Pure Mathematics 理论数学, 2018, 8(5), 480-485
Published Online September 2018 in Hans. http://www.hanspub.org/journal/pm
https://doi.org/10.12677/pm.2018.85064

Hans汉斯

Multidimensional Task Scheduling Problem under a Grade of Service Provision

Bingfei Dai

School of Mathematics and Statistics, Yunnan University, Kunming Yunnan
Email: 2290477053@qq.com

Received: Aug. 12th, 2018; accepted: Aug. 30th, 2018; published: Sep. 6th, 2018

Abstract

According to vector scheduling problem and task scheduling problem under a grade of service provision, we propose multidimensional task scheduling problem under a grade of service provision. We first design a 5 d/4 approximation algorithm. Then, we give a dynamic programming algorithm and a fully polynomial time approximation scheme.

Keywords

Vector Scheduling, Dynamic Programming, A Grade of Service Provision, Approximation Algorithm

带等级约束的多维任务调度问题

代兵飞

云南大学数学与统计学院，云南 昆明
Email: 2290477053@qq.com

收稿日期：2018年8月12日；录用日期：2018年8月30日；发布日期：2018年9月6日

摘　要

根据向量调度问题和带等级约束的任务调度问题，我们提出了带等级约束的多维任务调度问题。我们首先设计了一个5 d/4-近似算法。然后，我们给出一个动态规划算法和一个完全多项式时间近似方案。

关键词

向量调度，动态规划，服务等级，近似算法

文章引用: 代兵飞. 带等级约束的多维任务调度问题[J]. 理论数学, 2018, 8(5): 480-485.
DOI: 10.12677/pm.2018.85064

1. 引言

向量调度问题是经典的完工时间最小化问题的推广。一般来说，向量调度问题是将 n 个 d-维向量分配给 m 台机器以使得所有机器、所有维数的最大负载尽可能小。在文[1]中，由于 Graham 创造性的工作，单维负载均衡问题被广泛的研究。然而在许多问题中，一项任务可能需要不同的资源来完成，因此任务就无法用单一的量来度量。例如，如果一项任务同时需要 CPU 和内存，任务的需求最好用一个 2-维向量来刻画，向量的每个分量对应一种资源的数目。

Chekuri 和 Khanna [2]首次提出向量调度问题，算法调度 d-维工件给机器加工以使得所有机器、所有维数的最大负载尽可能小。当 d 是任意常数时，除非 $P=NP$，Chekuri 和 Khanna 证明向量调度问题不能在常数因子内近似。此外，他们[2]设计了一个 $O\left(\ln^2 d\right)$-近似算法。当 d 是固定常数时，他们提出了一个运行时间为 $\left(nd/\varepsilon\right)^{O\left(\left(\frac{\ln(d/\varepsilon)}{\varepsilon}\right)^d\right)}$ 的多项式时间近似方案(PTAS)。最近，Bansal 等人[3]在 $\exp\left(\left(1/\varepsilon\right)^{O(d\log\log d)}\right)+nd$ 时间内设计了一个有效的多项式时间近似方案(EPTAS)。

对于在线向量调度问题，Zhu 等人[4]提出了一个竞争比为 $O\left(\log dm\right)$ 的在线算法。Meyerson 等人[5]改进了文[4]中的算法，而且得到了一个 $O\left(\log d\right)$-近似算法。通过给出一个在线下界，Im 等人[6]证明 $\Theta\left(\log d/\log\log d\right)$ 是在线向量调度问题的最优竞争比。

对于经典的带等级约束的离线调度问题，当机器台数 m 为固定常数时，Hwang 等人在文[7]证明了 *LG-LPT* 算法的近似比：当 $m=2$ 时为 5/4，当 $m\geq 3$ 时为 $2-1/(m-1)$。在 2007 年，Glass 和 Kellerer 在文[8]对离线排序问题的特殊情形：当任务的加工时间只能是 1 或者 λ 时，证明了近似比趋于 $2-\lambda/(1+\lambda)$。当任务的加工时间无限制时，他们给出了一个 3/2-近似算法。Ou 等人在文[9]给出了一个 4/3-近似算法同时设计了一个多项式时间近似方案。Li 等人在文[10]中对 l_p 范数下具有等级约束的问题进行研究，并给出了一个基于线性规划解的全范数 2-近似算法。

