带惩罚费用的多重任务排序问题*

崔倩娜
（云南大学数学与统计学院　昆明　650000）

摘　要　提出带惩罚费用的多重任务排序问题，即每个用户提交多个加工时间和惩罚费用相同的任务。每个用户提交的任务要么全部被接受，并被安排在平行机上处理，或者全部被拒绝，并产生惩罚费用。目标是寻找一个排序方案，使得机器的最大完工时间与所有被拒绝用户的惩罚费用之和最小。设计了一般情形下的一个强多项式时间2-近似算法和机器数为固定常数时的一个全多项式时间近似方案。特别地，当机器数为2时，设计了一个竞争比为1.618的最优在线算法。

关键词　多重任务；惩罚费用；排序问题；近似算法；在线算法
中图分类号　TP301.6　　**DOI**：10. 3969/j. issn. 1672-9722. 2019. 01. 024

Multi-task Scheduling Problem with Rejection

CUI Qianna
（School of Mathematics and Statistics，Yunnan University，Kunming　650000）

Abstract　Multi-task scheduling problem with rejection is proposed，in which each user submits multiple tasks with the same processing time and penalty cost. All tasks submitted by each user are either accepted and processed by the machines，or rejected，in which case its penalty cost is paid. The objective is to minimize the sum of the makespan and the total penalty cost of the rejected tasks. A 2-approximation algorithm is presented in strongly polynomial time for the general case and a full polynomial time approximation scheme when the number of machines is fixed. Especially，when the number of machines is 2，an optimal online algorithm is designed with a competitive ratio of 1.618.

Key Words　multiple tasks，rejection，scheduling problem，approximate algorithm，online algorithm
Class Number　TP301.6

1　引言

长期以来，排序问题的近似算法研究都是算法理论领域研究的热点问题之一。受滑雪板租赁问题的启发，Bartal 等[1]提出带惩罚费用的平行机排序问题 $P\,/\!/\,C_{\max}+\Sigma_{j\in R}w_j$，其定义如下：给定 m 台平行机和 n 项任务，每项任务的处理时间为 p_j，惩罚费用为 w_j，一项任务要么被接受并在某台机器上处理，要么被拒绝并产生相应的惩罚费用。该问题的目标是寻找一个排序方案，使得机器的最大完工时间与被拒绝任务的惩罚费用之和最小。Bartal 等[1]设计了一个运行时间为 $O(n\log n)$ 的 $(2-1/m)$-近似算法和一个运行时间为 $O((n^3/\varepsilon)^{\lceil 9/\varepsilon\rceil})$ 的多项式时间近似方案（Polynomial Time Approximation Scheme，PTAS），推广了 Hochbaum 和 Shmoys[2]的结果。当机器数为固定常数时，Bartal 等[1]还给出了一个运行时间为 $O(n^{m+1}/\varepsilon^n)$ 的全多项式时间近似方案（Fully Polynomial Time Approximation Scheme，FPTAS）。随后，带惩罚费用的机器排序问题及其变种迅速成为排序领域的热点问题之一。Ou 等[3]给出 $P\,/\!/\,C_{\max}+\Sigma_{j\in R}w_j$ 问题的一个运行时间为 $O(n\log n+n/\varepsilon)$ 的 $(1.5+\varepsilon)$-近似算法（这里 $\varepsilon>0$ 为任意常数）。最近，Ou 和 Zhong[4]给出 $P\,/\!/\,C_{\max}+\Sigma_{j\in R}w_j$ 问题的一个运行时间为 $O(mn^2/\varepsilon^2)$ 的 $(4/3+\varepsilon)$-近似算法。

Zhang 和 Lu[5]考虑了带就绪时间和惩罚费用的

* **收稿日期**：2018 年 7 月 12 日，**修回日期**：2018 年 8 月 14 日
基金项目：国家自然科学基金（编号：61662088）；云南省自然科学基金（编号：2014FB114）和 IRTSTYN 资助。
作者简介：崔倩娜，女，硕士研究生，研究方向：机器排序问题。

