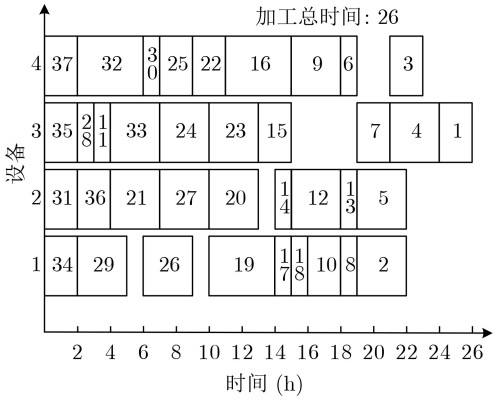

图 2 使用文献[11]算法调度产品A所得调度甘特图

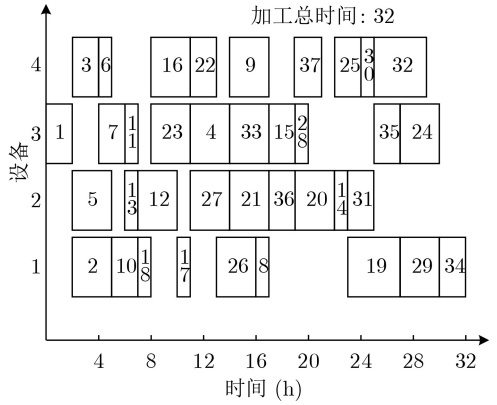

图 3 使用文献[13]算法调度产品A所得调度甘特图

工序进行排序，其中排序第1层叶子结点时的工序序列划分示意图如图4所示，工序排序结果为：A37，A34，A35，A24，A32，A31，A11，A21，A28，A2。

全部工序排序后，将Queue中的元素逆置，Queue中从前到后为：A1，A3，A7，A4，A13，A6，A9，A18，A12，A15，A23，A5，A17，A8，A20，A27，A10，A22，A14，A25，A33，A16，A26，A19，A30，A29，A36，A2，A28，A21，A11，A31，A32，A24，A35，A34，A37。Queue出队列，试调度工序，只有试调度A4，A12，A16，A24建立P_4，P_9，P_{22}，P_{34}的过程中涉及准调度时间点的选取。A4的准调度时间点为2和6，选择2；A12的准调度时间点为5和8，选择5；A16的准调度时间点为8, 11和15，选择8；A24的准调度时间点为14和19，选择19。4次试调度过程如图5所示，灰色工序对应的方案作为工序调度方案。本文算法调度产品A的调度甘特图如图6所示，生产周期为24 h。

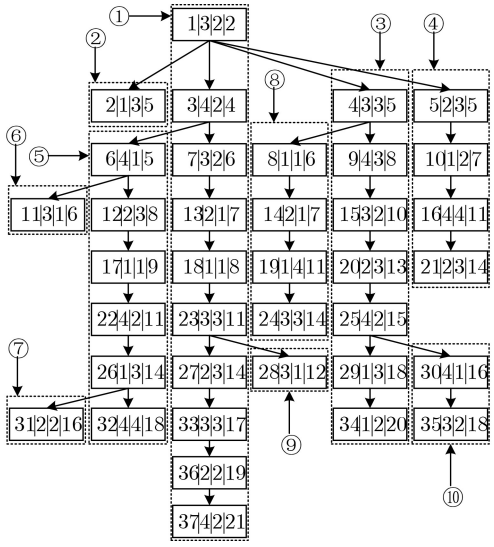

图 4 排序第1层叶子结点时的工序序列划分示意图

应用文献[14]和文献[15]算法调度产品A，调度结果与本文算法相同，产品生产周期为24 h，但因它们的工序排序策略与文献[13]相同，不能优先调度时间紧迫度值大的工序序列上的工序，所以算法复杂度高于本文。

