第 29 卷 第 6 期
2003 年 11 月

自 动 化 学 报
ACTA AUTOMATICA SINICA

Vol. 29, No. 6
Nov., 2003

工件从大到小到达的带处理器费用的半在线调度算法[1)]

蔡圣义[1,2] 何 勇[1]

[1](浙江大学数学系 杭州 310027)

[2](温州师范学院数学系 温州 325000)

(E-mail: mathhey@zju.edu.cn)

摘 要 对大多数调度问题来说,处理器集往往是事先给定的,而且在算法进行过程中,它是不变的. Imreh 和 Noga 第一次提出了在调度中考虑处理器有费用的模型. 他们研究了所谓的 List Model 问题,给出了竞争比为 1.618 的在线算法,同时证明了任意在线算法的竞争比至少是 4/3. 该文研究 List Model 问题的一个半在线情形,即假设工件是从大到小到达的,给出一个竞争比为 3/2 的半在线算法. 同时证明对该问题的这一半在线情形,任意半在线算法的竞争比至少是 4/3.

关键词 半在线,算法, 竞争比, 调度,处理器费用

中图分类号

Quasi-Online Algorithm for Scheduling Non-Increasing Processing-Time Jobs with Processor Cost

CAI Sheng-Yi[1,2] HE Yong[1]

[1](*Department of Mathematics, Zhejiang University, Hangzhou* 310027)

[2](*Department of Mathematics, Wenzhou Teachers College, Wenzhou* 325000)

(E-mail: mathhey@zju.edu.cn)

Abstract For most scheduling problems the set of processors is fixed initially and remains unchanged for the duration of the problem. Imreh and Noga recently proposed adding the concept of processor cost to the scheduling problem and considered the so-called List Model problem. An online algorithm of a competitive ratio 1.618 was given with a lower bound of 4/3. In this paper, we investigate a quasi-online version of the List Model by supposing that jobs arrive in the non-increasing order of their processing times. We present a quasi-online algorithm with the competitive ratio being 3/2 and the lower bound 4/3.

1) 国家"973"重点基础研究专项经费(G1998030401)、国家自然科学基金(10271110)、高校优秀青年教师教学科研奖励计划资助项目和温州师范学院课题经费(2001Z06)

Supported by National "973" Fundamental Research Project of P. R. China(G1998030401), National Natural Science Fundation of P. R. China(10271110), the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of MOE, P. R. China, and Research Project of Wenzhou Teachers College(2001Z06)

收稿日期 2001-05-17 收修改稿日期 2001-08-21

Received May 17, 2001; in revised form August 21, 2001

Key words Semi online, algorithm, competitive ratio, scheduling, machine cost

1 引言

在经典的在线并行同型处理器调度问题中,$\{p_1, p_2, \cdots, p_n\}$表示 n 个相互独立的工件,每一工件被指派给事先给定的 m 台并行同型(parallel identical) 处理器加工,目标是极小化最大处理器负载,即 makespan C_{max}. 工件逐个到达,到达后即知其加工时间. 当工件到达时,在线算法必须立即将它指派给某台处理器加工,一旦安排好后不允许重排. 在安排某一工件时,对其后到达的工件的信息一无所知. 对此问题已有较多研究[1]. Imreh 和 Noga[2]提出上述问题的一个变形,其不同之处在于: 1)事先没有处理器; 2)当工件到达时,算法可以决定是否购买处理器和处理器的数量; 3)目标是使 makespan 和购买处理器费用之和最小. 他们将新模型称为 List Model.