平行机排序问题 $P|r_j|C_{\max}+\Sigma_{j\in R}w_j$（这里 r_j 为任务的就绪时间），设计了一个运行时间为 $O(n^2)$ 的2-近似算法，当机器数为固定常数时，给出一个运行时间为 $O(n^{2m+1}/\varepsilon^m)$ 的FPTAS。随后，Zhong和Ou[6]提出一个运行时间为 $O(n\log n)$ 的2-近似算法，一个运行时间为 $O(n\log n+m^{O((\log 1/\varepsilon)/\varepsilon^2)})$ 的PTAS，和一个机器数为固定常数时运行时间为 $O(n\log n+1/\varepsilon^{3m+6})$ 的FPTAS，改进了Zhang和Lu[5]的结果。Zhong等[7]研究文献[4]中 $m=2$ 的情况，给出一个运行时间为 $O((n/\varepsilon)^2)$ 的 $(1.5+\varepsilon)$ -近似算法。Li等[8]研究了惩罚费用受限的平行机排序问题 $P|\Sigma_{j\in R}w_j|C_{\max}$，设计了一个运行时间为 $O(mn^2)$ 的2-近似算法，一个运行时间为 $O(nm^{O(1/\varepsilon^2)}+mn^2)$ 的PTAS和机器数为固定常数时的一个运行时间为 $O(1/\varepsilon^{2m+3}+mn^2)$ 的FPTAS。Shabtay等[9]研究了处理时间相同时的带惩罚费用的双标准同类机排序问题。

当机器数为1时，Zhang等[10]证明了带就绪时间和惩罚费用的单机排序问题 $1|r_j,rej|C_{\max}+\Sigma_{j\in R}w_j$ 是NP-难的，设计了一个运行时间为 $O(n^2)$ 的2-近似算法和一个运行时间为 $O(n^3/\varepsilon)$ 的FPTAS。Ou等[11]提出一个运行时间为 $O(n\log n)$ 的2-近似算法和所有任务的加工时间相同时的一个运行时间为 $O(n^2\log n)$ 的精确算法。Zhang等[12]证明了带就绪时间和惩罚费用受限的单机排序问题 $1|r_j,\Sigma_{j\in R}w_j\le U|C_{\max}$ 是NP-难的，给出一个运行时间为 $O(n^3/\varepsilon)$ 的FPTAS。Lu和Zhang[13]考虑了带生产费用和惩罚费用的单机排序问题。Shabtay等[14]研究了带惩罚费用和位置费用的双标准的单机排序问题，设计了多个最优算法或近似算法。He等[15]考虑了带就绪时间和惩罚费用的双标准的单机排序问题，对于最大完工时间与惩罚费用之和最小化问题，给出一个运行时间为 $O(n\log n)$ 的4/5-近似算法；当最大完工时间受限时，设计了一个运行时间为 $O(n^2/\varepsilon+n^2\log n)$ 的FPTAS；当惩罚费用受限时，设计了一个运行时间为 $O(n^2)$ 的2-近似算法和一个运行时间为 $O(n^2/\varepsilon+n^2\log n)$ 的FPTAS，推广了Gens和Levner[16]的结果。

Bartal等[1]还对带惩罚费用的在线机器排序问题做了研究，所有任务的加工时间和惩罚费用都是未知的，给出一个竞争比为2.618的最优在线算法；当机器数为2时，设计了一个竞争比为1.618的最优在线算法。Seiden[17]研究了带惩罚费用的可中断平行机在线排序问题，给出一个竞争比为 $(4+\sqrt{10})/3$ 的在线算法，并指出任何在线算法的下界为2.12。Gyorgy和Imreh[18]提出带惩罚费用和机器费用的在线排序问题，设计了一个竞争比为 $(3+\sqrt{5})/2$ 的最优在线算法。Epstein和Haider[19]研究了关于带惩罚费用的三台机器在线排序问题，当所有任务的处理时间为1，设计了一个竞争比1.839的最优在线算法。

由于在高性能计算环境中，用户提交的任务通常具有数量大且处理时间相同等特征，多重任务排序引起了学者的广泛关注[20~21]。另一方面，Goemans和Rothvoß[22]给出多重物品装箱问题的一个最优算法。受文献[20~22]的启发，将任务或物品的类型视为用户，本文提出了带惩罚费用的多重任务排序问题（Multi-task Scheduling Problem with Rejection，MTSR），其定义如下：给定 n 个用户，每个用户 j 提交 t_j 项任务，任务集记为 T_j，T_j 中的每个任务具有相同的加工时间 p_j 和惩罚费用 w_j。每个用户提交的任务要么全部被接受，并被安排在 m 台机器上处理；要么全部被拒绝，并产生相应的惩罚费用。本文假定，一旦某项任务开始在某台机器上处理，则中间不被打断直到完工，而且所有任务的安排没有优先权。目标是寻找一个排序方案，使得机器的最大完工时间与所有被拒绝的任务的惩罚费用之和达到最小。

