第21卷 第11期
Vol 21 No. 11

控 制 与 决 策
Control and Decision

2006年11月
Nov. 2006

文章编号: 1001-0920(2006)11-1293-05

单台批处理机总加权完成时间最小化的启发式算法

冯大光[1,2], 唐立新[1]
(1. 东北大学 物流优化与控制研究所, 沈阳 110004; 2. 沈阳农业大学 基础部, 沈阳 110161)

摘　要: 批处理机总加权完成时间最小化问题的复杂性目前还没有确定,因此有必要研究该问题的启发式算法. 基于对该问题最优解性质的分析,提出了工件分批的最优性质. 分别基于WSPT规则和SPT规则对工件进行总排序,利用工件最优分批性质进行分批,提出了两种启发式算法(简称为WSPTS和SPTS). 为了检验算法的性能,将提出的算法与此问题的基准算法和常规算法进行了比较,结果表明,启发式算法WSPTS要优于其他的算法,而SPTS算法的性能最优.

关键词: 批处理机; WSPT规则; SPT规则; 动态规划; 启发式算法
中图分类号: TP278　　**文献标识码**: A

Heuristic Algorithms for Single Batching Machine with Total Weighted Completion Time

FENG Da-guang[1,2], *TANG Li-xin*[1]
(1. The Logistics Institute, Northeastern University, Shenyang 110004, China; 2. Department of Basic Science, Shenyang Agricultual University, Shenyang 110161, China. Correspondent: TANG Li-xin, E-mail: qhjytlx@mail.neu.edu.cn)

Abstract: The problem of *n* jobs to be processed on single batching machine with a capacity to minimize the total weighted completion time is discussed. An analysis and proof are given for the quality of optimal solution under some condition, based on which two heuristic algorithms are carried out, selecting jobs ordered by WSPT rule (WSPTS) and selecting jobs ordered by SPT rule (SPTS). To compare the proposed algorithms, heuristic dynamic programming based on WSPT and SPT respectively is carried out, and so does full batch algorithm. The experiment result shows that the heuristic WSPTS is the best one and the SPTS is the most stable one among all the algorithms.

Key words: Batching machine; WSPT rule; SPT rule; Dynamic programming; Heuristic algorithm

1 引　　言

在实际生产过程中有的机器可以同时加工多个工件(如大规模集成电路生产中最后的检验阶段[1]),将单台机器可以同时加工多个工件的排序问题,称为批处理机调度问题. 许多其他生产过程,如电镀、钢铁生产过程中的加热炉等都存在批处理机调度问题,因而具有广泛的应用价值和现实意义.

Potts[2]等人对批处理机的调度问题做了综述. French[3]的WSPT规则被应用于批的调度问题,提出了BWSPT规则[4]. Liu[5]等对能力无限的批处理机问题进行了研究. Cheng[6]等对于工件具有优先权和到达期约束的批处理机调度问题进行了研究. Zhang[7]等给出了工件动态到达批处理机的makespan最小化的在线算法. Li[8]等对工件具有释放时间和尺寸的批处理机的makespan最小化问题进行了研究.

本文对单台批处理机总加权完成时间最小化问题部分工件的分批性质进行了研究,提出了启发式

收稿日期: 2005-09-07; **修回日期**: 2006-01-12.
基金项目: 国家杰出青年科学基金项目(70425003); 国家自然科学基金项目(70171030, 60274049); 高等学校优秀青年教师教学科研奖励计划项目(教育司[2002]383).
作者简介: 冯大光(1972—),男,辽宁锦州人,讲师,博士生,从事基于VLSN的智能优化算法和生产物流调度的研究; 唐立新(1966—),男,黑龙江绥化人,教授,博士生导师,从事制造业物流和供应链建模与优化等研究.

算法, 并通过数字实验与常规算法进行了对比.