近年来，许多研究人员开始关注云计算中的资源分配问题。Liu 等人在文[11]研究了异构物理机环境下的资源管理问题。云供应商给物理机提供异构资源，资源种类分别为 CPU，存储，硬盘大小等。在实际问题中，为了提高异构资源的利用效率，一些异构资源只能分配给一些特定的物理机。受 Liu 等人的启发，结合文[2]和[7]的创作思想，我们提出一种新的调度模型：带等级约束的多维任务调度问题。在这里，每个多维任务和每台机器都有一个等级，我们需要把每个多维任务调度到等级不高于它的机器上加工。目标是最小化所有机器、所有维数的最大负载。明显地，当 $d=1$ 时，带等级约束的多维任务调度问题就是经典的带等级约束的平行机调度问题。在本文，我们研究两台平行机上带等级约束的多维任务调度问题。为了便于分析，我们把这个问题简记为 $P_2\,|\,GoS,VC\,|\,C_{\max}$。

2. 预备知识

对于 $P_2\,|\,GoS,VC\,|\,C_{\max}$ 问题，给定实例 $I=(\mathrm{M},\mathrm{J},\mathrm{p},\mathrm{g})$。集合 $\mathrm{M}=\{M_1,M_2\}$ 包含两台平行机。集合 $\mathrm{J}=\{J_1,J_2,\cdots,J_n\}$ 包含 n 项任务。每一项任务 J_j 对应一个 d-维向量 $\mathrm{p}_j=(p_{j1},p_{j2},\cdots,p_{jd})$。在这里，我们要求 p_{jk} 是整数。$g(J_j)$ 是任务 J_j 的等级，机器 M_i 也有一个等级 $g(M_i)$。当 $g(J_j)\geq g(M_i)$ 时，任务 J_j 可以在机器 M_i 上加工。任务 J_j 到达后即可开始加工，加工过程不可中断而且每项任务只需加工一次。把 n 项任务分配给两台机器加工，我们可以得到任务集 J 的一个划分 (S_1,S_2)。在这里，我们令

$P_i=\max_k\sum_{J_j\in S_i}p_{jk}\left(i=1,2;k=1,\cdots,d\right)$。对于 $P_2\mid GOS,VC\mid C_{\max}$ 问题，我们的目标是使 $\max_i P_i$ 最小，也就是使所有机器、所有维数的最大负载尽可能小。

为了理解问题 $P_2\mid GOS,VC\mid C_{\max}$。我们给出一个例子：给定两台机器 M_1 和 M_2，它们的等级分别为 1 和 2。3 个任务 $J_1=(3,1,1)^{\mathrm{T}}$、$J_2=(3,2,5)^{\mathrm{T}}$、$J_3=(1,0,1)^{\mathrm{T}}$，工件的等级分别是 1，2，2。对于这个实例，最优的分配方案是把任务 J_1 和 J_3 分配给机器 M_1，任务 J_2 分配给机器 M_2，最优方案对应的最优值是 5。

在实例 I 的基础上，我们构造实例 $\hat{I}=(\mathrm{M},\mathrm{J},\hat{\mathrm{p}},\mathrm{g})$。实例 I 与实例 $\hat{I}$ 的差别是特征 p 与 $\hat{\mathrm{p}}$：在实例 I 中，任务 J_j 对应一个 d-维向量 $\mathrm{p}_j=\left(p_{j1},p_{j2},\cdots,p_{jd}\right)^{\mathrm{T}}$。在实例 $\hat{I}$ 中，任务 J_j 对应一个 1 维向量 $\hat{\mathrm{p}}_j=\sum_{k=1}^{d}p_{jk}$。对应于实例 I 的目标函数，因为实例 $\hat{I}$ 的每项任务对应一个 1 维向量，所以实例 $\hat{I}$ 的目标是使 $\max_i\left(\sum_{J_j\in S_i}\hat{\mathrm{p}}_j\right)\ (i=1,2)$ 最小。

3. $\frac{5d}{4}$-近似算法

对于实例 $\hat{I}=(\mathrm{M},\mathrm{J},\hat{\mathrm{p}},\mathrm{g})$，调用文[7]的 *LG-LPT* 算法，我们可以得到任务集 J 的一个调度方案 (S_1,S_2)。由实例 $\hat{I}$ 和实例 I 的关系，(S_1,S_2) 也是实例 $I=(\mathrm{M},\mathrm{J},\mathrm{p},\mathrm{g})$ 的一个调度方案。对于调度方案 (S_1,S_2)，我们假设实例 $\hat{I}$ 和实例 I 的目标值分别是 $OUT\left(\hat{I}\right)$ 和 $OUT(I)$。我们令 $OPT\left(\hat{I}\right)$ 和 $OPT(I)$ 分别表示实例 $\hat{I}$ 和实例 I 的最优值。

对与实例 I，我们设计了一个近似比为 $\frac{5}{4}d$ -近似算法。

算法 1：近似算法

步骤 1：根据实例 I，构造实例 $\hat{I}$；

步骤 2：对于实例 $\hat{I}$，调用 *LG-LPT* 算法，我们得到调度方案 (S_1,S_2)；

步骤 3：对于调度方案 (S_1,S_2)，实例 I 和实例 $\hat{I}$ 的目标值分别是 $OUT(I)$ 和 $OUT\left(\hat{I}\right)$；