为更加清晰地描述问题，用 x_{ij} 表示被接受的用户 j 提交的任务被安排在机器 i 上的数量，z_j 表示用户 j 是否被接受，若被接受，则 $z_j=1$，否则，$z_j=0$。MTSR问题的数学规划形式如下：

$$\begin{cases}\min \quad C+\sum_{j=1}^{n}w_jt_j(1-z_j)\\ \sum_{j=1}^{n}p_jx_{ij}\le C,i=1,2,\cdots,m\\ \sum_{i=1}^{m}x_{ij}=t_jz_j,j=1,2,\cdots,n\\ x_{ij}\in Z^+\cup\{0\},\ z_j\in\{0,1\}\end{cases}\tag{1}$$

注意到，当每个用户提交的任务数量 t_j 都为1时，MTSR问题即为Bartal等[1]研究的问题 $P//C_{\max}+\Sigma_{j\in R}w_j$。因此，一种直观的想法是将所有任务的集合看作问题 $P//C_{\max}+\Sigma_{j\in R}w_j$ 的任务集，从而使用[1]中的算法进行求解。但是，运行时间并不是关于输入长度的多项式函数。

2　MTSR 问题的离线算法

2.1　2–近似算法

本节讨论一般情形下的 MTSR 问题，并给出一个 2–近似算法 H 。算法 H 的主要思想是将用户集合 U 中一部分惩罚费用较小的用户都拒绝，这里 $|U|=n$ ；然后，将剩余用户提交的所有任务划分成 m 个集合；最后，把划分后的集合看作一个整体任务，并将其安排在 m 台机器上，接受一些加工时间较小的用户。

算法 1　H

1）令 $B=\{j|w_j\le p_j/m\}$ ，将集合 B 中的所有用户都拒绝；

2）对 $U-B$ 中的用户，以加工时间非减顺序排列；

3）将 $U-B$ 中的每一个用户 j 提交的所有任务分割成 m 个任务集合，即

$$T_j=T_{j1}\cup\cdots\cup T_{jk}\cup T_{j,k+1}\cup\cdots\cup T_{jm}$$

这里 $k=t_j-\left\lfloor t_j/m\right\rfloor m$

$$\left|T_{j1}\right|=\cdots=\left|T_{jk}\right|=\left\lceil t_j/m\right\rceil$$

$$\left|T_{j,\ k+1}\right|=\cdots=\left|T_{jm}\right|=\left\lfloor t_j/m\right\rfloor$$

即 $T_{j1},\cdots,T_{jk}$ 的每个集合中都含有用户 j 所提交的 $\left\lceil t_j/m\right\rceil$ 个任务，$T_{j,k+1},\cdots,T_{jm}$ 中每个集合都含有用户 j 所提交的 $\left\lfloor t_j/m\right\rfloor$ 个任务。计算每个被划分后的集合中总共的加工时间和惩罚费用，即

$$P_{j1}=\cdots=P_{jk}=p_j\left\lceil t_j/m\right\rceil$$

$$P_{j,k+1}=\cdots=P_{jm}=p_j\left\lfloor t_j/m\right\rfloor$$

$$W_{j1}=\cdots=W_{jk}=w_j\left\lceil t_j/m\right\rceil$$

$$W_{j,k+1}=\cdots=W_{jm}=w_j\left\lfloor t_j/m\right\rfloor$$

4）对每一个 $0\le h\le|U-B|$ ，从 $U-B$ 中选取的前 h 个用户，将这 h 个用户中的所有任务集（每个任务集视作一个任务，共 hm 个任务）用 LS（List Scheduling）算法安排到 m 台机器上，然后将剩下的所有用户都拒绝，令这一调度方案为 S_h 。