对比发现，应用本文算法调度产品A时，不但可以缩短产品的生产周期而且效率较高。

6 结束语

本文主要结论如下：

(1)工序排序策略，提出工序序列时间紧迫度

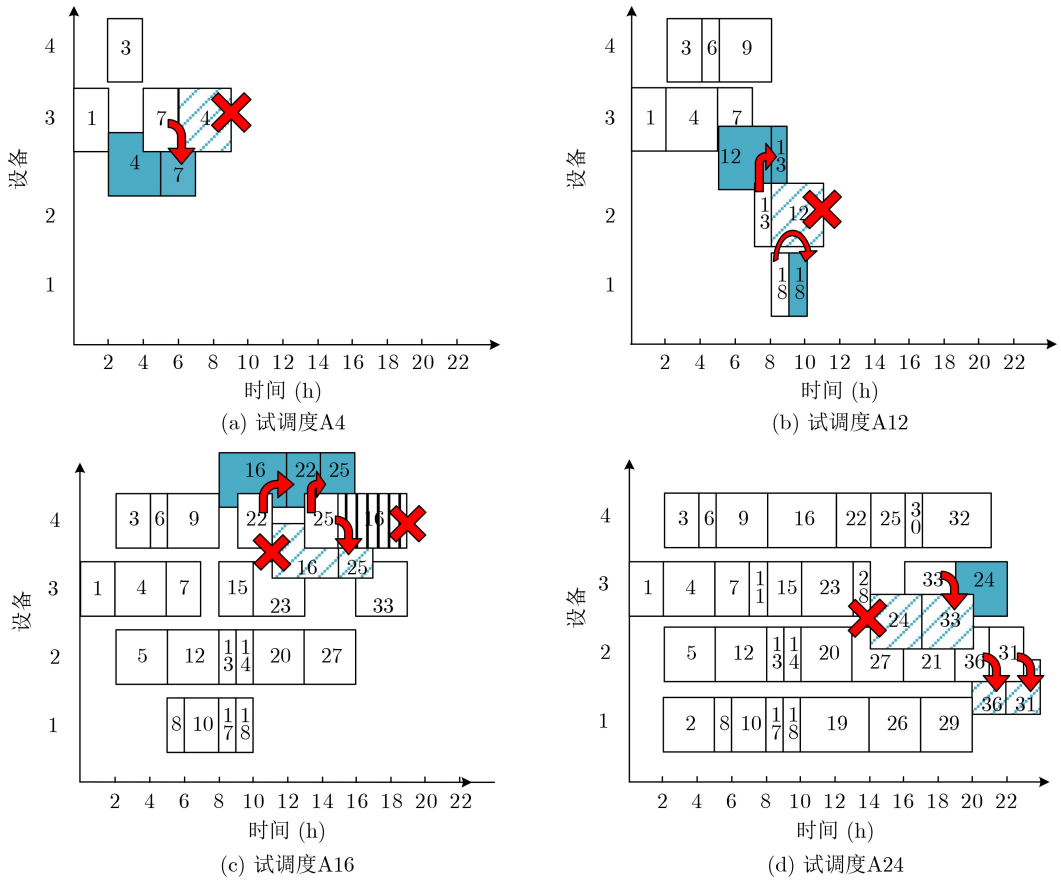

图 5 4次试调度工序的过程示意图

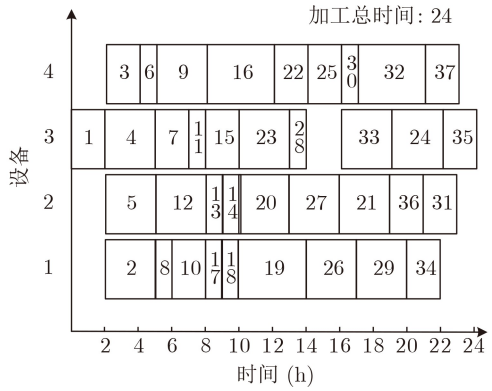

图 6 使用本文算法调度产品A所得调度甘特图

的定义，将同层工序按所属工序序列时间紧迫度值由大到小的顺序确定调度顺序，优化了调度结果；克服了文献[13]的排序策略导致调度结果易于陷入局部最优的缺点；

(2)逆序贪婪调度策略，每道工序只需考虑在空闲的准调度时间点调度，算法复杂度不超过二次多项式。

综上所述，本文算法优于目前典型的一般综合调度算法，工序序列时间紧迫度概念的提出为进一步深入研究综合调度问题拓展了思路，具有一定的理论和实际意义。该算法注重缩短横向同层可调度工序并行调度结束时间的同时，更强调以动态时间紧迫度值大的工序序列为主的纵向调度优化思想，优化了综合调度结果。

参 考 文 献

[1] WANG Zhen, ZHANG Xiaohuan, CAO Yuxiao, *et al*. An integrated scheduling algorithm for multi-device-processes with the strategy of exchanging adjacent parallel processes of the same device[J]. *EURASIP Journal on Wireless Communications and Networking*, 2021, 2021(1): 104. doi: 10.1186/s13638-021-01989-1.

[2] 苑迎春, 李小平, 王茜, 等. 基于逆向分层的网格工作流调度算法[J]. 计算机学报, 2008, 31(2): 282–290. doi: 10.3321/j.issn:0254-4164.2008.02.012.
YUAN Yingchun, LI Xiaoping, WANG Qian, *et al*. Bottom level based heuristic for workflow scheduling in grids[J]. *Chinese Journal of Computers*, 2008, 31(2): 282–290. doi: 10.3321/j.issn:0254-4164.2008.02.012.

[3] 苑迎春, 李小平, 王茜. 基于串归约的网格工作流费用优化方法[J]. 计算机研究与发展, 2008, 45(2): 246–253.
YUAN Yingchun, LI Xiaoping, and WANG Qian. Cost optimization heuristics for grid workflows scheduling based on serial reduction[J]. *Journal of Computer Research and Development*, 2008, 45(2): 246–253.