List Model 与以往那些预先给定处理器集的经典调度有很大不同,就以往的问题来说,设计算法时对处理器数目是没有决定权的,而且无须考虑处理器的购买费用. 文献[2]最早把处理器费用这一概念加入到调度中来出于以下考虑:首先,现实中处理器是要花钱买的;其次,让算法决定处理器数是因为考虑到对于给定的一个输入,算法的执行高度依赖于处理器数,这一点在分析最坏情况时是很明显的;再次,研究这一变形也有利于我们发现另外的有意义的问题和获得对原问题更深刻的理解. 对于 List Model, 文献[2]给出了竞争比为 1.618 的在线算法,并证明任意在线算法的竞争比至少是 4/3. 这里, 对算法 A 和实例 I,用 $A(I)$和 $OPT(I)$分别表示算法 A 解 I 的目标函数值和相应的离线情形的最优目标值, 竞争比定义为对任意实例 I, 使 $A(I) \leqslant c\, OPT(I)$都成立的最小的 c.

本文考虑在线 List Model 的一个变形,即考虑它在半在线情形下算法的设计. 这里半在线意味着在设计算法之前知道工件的部分信息,同时,对工件的指派一旦作出就不可改变. 由于在实际的调度问题中, 往往并不是对工件信息一无所知. 例如: 由于在一个生产周期内处理器系统的加工能力(资源)的限制,可完工工件的总加工时间是固定的; 或可事先知道每个工件的加工时间落在某个已知区间内; 或知道工件到达的顺序,譬如工件以加工时间从大到小的顺序到达, 等等. 如何利用这些部分信息设计更好的算法是很有实际意义的工作. 半在线算法自 1996 年出现以来(参见文献[3,4]),受到越来越多的重视, 已广泛地出现在调度问题的研究中. 本文首次讨论 List Model 的半在线问题,我们假设工件按从大到小的顺序到达, 工件到达后就知道其精确加工时间(这种半在线模型由文献[5]提出). 我们给出此模型下 List Model 问题的一个竞争比为 3/2 的半在线算法 A,由此明显地改进了已有的在线算法的竞争比.

2 A_ρ的进一步分析

以下也用 p_i表示该工件的加工时间, 并假设任务以顺序 $p_1, p_2, \cdots, p_n$到来, 用 L 表示工件的最大加工时间, P (P_i)表示所有工件(前 i 个工件)的总加工时间. 在某个调度算法下, 用 m 表示最终购买的处理器数, p_l表示最后完工的工件,k 表示在 p_l到来那一刻拥有的处理器数, C_{max}表示 makespan. 通过标准化使购一台处理器的费用是 1,因此,半在线算

法的目标函数值为 $C_{\max}+m$. 令 $\rho_1=0,\rho_i=i^2,i=2,3\cdots$. 下面的解 List Model 的在线算法来自文献[2].

算法 A_ρ. 当 p_i到来时，A_ρ购买处理器(如果有必要)以使当前处理器数 j 满足 $\rho_j\leqslant P_i<\rho_{j+1}$，然后 A_ρ把 p_i指派给负载最小的处理器.

引理 1. 1)离线情形的最优目标值至少是 $2\sqrt{P}$，且若 $L\geqslant\sqrt{P}$，则最优目标值至少是 $L+P/L$；2)设 m 是满足 $m\leqslant\sqrt{P}<m+1$ 的正整数，则离线情形的最优目标值至少是 $\min\{m+P/m,\ m+1+P/(m+1)\}$.

证明. 1)来自文献[2]. 2)显然 $OPT(I)\geqslant\min\{\ i+P/i\ ;i$ 是任意正整数$\}=\min\{m+P/m,\ m+1+P/(m+1)\}$.　　证毕.

引理 2. 当 $L\leqslant 3$ 时，A_ρ的竞争比是 3/2 .

证明. 分三种情况讨论.

1) $m=1$. 这时 $P<4$，由引理 1 之 2)的证明过程易知，$OPT(I)\geqslant\min\{1+P,2+P/2\}$，于是 $A_\rho(I)/OPT(I)\leqslant\min\{(1+P)/(1+P),(1+P)/(2+P/2)\}\leqslant 5/4$.