2 模 型

单台批处理机调度问题的模型为: n 个工件要在一台能力为 B 批处理机上进行加工, 每个工件在开始时刻就可以进行加工, 加工时间和权值分别为 p_i 和 w_i. 批处理机每次至多可以加工 B 个工件, 称 B 为批处理机的能力, 这些工件可以分成 r 个批, 记为 $B_1, B_2, \cdots, B_r$, 批的加工时间和权值分别记为 $p(B_i)$ 和 $w(B_i)$, $i = 1, 2, \cdots, r$. 第 j 批的加工时间 $p(B_j) = \max_{i \in B_j}\{p_i\}$, 并且权值为 $w(B_j) = \sum_{i \in B_j} w_i$, 所含有的工件数记为 $|B_j|$. 由于同一批的工件同时加工, 所以同一批的工件具有相同的完成时间. 单台批处理机调度问题包括如何把工件分批和确定批的加工顺序, 从而优化某个目标函数.

本文所研究的单台批处理机调度问题的目标函数为所有工件的加权完成时间最小化 $\sum_{i=1}^{n} w_i C_i$, 其中 C_i 为工件 i 的完成时间, 也可以表示成 $\sum_{j=1}^{r} w(B_j) \sum_{i=1}^{j} p(B_i)$, 即 $\sum_{j=1}^{r} \sum_{l \in B_j} w_l \sum_{i=1}^{j} p(B_i)$.

3 最优解的一些性质

为了更加详尽地研究能力有限的单台批处理机的调度问题, 下面介绍一些定义[2]. 对于批 B_j, 如果它所含有的工件的个数等于机器的容量 B, 即 $|B_j| = B$, 则称该批为满批; 否则, 称为非满批. 在批的排序中, 如果存在两个批 B_j 和 B_k, 批 B_j 在批 B_k 的前面, 且 $p(B_j) > p(B_k)$, 则称批 B_k 相对于批 B_j 是滞后的. 下面给出目标函数是总加权完成时间最小化的调度问题的一些性质.

对于给定批的工件的总加权完成时间最小化问题, WSPT 规则[3] 给出了最优的批的排序.

性质 1 (BWSPT 规则)[4] 对于给定的批 B_i, ($i = 1, 2, \cdots, r$), 加工时间和权值分别为 $p(B_i)$ 和 $w(B_i)$, 则批按 $p(B_i)/w(B_i)$ 的非减次序排列, 可以得到给定批的总加权完成时间最小化的最优次序.

性质 2 在最优的批调度中, 如果有两个工件 i 和 j 的加工时间相同, i 的权值小于 j 的权值, 且工件 i 在第 k 批 B_k 中, 则工件 j 或者在批 B_k 中或者在 B_k 前面的批中.

引理 1 在一个最优的批的调度中不能存在相对于不满批的滞后批.

由引理 1, 可以得到下面的结论:

引理 2 在一个最优的批调度中, 如果加工时间最大的工件所在的批是非满批, 则该批一定排在最后.

引理 3 在一个最优的批的调度中如果存在不满批, 则该批后的任意一个批中的工件的加工时间都大于该批的加工时间.

定理 1 当批处理机的容量为 B 时, 按 SPT 规则排序后的工件为 $1, 2, \cdots, B$, 如果满足 $p_i < \left(\sum_{l=1}^{i} w_l \Big/ \sum_{l=1}^{i-1} w_l\right) p_{i-1}$, 对一切 $2 \le i \le B$ 都成立, 则工件 $1, 2, \cdots, B$ 作为一批是最优的, 即总加权完成时间最小.

定理 2 当批处理机的容量为 B 时, 对于按任意次序排列的工件 $1, 2, \cdots, B$, 如果 $\max_{1 \le m \le i}\{p_m\} < \left(\sum_{l=1}^{i} w_l \Big/ \sum_{l=1}^{i-1} w_l\right) \max_{1 \le m \le i-1}\{p_m\}$, 对一切 $2 \le i \le B$ 都成立, 则工件 $1, 2, \cdots, B$ 作为一批是最优的, 即总加权完成时间最小.

4 启发式算法

由于 $1|B|\sum_{i=1}^{n} w_i C_i$ 的 NP 性到目前为止没有确定, 因而研究其启发式算法是十分有意义的.

首先, 根据工业生产中常采用的方法给出满批算法.

算法 1 SPT 满批算法 (SPTFB 算法)[3]:

Step 1: 把工件按加工时间的非减次序编号, 对于加工时间相同的工件按权值的非增顺序编号.

Step 2: 把编号从 $(i-1)B + 1$ 到 $\min\{n, iB\}$ $\left(i = 1, \cdots, \left[\frac{n}{B}\right]\right)$ 的工件放入同一批中;

Step 3: 把生成的批按 BWSPT 次序在机器上加工.

