运筹学与控制论

DOI :10. 3969/J. ISSN. 1672-6693. 2009. 03. 001

# 批处理机上有就绪和截止时间的等长度工件排序*

刘朝晖
(华东理工大学 数学系, 上海 200237)

摘要 :一台批处理机一次可以同时加工多个工件(称为一批),每批工件有相同的开工和完工时间,加工时间等于其中最长工件的加工时间。本文研究单台批处理机上有就绪时间和截止时间约束的 $n$ 个等长度工件的排序问题,目标是求一个可行时间表。就该问题,Baptiste 已经提出了一个复杂性为 $O(n^8)$ 的算法,在此基础上,本文推广 Garey 等人关于对应的经典排序问题的算法,得到了一个复杂性为 $O(n^2)$ 的算法。算法分两个阶段执行 :在阶级 I,算法找出所谓的禁止开工区间,在这些区间中将不允许有工件开工 ;在阶段 II,算法从时刻零开始,每当机器有空闲且不属于禁止开工区间的时候,就按照最早截止时间优先规则从已就绪的未加工工件中选择尽可能多的工件作为一批进行加工,若当前的机器空闲时刻属于某个禁止开工区间,则首先更新其到该禁止开工区间的右端点再进行决策。

关键词 :排序 ;批处理机 ;等长度工件 ;多项式时间算法
中图分类号 :O223 文献标识码 :A 文章编号 :1672-6693(2009)03-0001-04

一台批处理机一次可以同时加工多个工件(称为一批),每批工件有相同的开工和完工时间,并且开工后不允许中断,也不允许插入其他工件,加工时间等于其中最长工件的加工时间。批处理机排序模型源于半导体芯片制造中的预烧(burn-in)测试工艺,其过程是将芯片(工件)成批地放入烘箱(批处理机)中烘烤,能经受住预定烘烤时间(加工时间)的芯片为合格产品,Lee 等[1] 提供了相关的背景介绍。

本文研究下面的单台批处理机排序问题。有 $n$ 个工件 $J_1, J_2, \cdots, J_n$,加工时间皆为 $p$,工件 $J_i$($i = 1, 2, \cdots, n$) 的就绪时间为 $r_i$、截止时间为 $d_i$,即 $J_i$ 只能在时间区间 $[r_i, d_i]$ 内安排加工。设工件由一台可以同时加工至多 $b$ 个工件的批处理机进行加工,求一个可行时间表。

当工件有相同截止时间时,Ikura 和 Gimple[2] 证明了上述问题可以用动态规划方法在 $O(n^2)$ 时间内解决。对任意截止时间的情形,Baptiste[1] 证明了利用动态规划方法也可以在多项式时间内解决,但复杂性高达 $O(n^8)$。在 $b = 1$ 时,即对于对应的经典单机排序问题,Garey 等[6] 设计了一个复杂性为 $O(n^2)$(使用一定的数据结构,可降低为 $O(n\log n)$)的算法,该算法首先找出不允许工件在其中开工的所谓禁止开工区间,然后按照最早截止时间优先规则逐个安排已就绪的工件进行加工。本文将证明 Garey 等的算法可以推广到任意的 $b$,从而就一般情形给出了一个 $O(n^2)$ 算法,明显改进了 Baptiste 的结果。关于单台批处理机上等长度工件排序的其它结果参考 Webster 和 Baker[5]。当允许工件有不同加工时间时,上述问题是 NP 困难的,甚至在 $b = 1$ 和 $b = +\infty$ 的情形[6-7],或者在工件有相同截止时间的情形[8-9]。

在可行时间表不存在时,排序者可以考虑允许工件在截止时间之后完工,即允许延迟(此时通常称截止时间为工期或交货期),注意到 Baptiste[3] 已证明了最大延迟的值可以限定在基数为 $O(n^3)$ 的集合 $\{t_i + kp - d_j \mid i, j = 1, 2, \cdots, n, k = 0, 1, \cdots, n\}$ 内,因此通过二分搜索,调用笔者的算法可在 $O(n^2\log n)$ 时间内得到一个使得最大延误最小的时间表。