步骤 4：返回 $OUT(I)$。

定理 1：对于问题 $P_2\mid GoS,VC\mid C_{\max}$，算法 1 是一个 $\frac{5}{4}d$ -近似算法。

证明：根据文[7]的结论可知，当机器台数 $m=2$ 时，对于实例 $\hat{I}$，我们有

$$OUT\left(\hat{I}\right)\le\frac{5}{4}OPT\left(\hat{I}\right) \tag{1}$$

对于某个 S_i，我们假设 $OUT(I)=\max_k\sum_{J_j\in S_i}p_{jk}$。根据实例 I 的目标函数值的定义，我们有 $OUT(I)\le\sum_{J_j\in S_i}\sum_{k=1}^{d}p_{jk}$。再根据实例 $\hat{I}$ 的目标函数值的定义，我们有 $OUT\left(\hat{I}\right)\ge\sum_{J_j\in S_i}\sum_{k=1}^{d}p_{jk}$。因此，我们有

$$OUT(I)\le OUT\left(\hat{I}\right) \tag{2}$$

我们用反证法证明 $OPT\left(\hat{I}\right)\le dOPT(I)$，假设 $OPT\left(\hat{I}\right)>dOPT(I)$。我们设 $\left(S_1^*,S_2^*\right)$ 是实例 I 的一个最优调度方案。根据实例 I 的目标函数值的定义，对于任意的 S_i^*，我们有 $dOPT(I)\ge\sum_{J_j\in S_i^*}\sum_{k=1}^{d}p_{jk}$。结合假设 $OPT\left(\hat{I}\right)>dOPT(I)$，我们有

$$OPT\left(\hat{I}\right)>dOPT(I)\ge\sum_{J_j\in S_i^*}\sum_{k=1}^{d}p_{jk}$$

上式说明，如果用调度方案 $\left(S_1^*,S_2^*\right)$ 求解实例 $\hat{I}$，我们可以得到一个比 $OPT\left(\hat{I}\right)$ 更小的目标值，这与

$OPT\left(\hat{I}\right)$的最优性矛盾，所以我们有

$$OPT\left(\hat{I}\right)\le dOPT\left(I\right) \tag{3}$$

联立(1)式、(2)式、(3)式，我们可以得到

$$OUT\left(I\right)\le\frac{5}{4}dOPT\left(I\right)$$

因此对于 $P_2\left|GoS,VC\right|C_{\max}$ 问题，算法 1 是一个 $\frac{5}{4}d$ -近似算法。

4. 动态规划算法

在这一部分，我们将给出问题 $P_2\left|GoS,VC\right|C_{\max}$ 的动态规划算法。对于问题 $P_2\left|GoS,VC\right|C_{\max}$，动态规划算法的设计策略是把 n 项任务所有可能的调度方案遍历一次，因此我们可以得到问题的最优解。

我们令 e_i 表示第 i 列元素全为 1，其余元素全为 0 的 $d\times 2$ 维矩阵。p_k 表示任务 J_k 对应的向量，$\mathrm{p}_k=\left(p_{k1},p_{k2},\cdots,p_{kd}\right)^{\mathrm{T}}$。$\mathrm{p}_k\otimes e_i$ 表示向量 p_k 与矩阵 e_i 的第 i 列元素对应相乘。

比如 $\mathrm{p}_1\otimes e_2=\begin{pmatrix}p_{11}\\p_{12}\\\vdots\\p_{1d}\end{pmatrix}\otimes\begin{pmatrix}0&1\\0&1\\\vdots&\vdots\\0&1\end{pmatrix}=\begin{pmatrix}0&p_{11}\\0&p_{12}\\\vdots&\vdots\\0&p_{1d}\end{pmatrix}$。

我们用空间 $\Phi_k\left(k=0,1,\cdots,n\right)$ 表示分配完前 k 个任务后所有可能的负载矩阵集，而且空间 Φ_k 可由空间 Φ_{k-1} 拓展得到。Φ_k 的元素都是 $d\times 2$ 维负载矩阵，负载矩阵的第 j 列,第 i 行表示第 j 台机器的第 i 个负载分量。$v_k=\max\left\{i\,|\,g\left(M_i\right)\le g\left(J_k\right)\right\}$ 表示能加工任务 J_k 的下标最大的机器的下标。

我们假设负载空间 $\Phi_{k-1}=\{A_1,A_2,\cdots,A_l\}$。

我们下面定义 $\Phi_k=\Phi_{k-1}+\left\{\mathrm{p}_k\otimes e_i\,|\,i=1,\cdots,v_k\right\}$。