把算法 1 中的工件按 WSPT 规则排序, 其他分批方法相同则得到:

算法 2 WSPT 满批算法 (WSPTFB 算法). 满批算法的时间复杂性为 $O(n)$.

利用定理 2, 得到下面的选择分批算法. 介绍几个定义:

当前工件序列: 按某个规则排列的去掉已经分完批的工件的集合;

当前批: 当前正在添加工件的批;

当前工件 (cjob): 当前正被分批的工件;

$p(B)$: 表示当前批的加工时间;

$w(B)$: 表示当前批的权值;

$p(\text{cjob})$: 表示当前工件的加工时间;

$w(\text{cjob})$: 表示当前工件的权值.

算法 3 WSPT 选择分批算法 (WSPTS):

Step 1: 工件按 WSPT 规则排序, 得到当前工件序列.

Step2: 开始新的一批, 即当前批, 把当前工件序列中的第一个工件作为当前批的第一个工件, 在当前工件序列中去掉该工件, $p(B) = p(\text{cjob})$, $w(B) = w(\text{cjob})$.

Step3: 把当前工件序列中加工时间小于当前批加工时间且权值最大的工件称为当前工件, 放入当前批, $w(B) = w(\text{cjob}) + w(B)$, 在当前工件序列中去掉该工件. 如果还存在这样的工件, 继续放入当前批, 权值相应的变化, 并在当前工件序列中去掉相应的工件, 如果构成满批, 转 Step2; 如果不存在这样的工件, 转 Step4; 如果所有工件分完批, 转 Step5.

Step4: 如果所有的工件都分完批, 转 Step5. 把当前工件序列中的第一个工件作为当前工件, 如果 $p(\text{cjob}) < [(w(B) + w(\text{cjob}))/w(B)] \times p(B)$, 把该工件放入当前批, 如果直至最后一个工件都不满足该式子, 转 Step2; 如果构成满批转 Step2; 如果没有构成满批转 Step3.

Step5: 把分好批的工件按BWSPT 规则进行加工.

当工件按 SPT 顺序排列时, 只需要考虑工件的权值和加工时间是否满足定理 1 即可, 从而得到:

算法 4　SPT 选择分批算法(SPTS), 算法的具体步骤略去.

定理 3　选择分批算法的计算复杂度介于 $O(n^2)$ 和 $O\left(n\left(1 - \frac{1}{B}\right)\right)$ 之间.

证明　假设 n 个工件为 $1, 2, \cdots, n$, 机器的处理能力为 B, K 为不大于 n/B 的最大正整数.

1) 选择分批算法的最差情况为所有的工件在任何情况下都不满足判定条件, 具体分析如下:

① 选择工件 1 开始新的一批, 工件 $2, 3, \cdots, n$ 都不满足条件, 共判断了 $n - 1$ 次, 工件 1 自己作为一批;

② 选择工件 i 开始新的一批, 工件 $i + 1, i + 2, \cdots, n$ 都不满足条件, 共判断了 $n - i$ 次, 工件 i 自己作为一批 ($i = 2, 3, \cdots, n - 1$);

③ 工件 n 自己作为一批, 判断 0 次;

从而总的判断次数为

$$(n - 1) + (n - 2) + \cdots + 1 = \frac{(n - 1)n}{2},$$

所以最差情况的计算复杂性为 $O(n^2)$.

2) 选择分批算法的最好情况是所有的工件恰好按照其排列顺序满足需要判断的条件, 具体分析如下:

① 选择工件 1 开始新的一批, 工件 $2, 3, \cdots, B$ 都满足条件, 共判断了 $B - 1$ 次, 工件 $1, 2, 3, \cdots, B$ 作为一批;

② 选择工件 $iB + 1$ 开始新的一批, 工件 $iB + 2, \cdots, (i + 1)B$ 都满足条件, 共判断了 $B - 1$ 次, 工件 $iB, iB + 1, iB + 2, \cdots, (i + 1)B$ 作为一批($i = 2, 3, \cdots, K - 1$);

③ 选择工件 $KB + 1$(如果存在) 开始新的一批, 工件 $KB + 2, \cdots, n$ 都满足条件, 共判断了 $n - (KB + 1)$ 次.