5）在 $|U-B|+1$ 个可行方案中，选择目标函数值最小的方案 S_h 。

定理 1　算法 H 是一个运行时间为 $O(n^2\log n)$ 的 2–近似算法。

证明　令 A^* ，R^* 分别表示最优方案中被接受的用户集合和被拒绝的用户集合，A^*T^* 和 R^*T^* 分别表示用户集合 A^* 和 R^* 对应的任务集合，Z^* 为最优方案的目标函数值。令 UT 表示用户集合 U 对应的任务集，A（或 R ）表示算法 H 的输出解中被接受（或被拒绝）的用户集合，AT（或 RT ）表示对应的任务集。令 Z^H 表示算法 H 的输出解的目标函数值。对任意任务集 T ，令 $M(T)=\sum_{j\in T}p_j/m$ 和 $W(T)=\sum_{j\in T}w_j$ 分别表示任务集 T 中所有任务的平均负载和总惩罚费用。对任意用户集合 $X\subseteq U$ ，令 $C(X)$ 表示 X 中被接受用户提交的所有任务按 LS 算法安排在 m 台机器上之后，机器的最大完工时间。

假设算法第 2）步中，用户集合 $U-B$ 排序结果为 $1,\cdots,|U-B|$ 。如果最优方案拒绝 $U-B$ 中所有用户，由 B 的定义知，拒绝 B 中所有用户也是最有的。因此，方案 S_0 拒绝所有用户是最优的，即 $Z^H=Z^*$ 。

否则，令 l 为在最优方案中从用户集合 $U-B$ 中接受的最后一个用户。考虑算法 H 的输出解 S_l ，不失一般性，令 $A=\{1,\cdots,l\}$ 表示调度方案 S_l 所接受的用户集合，则 l 为所有被接受的用户里，任务加工时间最大的用户。由于用户集合 A 中的所有任务集合按 LS 方法安排在 m 台机器上。所以，调度方案 S_l 的机器的最大完工时间至多为

$$C(A)\le M(AT\backslash\{T_{l,m}\})+P_{lm}\le M(AT\backslash\{T_{l,m}\})+\left\lfloor t_l/m\right\rfloor p_l \tag{2}$$

由于算法 H 的输出的目标函数值至多为方案 S_l 的目标函数值，则

$$\begin{aligned}Z^H&=C(A)+W(RT)\\&\le M(AT\backslash\{T_{l,m}\})+\left\lfloor t_l/m\right\rfloor p_l+W(UT\backslash AT)\\&\le M((AT\backslash\{T_{l,m}\})\cap A^*T^*)\\&+M((AT\backslash\{T_{l,m}\})\cap R^*T^*)+W((UT\backslash AT)\cap A^*T^*)\\&+W((UT\backslash AT)\cap R^*T^*)+\left\lfloor t_l/m\right\rfloor p_l\end{aligned} \tag{3}$$

因为集合 A 不包含 B 中任何用户，所以，有 $M((AT\backslash\{T_{l,m}\})\cap R^*T^*)\le W((AT\backslash\{T_{l,m}\})\cap R^*T^*)$ 。由 l 的选择以及 B 的定义可知，$A^*\cap(U-A)\subseteq B$ 。这是由于 $U-A=\{l+1,\cdots,|U-B|\}\cup B$ ，而 l 为 A^* 中从 $U-B$ 中接受的最后一个用户，所以 A^* 中不包含用户集合 $\{l+1,\cdots,|U-B|\}$ 中任意一个用户。从而，可以得到

$W((UT\backslash AT)\cap A^*T^*)\le M((UT\backslash AT)\cap A^*T^*)$ 。因此，上面的式（3）可以整理为

$$\begin{aligned}Z^H &\le M((AT\backslash\{T_{l,m}\})\cap A^*T^*)+W((AT\backslash\{T_{l,m}\})\cap R^*T^*)\\&\quad+M((UT\backslash AT)\cap A^*T^*)+W((UT\backslash AT)\cap R^*T^*)\\&\quad+\lfloor t_l/m\rfloor p_l\\&\le M(A^*T^*\backslash\{T_{l,m}\})+W(R^*T^*\backslash\{T_{l,m}\})+\lfloor t_l/m\rfloor p_l\\&\le Z^*(UT\backslash\{T_{l,m}\})+\lfloor t_l/m\rfloor p_l\\&\le Z^*+Z^*=2Z^*\end{aligned}\tag{4}$$