## 1 禁止开工区间和算法

称开区间 $(t_1, t_2)$ 为禁止开工区间,如果在任何

* 收稿日期 :2009-06-04
资助项目 :国家自然科学基金资助项目(No. 10771067)
作者简介 :刘朝晖,男,教授,博士生导师,研究方向为组合最优化。

可行时间表中不可能有工件在$(t_1,t_2)$内开工。先考虑问题:给定若干禁止开工区间,若不考虑工件各自的就绪时间和截止时间,如何安排$k$个加工时间为$p$的工件,使得它们全在时刻$d$之前完工。

该问题可以用下面的算法1求解,并且算法1给出这$k$个工件中最早开工工件的最迟开工时间。

算法1 1)任意划分$k$个工件成$B_1,B_2,\cdots,B_{\lceil k/b\rceil}$共$\lceil k/b\rceil$个批;

2)从后向前依次安排批$B_{\lceil k/b\rceil},B_{\lceil k/b\rceil-1},\cdots,B_1$,其中$B_i$的开工时间$s_i$为不落入禁止开工区间且小于等于$s_{i+1}-p$(或$d-p$,如果$i=\lceil k/b\rceil$)的最迟时刻。

引理1 设$s_1$按算法1生成,若$k$个工件全在时刻$s_1$之后开工且开工时间皆不落入禁止开工区间,则它们不可能全在时刻$d$之前完工。

证明 注意到$k$个工件至少应分成$\lceil k/b\rceil$批进行加工并参照Garey等的文献[4]中Lemma 3的证明,结论易得。 证毕

按以下方式运用算法1。考虑就绪时间$r_i$和截止时间$d_j(d_j\geqslant d_i)$,设已经给出了$[r_i,d_j]$中的禁止开工区间集合,则约束于这些禁止开工区间及$d=d_j$,运用算法1于工件集$I(i,j)=\{J_k\mid r_i\leqslant r_k\leqslant d_k\leqslant d_j\}$可以发现$I(i,j)$中最早开工工件的最迟开工时间$s$。若$s<r_i$,则$I(i,j)$中的工件不可能全在$[r_i,d_j]$中加工,因而原问题不可行;若$r_i\leqslant s<r_i+p$,则可以断定$(s-p,r_i)$是一个禁止开工区间,因为若其中有工件开工,则这些工件不属于$I(i,j)$且会迫使$I(i,j)$中所有工件的开工时间大于$s$。

在求解单台批处理机上有就绪时间和截止时间约束的等长度工件排序问题时,将从右向左确定禁止开工区间。考虑按从大到小的顺序处理就绪时间,在处理$r_i$时,$r_i$右边的禁止开工区间都已被发现,对于每个$d_j\geqslant d_i$,约束于$d=d_j$及这些禁止开工区间,把算法1运用于工件集$I(i,j)$以确定其中最早开工工件的最迟开工时间$c_j$,称$c_j$为$d_j$(相应于$r_i$)的关键时间。设$c$是相应于$r_i$的最小关键时间,则$c<r_i$时原问题不可行;$r_i\leqslant c<r_i+p$时,发现禁止开工区间$(c-p,r_i)$。

一旦发现了所有禁止开工区间,将从时刻零开始,在每个时刻按照最早截止时间优先规则安排尽可能多(但不超过$b$个)已就绪的未加工工件作为一批进行加工;若当前时刻落入某个禁止开工区间,则首先更新其到禁止开工区间的右端点。

算法2 (假设工件编号满足$r_1\leqslant r_2\leqslant\cdots\leqslant r_n$)

阶段I(发现禁止开工区间)

首先对$j=1,2,\cdots,n$,置$b_j=0$;并设禁止开工区间集合为空集。然后对$i=n,n-1,\cdots,1$重复下面的1)和2)。

1)对于每个$d_j\geqslant d_i$做①和②:

①若$b_j=0$,置$b_j=1,c_j=d_j-p$;若$0<b_j<b$,置$b_j=b_j+1$;若$b_j=b$,置$b_j=1,c_j=c_j-p$;