从而总的判断次数为: n 能被 B 整除时, 有

$$K(B - 1) = n - \frac{n}{B} = n\left(1 - \frac{1}{B}\right);$$

n 不能被 B 整除时, 有

$$K(B - 1) + n - (KB + 1) = n\left(1 - \frac{1}{B}\right) - 1.$$

从而最好情况的计算复杂性为 $O\left(n\left(1 - \frac{1}{B}\right)\right)$. 综上可得定理成立. □

用动态规划算法解决总加权完成时间最小化问题时, 仿照 Brucker[9] 的动态规划算法, 工件按 WSPT 规则排序, 得到如下的动态规划算法 (WSPTHD), 迭代方程为

$$G_{n+k} = 0,\ p_{n+k} = 0,\ w_{n+k} = 0;\quad k = 1, 2, \cdots, B;$$

$$G_j = \min_{j \le k \le j+B}\left\{G_{k+1} + \max_{j \le m \le k}\{p_m\}\sum_{h=j}^{n} w_h\right\},\quad j = 1, 2, \cdots, n.$$

G_1 即为最优解.

5　实验研究

产生测试问题时, 按照 Uzsoy[4] 产生测试问题的方法, 权值和加工时间分别在 [1, 10] 和 [1, 20] [10, 20] 上服从均匀分布, 随机产生, 工件数为 10, 20, …, 100, 每种情况产生 10 个例子, 机器的能力为 3 和 5, 一共产生 1 800 个测试问题. 算法的程序用 C 语言编写, 在 AMD Athlon (tm) XP 3000 +, 2.16GHz DDR 256M 的机器上运行. 各种算法的归一化实验结果如表 1、表 2 所示, 其中 SPTFB 表示基于 SPT 规则的满批算法; WSPTFB 表示基于 WSPT 规则的满批算法; SPTHD 表示基于 SPT 规则的动态规划算法; WSPTHD 表示基于 WSPT 规则的动态规划算法; SPTS 表示基于 SPT 规则的选择分批算法; WSPTS 表示基于 WSPT 规则的选择分批算法. 本文把 Brucker[9] 的算法 SPTHD 作为基准算法.

从表 1 可以看到, 对于满批算法和动态规划算法而言, 工件按 SPT 规则排序显然要优于工件按 WSPT 规则排序; 对于选择分批算法工件按 WSPT 规则排序要明显强于工件按 SPT 规则排序, 并且选择分批算法要优于其他各种算法. 从统计学的角度

来看, 基于SPT 规则的算法要比基于W SPT 规则的算法稳定, 虽然从平均值来看,W SPTS 算法优于SPTS 算法, 但SPTS 的稳定性要明显超过W SPTS 算法, 并且它是所有这些算法中性能最稳定的