所以,算法 H 的近似比为2。

算法 H 的第一步选出属于集合 B 中的用户需要 $O(n)$ 时间,第二步给用户排序最多需要 $O(n\log n)$ 时间,第三步将集合 $U-B$ 中每个用户提交的所有任务划分成 m 个任务集最多需要 $O(m)$ 时间,分割所有用户提交的所有任务最多需要 $O(nm)$ 时间。计算每个集合的运行总时间和总惩罚费用最多可以在 $O(nm)$ 内完成。第四步和第五步是选取方案 S_h,使用了 LS算法,所需要时间为 $O(nm\log m)$。当 $n>m$ 时,算法的运行时间不会超过 $O(n^2\log n)$。证毕。

2.2 机器数为固定常数时的一个 FPTAS

当机器数 m 为固定常数时,在动态规划的基础上采用舍入取整技术,设计了一个FPTAS。

定理 2 当机器数为固定常数时,MTSR 问题可以在多项式时间 $O(n(t_{\max}Z^*)^m)$ 内解决,其中 $t_{\max}=\max\{t_1,\cdots,t_n\}$,$Z^*$ 是最优方案的目标函数值。

证明 采用动态规划方法。

令 $L_i(i=1,2,\cdots,m)$ 为机器 i 的当前负载,对于每个 $L_1,L_2,\cdots,L_m\le Z^*$,计算在这些负载下可以获得的总惩罚费用的最小值。在第 j 个用户被安排或拒绝后,用 $E_j(L_1,L_2,\cdots,L_m)$ 来定义当前负载下可以获得的总惩罚费用的最小值。令 a_{ij} 为将用户 j 提交的任务被安排在机器 i 上的数量。当 $L_i<0$ 时,定义惩罚费用的最小值为 ∞。同时,可以在具有机器负载 $L_1,L_2,\cdots,L_m$ 的情况下可以计算出最后总花费 $Z(L_1,\cdots,L_m)$。对于 $L_1,\cdots,L_m\ge 0$,这些值的计算可以用以下方式得出:

初始化:$E_0(0,\cdots,0)=0$;

$E_0(L_1,\cdots,L_m)=\infty$,$L_1^2+\cdots+L_m^2\ne 0$。

$$E_j(L_1,\cdots,L_m)=\min\{t_jw_j+E_{j-1}(L_1,\cdots,L_m),$$
$$\min\{E_{j-1}(L_1-a_{1j}p_j,\cdots,L_m-a_{mj}p_j)\Big|\textstyle\sum_{i=1}^m a_{ij}=t_j\}\},$$
$$Z(L_1,\cdots,L_m)=E_n(L_1,\cdots,L_m)+\max_i L_i。$$

一旦 $\max_i L_i$ 使目标函数值达到了最优,则停止迭代,输出最优函数值 Z。每一个 $L_i\le Z^*$ 都有 Z^*+1 种情况可以选择,每一个 a_{ij} 都有 t_j+1 种选择。所以在每一个用户 j 所提交的所有任务被安排或拒绝后,惩罚费用最小值 $E_j(L_1,\cdots,L_m)$ 可以在关于多项式 $(Z^*)^m$ 和 $(t_{\max})^m$ 的时间内获得。将所有用户提交的所有任务都安排之后,得到 $Z(L_1,\cdots,L_m)$,故最终所需要的时间为 $O(n(t_{\max}Z^*)^m)$。证毕。

定理 3 对任意 $\varepsilon>0$,MTSR 问题存在一个运行时间为 $O((\frac{4}{\varepsilon})^m(nt_{\max})^{m+1})$ 的FPTAS。

证明 将所研究的实例 I,通过舍入取整技术构造一个新的实例 I'。每个用户 j 提交 t_j 个任务,这 t_j 个任务有相同的加工时间 $p_j'=\lfloor p_j/\delta\rfloor$ 和惩罚费用 $w_j'=\lfloor w_j/\delta\rfloor$,其中,$\delta=\varepsilon Z^H/2n(t_{\max})$,这里 Z^H 为通过算法 H 获得的目标函数值。采用定理2中的动态规划获得实例 I' 的一个最优排序方案,将此方案用在实例 I 上,获得实例 I 的一个近似方案。

通过上述方式获得的实例 I 的近似函数值 $Z(I)$ 与最优函数值最多相差 $\delta nt_{\max}=\varepsilon Z^H/2$。由已经获得的最优函数值的下界 $Z^*\ge Z^H/2$,可以得到

$$\frac{|Z(I)-Z^*|}{Z^*}\le\frac{\delta nt_{\max}}{Z^*}\le\frac{\varepsilon Z^H/2}{Z^H/2}=\varepsilon\tag{5}$$