2) $m=2$. 由算法的定义知 $4\leqslant P<9$，$OPT(I)\geqslant\min\{2+P/2,\ 3+P/3\}$. 若 $C_{\max}<4$，则 $A_\rho(I)/OPT(I)\leqslant(C_{\max}+m)/\min\{2+P/2,3+P/3\}<3/2$，因此假设 $C_{\max}\geqslant 4$. 考虑到购买第二台处理器时，第一台的负载小于 4，而购买第二台处理器后我们总是将工件分给负载最小的处理器加工，因此要使 $C_{\max}\geqslant 4$，必有在分配完所有工件后两台处理器的负载至多相差 $p_l\leqslant L$. 于是 $C_{\max}\leqslant(P-p_l)/2+p_l\leqslant(P-L)/2+L$，从而

$$\frac{A_\rho(I)}{OPT(I)}\leqslant\max\left\{\frac{(P-L)/2+L+2}{2+P/2},\frac{(P-L)/2+L+2}{3+P/3}\right\}\leqslant\max\left\{\frac{11}{8},\frac{4}{3}\right\}\leqslant\frac{3}{2}$$

3) $m\geqslant 3$. 由算法知 $m\leqslant\sqrt{P}<m+1$ ，$k\leqslant\sqrt{P_l}<k+1$，$p_l\leqslant L<3\leqslant m\leqslant\sqrt{P}$. 于是有 $A_\rho(I)\leqslant m+(P_l-p_l)/k+p_l$. 若 $m>k$，由引理 1 之 1)和上一个不等式，$A_\rho(I)/OPT(I)\leqslant(m+(k+1)^2/k+(k-1)\sqrt{P}/k)/(2\sqrt{P})\leqslant(2m+\sqrt{P})/(2\sqrt{P})\leqslant 3/2$；若 $m=k\geqslant 3$，同情况 2)可证结论成立. 至此，我们证明了算法的竞争比不超过 3/2. 为证 3/2 是紧的，考虑 $4N$ 个长为 $1/N$ 的工件(N 是充分大)，因此 $L=1/N<3$，显然算法解该实例的目标值是 $6-1/N$，故竞争比为 $3/2-1/(4N)\to 3/2$ ，当 $N\to\infty$时.　　证毕.

3　已知工件从大到小到达的 List Model

本节继续讨论 List Model 问题，且假设工件是以从大到小的顺序到达的. 下面给出该情形的一个竞争比为 3/2 的半在线算法. 注意，此时有 $L=p_1$. 为方便计，以下记 $T=\lceil 7L/4\rceil$.

算法 A.

1) 如果 $p_1\leqslant 3$，调用 A_ρ决定如何购买处理器并安排完所有工件，结束. 否则购买一台处理器加工 p_1.

2) i) 若到来的工件 $p_i>3L/4$，购一台新处理器加工它；ii) 若到来的工件 $p_i\leqslant 3L/4$，这时如果将它放在已购最小负载处理器上加工，使得新完工时间不超过 $3L/2$，则就将它放在已购的负载最小的处理器上加工；否则购一台新处理器加工它.

3) 如果已购处理器数少于 T 台，转 2).

4) 把到来的工件安排到已购的负载最小的处理器上加工,直到已安排的工件总加工时间至少是 T^2.

5) 调用算法 A_ρ来决定新处理器的购买和所有剩余工件的安排,结束.

注 1. 在第 4)步的进行过程中,处理器台数是 T.

注 2. 假定 $L=p_1>3$. 如果 $C_{\max}\leqslant 3L/2$, 则算法不会进入到第 4)步,这时最终购买的处理器数 $m\leqslant T$,此外,这时至多只有一台处理器的负载少于 $3L/4$,如果 $C_{\max}>3L/2$, 则这时最终购买的处理器数 $m\geqslant T$,此外,这时若算法一直没有进入第 5)步,则最终购买 T 台处理器,且每台处理器的负载至少是 $3L/4$,而处理器最大负载与最小负载至多相差 $p_l\leqslant L$.