从表2 可得结论: 1) 当权值在[10, 20]内产生时, 不论加工时间在哪个范围, 基于SPT 规则的选

表1　目标函数值作为工件数的函数的平均归一化对比结果

	工件数量	SPTHD	W SPTHD	SPTFB	W SPTFB	SPTS	W SPTS
	10	1.048 3	1.021 2	1.070 7	1.066 4	1.051 5	1.029 2
	20	1.047 6	1.057 3	1.045 8	1.074 4	1.038 8	1.018 6
	30	1.064 1	1.069 9	1.043 6	1.088 8	1.043 2	1.009 7
	40	1.067 1	1.085 5	1.043 1	1.097 3	1.040 9	1.008 5
	50	1.079 8	1.096 1	1.044 2	1.108 1	1.043 0	1.004 2
	60	1.072 1	1.098 5	1.036 5	1.110 6	1.035 8	1.004 6
$B = 3$	70	1.075 5	1.104 7	1.037 2	1.116 1	1.036 3	1.003 5
	80	1.081 6	1.108 8	1.038 6	1.121 1	1.038 3	1.003 3
	90	1.076 8	1.111 4	1.032 6	1.123 9	1.031 5	1.004 2
	100	1.077 7	1.111 0	1.029 6	1.121 3	1.028 9	1.003 1
	平均值	1.069 1	1.086 4	1.042 2	1.102 8	1.038 8	1.008 9
	标准差	0.000 1	0.000 8	0.000 1	0.000 4	0.000 0	0.000 1
	改进率		-1.730 0	2.690 0	-3.370 0	3.030 0	6.020 0
	10	1.027 1	1.034 4	1.072 0	1.111 8	1.044 4	1.047 3
	20	1.028 6	1.045 1	1.034 0	1.083 2	1.028 0	1.033 0
	30	1.037 8	1.069 8	1.030 6	1.094 0	1.037 4	1.028 5
	40	1.050 6	1.084 5	1.040 1	1.101 2	1.041 0	1.011 2
	50	1.060 9	1.100 4	1.046 0	1.116 8	1.041 6	1.005 6
	60	1.075 1	1.114 2	1.050 2	1.133 5	1.052 1	1.006 2
$B = 5$	70	1.083 0	1.130 1	1.054 0	1.141 6	1.053 1	1.004 2
	80	1.090 3	1.136 2	1.056 3	1.148 5	1.051 6	1.004 2
	90	1.087 7	1.144 7	1.050 7	1.156 5	1.047 2	1.002 5
	100	1.094 3	1.145 3	1.050 9	1.156 6	1.048 5	1.003 2
	平均值	1.063 5	1.100 5	1.048 5	1.124 4	1.044 5	1.014 6
	标准差	0.000 6	0.001 5	0.000 1	0.000 6	0.000 1	0.000 2
	改进率		-3.700 0	1.500 0	-6.090 0	1.900 0	4.890 0

表2　按工件参数的分布归一化目标函数值

	加工时间范围	权值范围	SPTHD	W SPTHD	SPTFB	W SPTFB	SPTS	W SPTS
	[1, 10]	[1, 10]	1.048 7	1.028 2	1.070 2	1.071 5	1.053 0	1.027 5
		[1, 20]	1.057 3	1.063 6	1.048 1	1.082 7	1.042 4	1.016 7
		[10, 20]	1.059 6	1.072 2	1.040 6	1.089 8	1.037 6	1.010 7
	[1, 20]	[1, 10]	1.068 2	1.092 9	1.041 0	1.104 4	1.040 9	1.006 1
$B = 3$		[1, 20]	1.078 8	1.097 3	1.040 5	1.110 4	1.038 7	1.004 2
		[10, 20]	1.067 5	1.103 7	1.034 9	1.114 8	1.034 5	1.004 9
	[10, 20]	[1, 10]	1.082 3	1.103 6	1.040 2	1.114 5	1.040 0	1.002 6
		[1, 20]	1.086 6	1.108 7	1.037 4	1.120 5	1.036 4	1.003 7
		[10, 20]	1.072 6	1.107 7	1.026 7	1.116 7	1.026 1	1.003 5
	[1, 10]	[1, 10]	1.028 3	1.037 2	1.069 4	1.111 7	1.045 5	1.048 1
		[1, 20]	1.028 6	1.053 8	1.032 2	1.089 9	1.029 9	1.029 2
		[10, 20]	1.041 6	1.075 0	1.034 4	1.095 2	1.037 8	1.022 4
	[1, 20]	[1, 10]	1.051 2	1.093 9	1.041 0	1.114 2	1.037 2	1.009 5
$B = 5$		[1, 20]	1.070 0	1.107 1	1.046 7	1.124 9	1.048 6	1.006 2
		[10, 20]	1.073 9	1.125 7	1.050 9	1.140 3	1.049 2	1.006 9
	[10, 20]	[1, 10]	1.094 6	1.130 7	1.061 1	1.141 9	1.056 9	1.002 8
		[1, 20]	1.097 3	1.140 0	1.054 5	1.150 3	1.051 4	1.002 9
		[10, 20]	1.086 2	1.140 8	1.046 1	1.151 0	1.043 9	1.003 3

择算法是较优的; 基于SPT 规则的满批算法的性能也较好; 2) 对于满批算法而言, SPTFB 算法要优于W SPTFB 算法, 这与前面的结论是一致的; 3) 在一般情况下, 基于W SPT 规则的选择算法W SPTS 是性能最强的

6　结　　语

实验表明, 根据最优分批性质提出的选择分批算法W SPTS 是各种算法中最有效的, 但该算法相对基于SPT 规则的选择分批算法而言, 稳定性较差 另外, 选择分批算法要优于常规算法 实验结果还表明, 工件的初始次序对算法的结果是有影响的, 这提示未来的研究方向应该把算法和邻域搜索结合起来, 这是一个极有潜力的研究方向

参考文献(References)

[1] Chandreu V, Lee C Y, Uzsoy R. Minimizing Total Completion Time on Batch Processing Machines[J]. *Int J of Production Research*, 1993, 31(9): 2097-2121.