所以,有 $\frac{Z(I)}{Z^*}\le 1+\varepsilon$。

由于实例 I' 的最优函数值与实例 I 的最优函数值满足 $Z^*\le Z^*(I)/\delta\le 2Z^H/\delta$,所以 $Z^*\le 4nt_{\max}/\varepsilon$。定理2指出,实例 I 的最优解可以在时间 $O(n(t_{\max}Z^*)^m)$ 内得到,故实例 I' 的最优解可以在多项式时间 $O((\frac{4}{\varepsilon})^m(nt_{\max})^{m+1})$ 内得到。证毕。

3 关于两台机器的在线算法

在离线问题中,所有用户提交的任务数量,每项任务的加工时间,以及被拒绝后的惩罚费用都是已知的。而要研究的在线问题,在上一个用户提交的所有任务被安排完之后,才会获得一个新用户的信息。当机器数为2时,设计了一个在线算法 A_α,算法的设计思想是将惩罚费用较小的用户都拒绝,再将剩余用户提交的任务划分成两个任务集,把这两个任务集分别安排在两台机器上。

算法 2 A_α

1)如果用户 j 提交的所有任务的加工时间和惩罚费用满足 $w_j\le\alpha p_j$,则拒绝用户 j;

2)否则,将用户 j 提交的所有任务分割成 2 个任务集,$T_j=T_{j1}\cup T_{j2}$,其中 $|T_{j1}|=\lceil t_j/2\rceil$,而 $|T_{j2}|=\lfloor t_j/2\rfloor$,计算两个被划分后的任务集中总共的加工时间和惩罚费用,即 $P_{j1}=\lceil t_j/2\rceil p_j$,$P_{j2}=\lfloor t_j/2\rfloor p_j$, $W_{j1}=\lceil t_j/2\rceil w_j$, $W_{j2}=\lfloor t_j/2\rfloor w_j$,且将两个集合依次安排在当前机器负载最小的机器上。

定理 4　如果 α 满足不等式(6),则在线算法 A_α 的竞争比为 1.618。

$$\begin{cases}2\alpha\geq 1\\ 1+\alpha\leq 1/\alpha\\ 2\alpha\leq 1/\alpha\end{cases}\tag{6}$$

证明　分两种情况来证明:

1)若算法 A_α 拒绝了所有用户,则由算法输出的目标函数值

$$\begin{aligned}Z^{A_\alpha}&=W(UT)=W(A^*T^*\cup R^*T^*)\\&=W(A^*T^*)+W(R^*T^*)\end{aligned}\tag{7}$$

由于所有属于 A^* 的用户都被算法拒绝,所以,对于每一个 $j\in A^*$ 满足 $w_j\leq\alpha p_j$。则 A^* 提交的所有任务的惩罚费用之和满足 $W(A^*T^*)\leq 2\alpha M(A^*T^*)$。由条件(6)可将式(7)经整理得

$$Z^{A_\alpha}\leq 2\alpha M(A^*T^*)+W(R^*T^*)\leq 2\alpha Z^*\tag{8}$$

2)否则,令 l 为最后一个被接受的用户,T_{l2} 为 T_l 中最后一个被安排的任务集合。则所有被接受的用户提交的所有任务被安排到 2 台机器上之后,机器的最大完工时间为

$$C(A)\leq M(AT\backslash T_l)+M(T_l\backslash\{T_{l2}\})+P_{l2}\tag{9}$$

由于算法 A_α 拒绝的用户集合 R,其中每项任务的加工时间和惩罚费用满足 $w_j\leq\alpha p_j$,则会有 $\sum_{j\in R}t_jw_j\leq 2\alpha\sum_{j\in R}t_jp_j/2$,所以算法输出的用户集合 R 提交的所有任务的惩罚费用之和满足:

$$\begin{aligned}W(RT)&=W(RT\cap A^*T^*)+W(RT\cap R^*T^*)\\&\leq 2\alpha M(RT\cap A^*T^*)+W(RT\cap R^*T^*)\end{aligned}\tag{10}$$