[4] 苑迎春, 李小平, 王茜, 等. 成本约束的网格工作流时间优化方法[J]. 计算机研究与发展, 2009, 46(2): 194–201.
YUAN Yingchun, LI Xiaoping, WANG Qian, *et al*. Time optimization heuristics for scheduling budget-constrained grid workflows[J]. *Journal of Computer Research and Development*, 2009, 46(2): 194–201.

[5] LI Xiaoping, QIAN Lihua, and RUIZ R. Cloud workflow scheduling with deadlines and time slot availability[J]. *IEEE Transactions on Services Computing*, 2018, 11(2): 329–340. doi: 10.1109/TSC.2016.2518187.

[6] LI Xiaoping, YU Wei, RUIZ R, *et al*. Energy-aware cloud workflow applications scheduling with geo-distributed data[J]. *IEEE Transactions on Services Computing*, 2022, 15(2): 891–903. doi: 10.1109/TSC.2020.2965106.

[7] LI Xiaoping, CHEN Fuchao, RUIZ R, *et al*. MapReduce task scheduling in heterogeneous geo-distributed data centers[J]. *IEEE Transactions on Services Computing*, To be published. doi: 10.1109/TSC.2021.3092563.

[8] 李克强. 政府工作报告-2021年3月5日在第十三届全国人民代表大会第四次会议上[EB/OL]. http://www.gov.cn/gongbao/content/2021/content_5593438.htm?ivk_sa=1024320u, 2021.
LI Keqiang. Government work report - at the fourth session of the 13th National People's Congress on March 5, 2021[EB/OL]. http://www.gov.cn/gongbao/content/2021/content_5593438.htm?ivk_sa=1024320u, 2021.

[9] GAO Yilong, XIE Zhiqiang, YANG Dan, *et al*. Flexible integrated scheduling algorithm based on remaining work probability selection coding[J]. *Expert Systems*, 2021, 38(4): e12683. doi: 10.1111/exsy.12683.

[10] 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. doi: 10.3724/SP.J.1004.2011.01332.
XIE Zhiqiang, XIN Yu, and YANG Jing. Machine-driven integrated scheduling algorithm with rollback-preemptive[J]. *Acta Automatica Sinica*, 2011, 37(11): 1332–1343. doi: 10.3724/SP.J.1004.2011.01332.

[11] 谢志强, 杨静, 周勇, 等. 基于工序集的动态关键路径多产品制造调度算法[J]. 计算机学报, 2011, 34(2): 406–412. doi: 10.3724/SP.J.1016.2011.00406.
XIE Zhiqiang, YANG Jing, ZHOU Yong, *et al*. Dynamic critical paths multi-product manufacturing scheduling algorithm based on operation set[J]. *Chinese Journal of Computers*, 2011, 34(2): 406–412. doi: 10.3724/SP.J.1016.2011.00406.

[12] 张雍达, 宋嘉. 工业4.0时代的智能制造[J]. 中国工业和信息化,

2021(9): 32–34. doi: 10.19609/j.cnki.cn10-1299/f.2021.09.002.

ZHANG Yongda and SONG Jia. Intelligent manufacturing in the industrial 4.0 era[J]. *China Industry & Information Technology*, 2021(9): 32–34. doi: 10.19609/j.cnki.cn10-1299/f.2021.09.002.

[13] 谢志强, 张晓欢, 高一龙, 等. 考虑串行工序紧密度的择时综合调度算法[J]. 机械工程学报, 2018, 54(6): 191–202. doi: 10.3901/JME.2018.06.191.

XIE Zhiqiang, ZHANG Xiaohuan, GAO Yilong, *et al*. Time-selective integrated scheduling algorithm considering the compactness of serial processes[J]. *Journal of Mechanical Engineering*, 2018, 54(6): 191–202. doi: 10.3901/JME.2018.06.191.

[14] 谢志强, 张晓欢, 辛宇, 等. 考虑后续工序的择时综合调度算法[J]. 自动化学报, 2018, 44(2): 344–362. doi: 10.16383/j.aas.2018.c160562.

XIE Zhiqiang, ZHANG Xiaohuan, XIN Yu, *et al*. Time-selective integrated scheduling algorithm considering posterior processes[J]. *Acta Automatica Sinica*, 2018, 44(2): 344–362. doi: 10.16383/j.aas.2018.c160562.

[15] WANG Zhen, ZHANG Xiaohuan, and PENG Gang. An improved integrated scheduling algorithm with process sequence time-selective strategy[J]. *Complexity*, 2021, 2021: 5570575. doi: 10.1155/2021/5570575.

曹望成：男，1980年生，副教授，博士生，研究方向为企业智能计算与调度系统.

谢志强：男，1962年生，教授，研究方向为企业智能计算与调度系统.

裴莉榕：女，1994年生，博士生，研究方向为企业智能计算与调度系统.

责任编辑：陈 倩