②若$c_j$落入某个已经发现的禁止开工区间$F$,则置$c_j=\inf(F)$。

2)若$i=1$或$r_i>r_{i-1}$,置$c=\min\{c_j\mid b_j\neq 0\}$,然后做:若$c<r_i$,宣称问题不可行,算法结束;若$r_i\leqslant c<r_i+p$,宣称$(c-p,r_i)$为禁止开工区间。

阶段II(生成可行时间表)

首先置$t=0$,然后重复下面的步骤1)、2)、3),直到所有工件安排完为止:

1)若在时刻$t$没有已就绪的未排序工件,则置$t$为下一个就绪时间;

2)若$t$落入某个禁止开工区间$F$,则置$t=\sup(F)$;

3)在时刻$t$按最早截止时间优先规则安排尽可能多(但不超过$b$个)已就绪的未加工工件作为一批进行加工,然后置$t=t+p$。

算法2在计算$d_j$(相应于下一个就绪时间)的关键时间时采用了更新原有关键时间的方式,这使得整个算法能在$O(n^2)$时间内运行。

例1 设$b=3,n=6,p=3,r_i$、$d_i$如表1。

表1 $r_i$、$d_i$相关数据

| | $J_1$ | $J_2$ | $J_3$ | $J_4$ | $J_5$ | $J_6$ |
|---|---|---|---|---|---|---|
| $r_i$ | 0 | 1 | 2 | 3 | 4 | 5 |
| $d_i$ | 5 | 7 | 12 | 10 | 11 | 8 |

对例1运用算法2,计算过程中的关键时间如表2所示。当处理$r_6=5$时,$d_6=8$的关键时间5最小,由于$r_6\leqslant 5<r_6+p$,所以宣称(2,5)为禁止开工区间。当处理$r_2=1$时,由于(2,5)是禁止开工区间,所以$d_2=7$的关键时间为2,并且由于$r_2\leqslant 2<r_2+p$,所以宣称(-1,1)为禁止开工区间。最后生成时间表如图1。

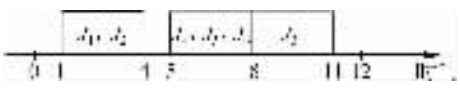

图1 生成时间

表 2　例 1 中 $d_j$ 相应于 $r_j$ 的关键时间

| $d_j$ \ $r_i$ | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|
| 12 | 9 | 9 | 9 | 6 | 6 | 6 |
| 11 | 8 | 8 | 8 |  | 5 | 5 |
| 10 | 7 |  | 7 |  | 7 | 2 |
| 8 | 5 |  |  |  | 5 | 5 |
| 7 |  |  |  |  | 2 | 2 |
| 5 |  |  |  |  |  | 2 |

## 2 算法的正确性证明

**定理 1**　在任何可行时间表中，没有工件在算法 2 发现的禁止开工区间中开工；若算法 2 宣称问题不可行，则该问题必不可行。

**证明**　利用引理 1，并对禁止开工区间的个数进行归纳易证结论成立。　　证毕

**定理 2**　若算法 2 没有宣称问题不可行，则生成一个可行时间表。

**证明**　用反证法。设对于某问题，算法 2 没有宣称不可行，但生成的时间表 $S=(B_1,B_2,\cdots,B_j,\cdots)$ 不可行，其中 $B_j$ 是第一个包含延误工件的批。设 $s(B_j)$、$r(B_j)$、$d(B_j)$ 分别表示 $B_j$ 的开工时间、$B_j$ 中工件的最早就绪时间、$B_j$ 中工件的最小截止时间。

不失一般性，设 $[0,s(B_j)]$ 中所有机器闲置时间属于禁止开工区间，若不然，令 $t'$ 是其中不属于禁止开工区间的最后一段机器闲置时间的右端点，并设 $r$ 是在 $t'$ 或之后开工的工件的最早就绪时间，依据 $S$ 的生成方法，应有 $r\geqslant t'$，所以 $[r,s(B_j)]$ 中没有不属于禁止开工区间的机器闲置时间，而且所有在 $t'$ 之前完工的工件的就绪时间小于 $r-p$，这些工件对于确定 $r$ 之后的禁止开工区间没有作用，因此删去所有在 $r$ 之前完工的工件，并将余下工件的就绪时间和截止时间减小 $r$ 可获得一个新的反例，对于该新的反例而言，$[0,s(B_j)]$ 中的所有机器闲置时间属于禁止开工区间。以下分几种情形讨论。