由条件(6)可知,算法 A_α 输出的被接受的用户集合 A 提交的任务满足 $w_j\geq\alpha p_j\geq\frac{1}{2}p_j$,则集合 A 提交的所有任务的平均负载小于总惩罚费用。从而会有

$$\begin{aligned}M(AT\backslash T_l)&=M((AT\cap A^*T^*)\backslash T_l)+M((AT\cap R^*T^*)\backslash T_l)\\&\leq M((AT\cap A^*T^*)\backslash T_l)+W((AT\cap R^*T^*)\backslash T_l)\end{aligned}\tag{11}$$

由条件(6)和(9)、(10)、(11)可以得到算法 A_α 输出的目标函数值为

$$\begin{aligned}Z^{A_\alpha}&=C(A)+W(RT)\\&\leq M(A\backslash T_l)+M(T_l\backslash\{T_{l2}\})+P_{l2}+W(RT)\\&\leq M((AT\cap A^*T^*)\backslash T_l)+W((AT\cap R^*T^*)\backslash T_l)\\&\quad+2\alpha M(RT\cap A^*T^*)+W(RT\cap R^*T^*)\\&\quad+M(T_l\backslash\{T_{l2}\})+P_{l2}\\&\leq 2\alpha M(A^*T^*\backslash T_l)+W(R^*T^*\backslash T_l)+M(T_l\backslash\{T_{l2}\})+P_{l2}\end{aligned}\tag{12}$$

为证明竞争比,则需要讨论算法输出的被接受的最后一个用户 l 是否属于最优方案中被接受的用户集合,分成两种情况来说明:

1)如果用户 $l\in A^*$,由不等式(6)和等式

$$2\alpha M(\{T_{l2}\})+(1-\alpha)P_{l2}=\alpha P_{l2}+(1-\alpha)P_{l2}=P_{l2}$$

可以将上式转化为

$$\begin{aligned}Z^{A_\alpha}&\leq 2\alpha M(A^*T^*\backslash\{T_{l2}\})+W(R^*T^*\backslash\{T_l\})\\&\quad+2\alpha M(\{T_{l2}\})+(1-\alpha)P_{l2}\\&\leq 2\alpha M(A^*T^*)+W(R^*T^*\backslash T_l)+(1-\alpha)P_{l2}\\&\leq 2\alpha Z^*+(1-\alpha)Z^*\\&=(\alpha+1)Z^*\end{aligned}\tag{13}$$

2)如果用户 $l\in R^*$,但是算法输出用户 l 被接受,所以 $w_l>\alpha p_l\geq\frac{1}{2}p_l$,由条件(6),可将上面式(12)整理后可得

$$\begin{aligned}Z^{A_\alpha}&\leq 2\alpha M(A^*T^*\backslash T_l)+W(R^*T^*\backslash T_l)\\&\quad+W(T_l\backslash\{T_{l2}\})+\frac{1}{\alpha}W_{l2}\\&\leq 2\alpha M(A^*T^*\backslash T)+W(R^*T^*\backslash\{T_{l2}\})+\frac{1}{\alpha}W_{l2}\\&\leq 2\alpha M(A^*T^*\backslash T)+\frac{1}{\alpha}W(R^*T^*)\\&\leq\frac{1}{\alpha}Z^*\end{aligned}\tag{14}$$

故,当 A_α 满足不等式(6)时,在线算法 A_α 的竞争比为 $\frac{1}{\alpha}$。

对式(6)中不等式组做计算后,得出 α 的取值范围为 $\frac{1}{2}\leq\alpha\leq\frac{\sqrt{5}-1}{2}$,所以,竞争比的可取范围为 $1.618\leq\frac{1}{\alpha}\leq 2$。由此,若 $\alpha=\frac{\sqrt{5}-1}{2}$ 时,可以获得最优在线算法,其竞争比为 1.618。

当用户提交的任务数量都为 1 时,Bartal 等[1]已证明不存在竞争比小于 1.618 的在线算法,所以,算法 A_α 为最优在线算法。证毕。

4　结语

本文提出一个带惩罚费用的多重任务排序问题,针对离线问题,设计了一个 2-近似算法和一个

FPTAS。对于在线问题,当机器数为2时,设计了一个竞争比为1.618的最优在线算法。

未来值得研究的问题有:利用Ou等[3]的算法思想设计MTSR问题的一个$(1.5+\varepsilon)-$的近似算法;设计MTSR问题的一个运行时间更低的FPTAS的;利用文献[22]中的算法思想设计一个用户数为固定常数时MTSR问题的一个最优算法。

参考文献

[1] Y. Bartal, S. Leonardi, A.M. Spaccamela, J. Sgall and L. Stougie, Multiprocessor scheduling with rejection[J]. SIAM J. Discrete Math,2000(13):64-78.