定理 1. 算法 A 的竞争比为 3/2.

证明. 由引理 2 ,只须考虑 $L>3$ 的情形. 下面分三种情况来证明.

1) $m=1$. 这时 $A(I)\leqslant 3L/2+1$, $OPT(I)\geqslant L+1$,故 $A(I)/OPT(I)\leqslant 3/2$.

2) $m\geqslant 2$ 且 $C_{\max}\leqslant 3L/2$. 因此可设 $C_{\max}=(x+3/4)L$,这里 $x\in[1/4,\ 3/4]$. 由注 2 知, $P\geqslant (m-2)3L/4+(x+3/4)L=3(m-1)L/4+xL$.

i)当 $P\leqslant L^2$时,由引理 1 知 $OPT(I)\geqslant L+P/L$,因此

$$\frac{A(I)}{OPT(I)}\leqslant\frac{C_{\max}+m}{L+P/L}\leqslant\frac{3L/4+xL+m}{L+3(m-1)/4+x}\leqslant\frac{3L/2+m}{L+3m/4}\leqslant\frac{3}{2}$$

ii)当 $P>L^2$时,由 $C_{\max}\leqslant 3L/2$ 和 $L^2<P\leqslant mC_{\max}$知 $m>2L/3$;此外,由注 2 知 $m\leqslant T$. 又由引理 1 知 $OPT(I)\geqslant 2\sqrt{P}$. 因此,当 $3L/4\leqslant m\leqslant T$ 时

$$\frac{A(I)}{OPT(I)}\leqslant\frac{C_{\max}+m}{2\sqrt{P}}\leqslant\frac{3L/4+xL+m}{2\sqrt{3L(m-1)/4+xL}}\leqslant\frac{3L/2+m}{2\sqrt{3mL/4}}\leqslant\frac{3}{2}$$

而当 $2L/3<m<3L/4$ 时,有 $A(I)/OPT(I)\leqslant(3L/2+m)/(2\sqrt{P})\leqslant 3/2$.

3) $m\geqslant 2$ 且 $C_{\max}>3L/2$. 由注 2 知 $m\geqslant T$.

i) $P\geqslant T^2$. 此时有某些工件由第 5)步指派,而 $5<7L/4\leqslant T\leqslant m\leqslant\sqrt{P}<m+1$. 如果 $m=k$,则 $C_{\max}\leqslant(P_l-p_l)/k+p_l\leqslant P/m+L$,因此

$$A(I)/OPT(I)\leqslant(m+P/m+L)/(2\sqrt{P})\leqslant(1+(m+1)/m+4/7)/2\leqslant 3/2$$

如果 $m>k$,类似于引理 2 中之 3)的证明可得 $A(I)/OPT(I)\leqslant 3/2$.

ii) $P<T^2$. 这时最后一个到达的工件由第 4)步安排,由注 1 知 $m=T$. 设最小负载处理器的负载为 y,由注 2 知 $3L/4\leqslant y$,又由 $ym\leqslant P$ 可知 $y<m=T$. 分两种可能讨论:如果 $p_l\leqslant 3L/4$,则

$$\frac{A(I)}{OPT(I)}\leqslant\frac{y+p_l+T}{2\sqrt{yT}}\leqslant\frac{y+3L/4+T}{2\sqrt{yT}}\leqslant\frac{3L/2+T}{2\sqrt{3TL/4}}$$