情形 1　$|B_{j-1}|<b$。依据算法 2 生成 $S$ 的方法，应有 $r(B_j)>s(B_{j-1})$；由于 $B_j$ 包含延误工件，所以 $d(B_j)-p<s(B_j)$；此外，注意到包含 $r(B_j)$ 及 $r(B_j)$ 之后的禁止开工区间皆在处理 $r(B_j)$ 之前已经发现，易证以下结论：

1）若 $r(B_j)\geqslant s(B_{j-1})+p$，则 $d(B_j)$ 相应于 $r(B_j)$ 的关键时间小于 $r(B_j)$，算法 2 将宣称问题不可行，与定理假设矛盾！

2）若 $r(B_j)<s(B_{j-1})+p$，则 $d(B_j)$ 相应于 $r(B_j)$ 的关键时间 $c<s(B_{j-1})+p$，算法 2 或者宣称问题不可行（若 $c<r(B_j)$），或者宣称包含 $s(B_{j-1})$ 的禁止开工区间 $(c-p,r(B_j))$（若 $r(B_j)\leqslant c<s(B_{j-1})+p$），前一种情形与定理假设矛盾，后一种情形与按算法 2 生成的时间表 $S$ 不会有工件的开工时间落入禁止开工区间矛盾！

情形 2　$|B_{j-1}|=b$。设 $i$ 是使得 $|B_i|=|B_{i+1}|=\cdots=|B_{j-1}|=b$ 的最小指标。

情形 2.1　$B_i\cup B_{i+1}\cup\cdots\cup B_{j-1}$ 中有工件的截止时间大于 $d(B_j)$，设 $B_k$ 是包含这种工件的最迟开工的批，$r$ 是 $B_{k+1},B_{k+2},\cdots,B_j$ 中截止时间小于等于 $d(B_j)$ 的工件的最早就绪时间，显然 $r>s(B_k)$。

约束于 $d=d(B_j)$，对 $B_{k+1},B_{k+2},\cdots,B_{j-1}$ 中的工件及 $B_j$ 中截止时间等于 $d(B_j)$ 的工件使用算法 1，将生成 $j-k$ 个开工时间 $s_{k+1},s_{k+2},\cdots,s_j$，现在递推地证明 $s(B_{j-l})>s_{j-l}(0\leqslant l\leqslant j-k-1)$。由于 $B_j$ 包含延误工件，所以 $s(B_j)>s_j$。如果已证得 $s(B_j)>s_j$，$s(B_{j-1})>s_{j-1},\cdots,s(B_{j-l})>s_{j-l}(0\leqslant l\leqslant j-k-2)$，现证 $s(B_{j-l-1})>s_{j-l-1}$。

若 $s(B_{j-l})=s(B_{j-l-1})+p$，则由 $s(B_{j-1})>s_{j-l}$ 可推出 $s(B_{j-l-1})>s_{j-l-1}$；若 $s(B_{j-1})>s(B_{j-l-1})+p$，则 $[s(B_{j-l-1})+p,s(B_{j-l}))$ 属于禁止开工区间，所以算法 1 生成的 $s_{j-l}<s(B_{j-l-1})+p$，因而

$$s(B_{j-l-1})>s_{j-l}-p\geqslant s_{j-l-1}$$

由于已证得 $s(B_{k+1})>s_{k+1}$，所以算法 2 在处理就绪时间 $r$ 时，相应于 $r$ 的最小关键时间 $c<s(B_{k+1})$，进而，由于 $B_{k+1}$ 在 $[s(B_k)+p,s(B_j)]$ 中不属于禁止开工区间的最早时刻开工且 $c$ 也不落入禁止开工区间，所以 $c<s(B_k)+p$。如果 $c<r$，算法 2 将宣称问题不可行；如果 $c\geqslant r$，则 $s(B_k)<r\leqslant c<s(B_k)+p<r+p$，算法 2 将宣称包含 $s(B_k)$ 的禁止开工区间 $(c-p,r)$，矛盾！