[2] Potts C N, Mikhail Y Kovalyov. Scheduling with Batching: A Review [J]. *European J of Operational Research*, 2000, 120(2): 228-249.

[3] French S. *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-shop* [M]. New York: John Wiley, 1982.

[4] Uzsoy R, Yang Y Y. Minimizing Total Weighted Completion Time on a Single Batch Processing Machine [J]. *Production and Operations Management*, 1997, 6(1): 57-73.

[5] Liu Z H, Yuan J J, Edwin Cheng T C. On Scheduling an Unbounded Batch Machine[J]. *Operations Research Letters*, 2003, 31(1): 42-48.

[6] Cheng T C E, Yuan J J, Yang A F. Scheduling a Batch-processing Machine Subject to Precedence Constraints, Release Dates and Identical Processing Times[J]. *Computers and Operations Research*, 2005, 32(4): 849-859.

[7] Zhang G C, Cai X Q, Lee C-Y, et al. Mimizing Makespan on a Single Batch Processing Machine with Nonidentical Job Sizes[J]. *Naval Research Logistics*, 2001, 48(3): 226-240.

[8] Li S G, Li G J, Wang X L, et al. Minimizing Makespan on a Single Batching Machine with Release Times and Non-identical Job Sizes [J]. *Operations Research Letters*, 2005, 33(2): 157-164.

[9] Brucker P, Gladky A, Hoogeveen H, et al. Scheduling a Batching Machine[J]. *J of Scheduling*, 1998, 1(1): 31-54.

(上接第1292 页)

5　结　　语

本文针对一类带有未知惯性参数、未建模动态及外界干扰的非完整动力学系统的鲁棒镇定问题, 基于滑模控制思想及非完整运动学系统的镇定策略, 给出了该类系统的鲁棒镇定方法; 并将其用于一类不确定非完整移动机械臂的鲁棒镇定分析, 仿真结果验证了所提出控制方法的正确有效性

参考文献(References)

[1] Brockett R W. *Asymptotic Stability and Feedback Stabilization* [A]. Differential Geometric Control Theory[C]. Boston: Birkhauser, 1983: 181-191.

[2] Komanovsky H, MacClamroch N H. Developments in Nonholonomic Control Systems [J]. *IEEE Control Systems Magazine*, 1995, 15(6): 20-36.

[3] 吴卫国, 陈辉堂, 王月娟 移动机器人的全局轨迹跟踪控制[J]. *自动化学报*, 2001, 27(3): 326-331.
(Wu W G, Chen H T, Wang Y J. Global Trajectory Tracking Control of Mobile Robots [J]. *Acta Automation Sinica*, 2001, 27(3): 326-331.)

[4] 李胜, 马国梁, 胡维礼 一类不确定非完整移动机器人的时变自适应镇定[J]. *机器人*, 2005, 27(1): 10-13.
(Li S, Ma G L, Hu W L. Time-varying Adaptive Stabilization of an Uncertain Nonholonomic Mobile Robot[J]. *Robot*, 2005, 27(1): 10-13.)

[5] 王朝立, 霍伟 用滑动模态实现一类非完整动力学系统的指数镇定[J]. *自动化学报*, 2000, 26(2): 254-257.
(Wang C L, Huo W. Exponential Stabilization of a Nonholonomic Dynamic System via Sliding Modes[J]. *Acta Automation Sinica*, 2000, 26(2): 254-257.)

[6] Lewis F, Abdallah C, Dawson D. *Control of Robot Manipulators*[M]. New York: Macmillan, 1993.

[7] Hassan K Khalil. *Nonlinear System* [M]. New York, 1992.

[8] Wu Y X, Hu Y M. Kinematics, Dynamics and Motion Planning of Wheeled Mobile Manipulators[A]. *Proc of Int Conf on CSIMTA'04* [C]. Cherbourg, 2004: 221-226.