[2] D.S. Hochbaum, D.B. Shmoys. Using dual approximation algorithm for scheduling problems: theoretical and practical results[J]. Journal of the Association for Computing Machinery,1987(34):144-162.

[3] J.W. Ou, X.L. Zhong, G.Q. Wang. An improved heuristic for parallel machine scheduling with rejection[J]. European Journal of Operational Research,2015(241): 653-661.

[4] X.L. Zhong, J.W. Ou. Bicriteria order acceptance and scheduling with consideration of fill rate[J]. European Journal of Operational Research, 2017(forthcoming).

[5] L.Q. Zhang, L.F. Lu. Parallel-machine scheduling with release dates and rejection[J]. 4OR A Quarterly Journal of Operations Research,2016(14):165-172.

[6] X.L. Zhong, J.W. Ou. Improved approximation algorithms for parallel machine scheduling with release dates and job rejection[J]. 4OR A Quarterly Journal of Operations Research, 2016: 1-12.

[7] X.L. Zhong, Z.M. Pan, D.K. Jiang. Scheduling with release times and rejection on two parallel machines[J]. Journal of Combinatorial Optimization, 2016 (33): 934-944.

[8] W.D. Li, J.P. Li, X.J. Zhang, Z.B. Chen. Penalty cost constrained identical parallel machine scheduling problem [J]. Theoretical Computer Science,2015(607):181-192.

[9] D. Shabtay, S. Karhi, D. Oron. Multipurpose machine scheduling with rejection and identical job processing times[J]. Journal of Scheduling, 2014(18):75-88.

[10] L.Q. Zhang, L.F. Lu, J.J. Yuan. Single machine scheduling with release dates and rejection[J]. European Journal of Operational Research, 2009(198):975-978.

[11] J.W. Ou, X.L. Zhong, C.L. Li. Faster algorithms for single machine scheduling with release dates and rejection [J]. Information Processing Letters, 2016 (116): 503-507.

[12] L.Q. Zhang, L.F. Lu, J.J. Yuan. Single-machine scheduling under the job rejection constraint[J]. Theoretical Computer Science,2010(411):1877-1882.

[13] L.F. Lu, L.Q. Zhang. Single-machine with production and rejection costs to minimize the maximum earliness [J].Journal of Combinatorial Optimization,2016.

[14] D. Shabtay, N. Gaspar, L. Yedidsion. A bicriteria approach to scheduling a single machine with job rejection and positional penalties[J]. Journal of Combinatorial Optimization,2012(23):359-424.

[15] C. He, J.Y.T. Leung, K.B. Lee, M.L. Pinedo. Improved algorithms for single machhine scheduling with release dates and rejections[J]. 4OR A Quarterly Journal of Operations Research, 2016(14): 41-55.

[16] G.V. Gens, E.V. Levener. Fast approximation algorithm for job sequencing with deadlines[J]. Discrete Applied Mathematics,1981(3): 313-318.

[17] S.S. Seiden. Preemptive multiprocessor scheduling with rejection [J]. Theoretical Computer Science, 2001 (262):437-458.

[18] J.N. Gyorgy, C. Imreh. Online scheduling with machine cost and rejection [J].Discrete Applied Mathematics, 2007(155): 2546-2554.

[19] L. Epstein, H.Z. Haider. Online scheduling of unit jobs on three machines with rejection: A tight result[J]. Information Processing Letters,2016(116): 252-255.

[20] K. M. Tarplee, R. Friese, A. A. Maciejewski, H. J. Siegel, E. KP Chong. Energy and makespan tradeoffs in heterogeneous computing systems using efficient linear programming techniques[C]//IEEE Transactions on Parallel and Distributed Systems, 2016, 27(6):1633-1646.

[21]W.D. Li, X. Liu, X.J. Zhang, X.B. Cai. A task-type-based algorithm for the energy-aware profit maximizing scheduling problem in heterogeneous computing systems[J]. Proceeding of the 15th IEEE/ACM International Symposium on luster, Cloud and Grid Computing (CCGrid), 2015:1107-1110.

[22] M. X. Goemans, T. Rothvoß. Polynomiality for bin packing with a constant number of item types[J]. Proceeding of the the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2014:830-839.