情形 2.2　$B_i\cup B_{i+1}\cup\cdots\cup B_{j-1}$ 中工件的截止时间皆小于等于 $d(B_j)$。如果 $i>1$，类似于情形 2.1 可证明结论，以下考虑 $i=1$ 的情形。

设 $r$ 是 $B_1,B_2,\cdots,B_j$ 中截止时间小于等于 $d(B_j)$ 的工件的最早就绪时间，类似于情形 2.1，约束于 $d=d(B_j)$，算法 1 将指派给 $B_1$ 一个比 $s(B_1)$ 更小的开工时间 $s_1$，既然按算法 2，$B_1$ 开工尽可能早（除非受禁止开工区间影响），因此 $s_1<r$，即处理就绪时间 $r$

时将发现小于 $r$ 的关键时间,算法2将宣称问题不可行,矛盾!

情形3 $j = 1$。类似于情形2.2可证明算法2将宣称问题不可行。 证毕

## 参考文献:


[1] Lee C Y,Uzsoy R,Martin-Vega L A. Efficient algorithms for scheduling semiconductor burn-in operations[J]. Oper Res, 1992,40:764-775.

[2] Ikura Y,Gimple M. Efficient scheduling algorithms for a single batch processing machine[J]. Oper Res Lett,1986, 5:61-65.

[3] Baptiste P. Batching identical jobs[J]. Math Meth Oper Res, 2002,52:355-367.

[4] Garey M R,Johnson D S,Simons B B,et al. Scheduling unit-time tasks with arbitrary release times and deadlines[J]. SIAM J Comput,1981,10:256-269.

[5] Webster S,Baker K R. Scheduling groups of jobs on a single machine[J]. Oper Res,1995,43:692-703.

[6] Cheng T C E,Liu Z,Yu W. Scheduling jobs with release dates and deadlines on a batch processing machine[J]. IIE Trans,2001,33:685-690.

[7] Lenstra J K,Rinnooy Kan A H G,Brucker P. Complexity of machine scheduling problems[J]. Ann Discrete Math,1977 (1):343-362.

[8] Liu Z,Yu W. Scheduling one batch processor subject to job release dates[J]. Discrete Appl Math,2000,105:129-136.

[9] Brucker P,Gladky A,Hoogeveen H,et al. Scheduling a batching machine[J]. J Sched,1998(1):31-54.


## Scheduling Equal-length Jobs with Release Times and Deadlines on a Batch Machine


*LIU Zhao-hui*

(Dept. of Mathematics, East China University of Science and Technology, Shanghai 200237, China)



**Abstract**: A batch machine can process a number of jobs simultaneously as a batch, where all the jobs processed in a batch have the same start time and the same completion time, and the processing time of a batch is given by the longest processing time of the jobs assigned to it. This paper studies the problem of scheduling n equal-length jobs with release times and deadlines on a batch machine, so as to obtain a feasible schedule. Baptiste has presented an $O(n^8)$ algorithm for the problem. Comparatively, Garey and others have given an $O(n^2)$ algorithm for the corresponding classical single machine scheduling problem. In this paper, we generalize Garey and others' algorithm to obtain an $O(n^2)$ algorithm for the above batch machine scheduling problem, which improves greatly on Baptiste's algorithm. Our algorithm is divided into two phases. In the first phase, we declare the so-called forbidden regions in which no jobs can start if the schedule is to be feasible. In the second phase, we schedule the jobs forward from time zero by using the earliest deadline scheduling rule. If the machine is idle and some unscheduled jobs are available at some time which is not in any forbidden region, we schedule the available unscheduled jobs with earlier deadlines as many as possible as a batch. If the idle time of the machine falls in some forbidden region, we first update the time to the right end of the forbidden region, and then make decision.



**Key words**: scheduling; batch machine; equal-length job; polynomial time algorithm

(责任编辑 黄 颖)