由于 $T/(3L/4)\in[7/3,\ 25/9]$,所以 $A(I)/OPT(I)\leqslant 43/30\leqslant 3/2$; 反之 $p_l>3L/4$, 这时在 p_l到来之前那一刻, 算法已购买了 $m=T$ 台处理器, 且每台处理器上都至少已加工了一个工件(否则 p_l的完工时间不可能超过 L),故 $y\geqslant p_l$. 设离线最优解购买的处理数为 m^*,相应的 makespan 为 $C^*_{\max}$. 如果 $m\leqslant m^*$,则 $OPT(I)\geqslant m^*+P/m^*\geqslant m+P/m\geqslant m+y$, $A(I)\leqslant m+y+p_l\leqslant m+2y$,故 $A(I)/OPT(I)\leqslant 3/2$. 因此,假设 $m^*<m$. 同法可证 $A(I)/OPT(I)\leqslant 3/2$. 从而结束了 3)之(ii), 以及 3)的证明. 引理 2 的证明中的实例满足工件从大到小到达,因此 3/2 对算法 A 是紧的. 证毕.

定理 2. 任意工件从大到小到达的 List Model 的半在线算法的竞争比至少是 4/3.

证明. 设 B 是任意给定的半在线算法. 任取 $\varepsilon>0$,考虑这样一组实例 I_n,它有 n 个工件,工件的长度皆是 ε. 算法 B 有两种可能：1)对任意 I_n,B 只购买一台处理器. 此时有 $B(I_n)/OPT(I_n)\geqslant(n\varepsilon+1)/(n\varepsilon/2+2)\rightarrow 2(n\rightarrow\infty)$；2)存在某一 I_n,B 至少购买了两台处理器. 可设当第 d 个工件到来时,B 买了第二台处理器,以下考虑实例 I_d. 若 $P_d\leqslant 2$,则 $B(I_n)/OPT(I_n)\geqslant(2+P_d-\varepsilon)/(P_d+1)\geqslant(4-\varepsilon)/3$;若 $P_d>2$,则 $B(I_n)/OPT(I_n)\geqslant(2+P_d-\varepsilon)/(P_d/2+\varepsilon+2)\geqslant(4-\varepsilon)/(3+\varepsilon)$. 令 $\varepsilon\rightarrow 0$ 即证. 证毕.

References

1　Albers S. Better bounds for online scheduling. *SIAM Journal on Computing*, 1999, **29**(3): 459～473

2　Imreh C, Noga J. Scheduling with machine cost. Lecture Notes in Computer Science, Berlin: Springer-Verlag, 1999. 168～176

3　Liu W P, Sidney J B, van Vliet A. Ordinal algorithms for parallel machine scheduling. *Operations Research Letters*, 1996, **18**(2): 223～232

4　He Yong, Yang Qi-Fan, Tan Zhi-Yi. Semi online scheduling on parallel machine(I). *Applied Mathematics-A Journal of Chinese University*, 2003, **18A**(1): 105～114(in Chinese)

5　Seiden S, Sgall J, Woeginger G. Semi-online scheduling with decreasing job sizes. *Operations Research Letters*, 2000, **27**(2): 215～221

蔡圣义　1997 年北京师范大学数学系获学士学位,2003 年在浙江大学数学系获硕士学位,现为温州师范学院数学系讲师. 研究兴趣为调度理论及应用、算法设计与分析等.

(**CAI Sheng-Yi**　Received his bachelor degree from Beijing Normal University in 1997, master degree from Zhejiang University in 2003. He is currently working in the Department of Mathematics at Wenzhou Teachers College. His research interests include scheduling theory and applications, the design and analysis of algorithms.)

何　勇　1996 年浙江大学数学系获博士学位,1997 年在奥地利格拉茨工业大学数学研究所从事博士后研究. 现为浙江大学数学系教授,博士生导师. 研究兴趣为组合最优化、调度理论及应用、数学建模等.

(**HE Yong**　Received his bachelor, master, and Ph. D. degrees from Zhejiang University in 1989, 1992 and 1996, respectively. He did research as postdoctor at Institute of Mathematics, Technical University Graz in 1997. Currently he is working in the Department of Mathematics at Zhejiang University, as a professor. His research interests include combinatorial optimization, scheduling theory and applications, the design and analysis of algorithms, and mathematical modelling.)