如果 $v_k=1$，我们有 $\Phi_k=\Phi_{k-1}+\left\{\mathrm{p}_k\otimes e_i\,|\,i=1\right\}=\left\{A_1+\mathrm{p}_k\otimes e_1,A_2+\mathrm{p}_k\otimes e_1,\cdots,A_l+\mathrm{p}_k\otimes e_1\right\}$。

$$\Phi_k=\Phi_{k-1}+\left\{\mathrm{p}_k\otimes e_i\,|\,i=1,2\right\}$$

如果 $v_k=2$，我们有 $=\left\{A_1+\mathrm{p}_k\otimes e_1,A_2+\mathrm{p}_k\otimes e_1,\cdots,A_l+\mathrm{p}_k\otimes e_1,\right.$

$$\left.A_1+\mathrm{p}_k\otimes e_2,A_2+\mathrm{p}_k\otimes e_2,\cdots,A_l+\mathrm{p}_k\otimes e_2\right\}$$

如果 B 是一个负载矩阵，我们定义 B 的最大元素是 B 中所有元素的最大值。

例如负载矩阵 $B=\begin{pmatrix}b_{11}&b_{21}\\b_{12}&b_{21}\\\vdots&\vdots\\b_{1d}&b_{21}\end{pmatrix}$，则 B 的最大元素为 $\max\left\{b_{11},\cdots,b_{1d},b_{21},\cdots,b_{2d}\right\}$。

动态规划算法

步骤 1：我们将机器和任务都按等级从低到高的顺序重新标号，使得 $g\left(M_1\right)\le g\left(M_2\right)$ 且 $g\left(J_1\right)\le g\left(J_2\right)\le\cdots\le g\left(J_n\right)$；

步骤 2：令 $\Phi_0=\left\{\left(0\right)_{d\times 2}\right\}$；

步骤 3：对于 $j=1,\cdots,n$，做如下循环

$$\Phi_k=\Phi_{k-1}+\left\{\mathrm{p}_k\otimes e_i\,|\,i=1,\cdots,v_k\right\};$$

步骤 4：对于 Φ_n 中的每个矩阵，我们分别提取最大元素得到集合 $\alpha=\left\{c_1,c_2,\cdots,c_{|\Phi_n|}\right\}$

然后返回集合 α 中的最小值。

根据动态规划算法可知，负载空间 Φ_n 包含了所有可能的负载矩阵，自然也包含对应于最优解的负载矩阵，所有动态规划算法可以得到问题 $P_2|GoS,VC|C_{\max}$ 的最优解。

定理 2 当 d 为固定常数时，动态规划算法的运行时间为 $O\left(n^{2d+1}p_{\max}^{2d}\right)$，其中 $p_{\max}=\max_{j,k}p_{jk}\left(j=1,\cdots,n;k=1,\cdots,d\right)$。

证明：由于 $p_{\max}=\max_{j,k}p_{jk}$，则任意机器、任意维数的负载不超过 $np_{\max}$。当任务 J_j 被调度后，因为负载矩阵中的每个元素在集合 $\{0,1,\cdots,np_{\max}\}$ 内取值，所以我们可以在 $O\left(1+np_{\max}\right)^{2d}$ 时间内得到当前的负载矩阵。又因为一共有 n 项任务，所以动态规划算法的运行时间为 $O\left(n\left(1+np_{\max}\right)^{2d}\right)=O\left(n^{2d+1}p_{\max}^{2d}\right)$。

5. 完全多项式时间近似方案

在这一部分，对于问题 $P_2|GoS,VC|C_{\max}$，当任务维数 d 为固定常数时，我们用动态规划算法设计了一个完全多项式时间近似方案。

定理 3 当 d 是固定常数时，问题 $P_2|GoS,VC|C_{\max}$ 在 $O\left(n^{2d+1}\left(\frac{d}{\varepsilon}\right)^{2d}\right)$ 时间内存在一个完全多项式时间近似方案。

证明：我们用 L 表示算法 1 求解实例 I 得到的目标值。根据定理 1，我们有

$$OPT(I)\le L\le\frac{5}{4}dOPT(I)$$

在实例 $I=(\mathrm{M},\mathrm{J},\mathrm{p},\mathrm{g})$ 的基础上，我们构造实例 $I'=(\mathrm{M},\mathrm{J},\mathrm{p}',\mathrm{g})$。实例 I 与实例 I' 的差别是特征 p 与 p′：在实例 I 中，任务 J_j 对应一个 d-维向量 $\mathrm{p}_j=\left(p_{j1},p_{j2},\cdots,p_{jd}\right)^{\mathrm{T}}$。在实例 I' 中，任务 J_j 对应一个 d-维向量 $\mathrm{p}'_j=\left(p'_{j1},p'_{j2},\cdots,p'_{jd}\right)^{\mathrm{T}}$。在这里，我们有 $p'_{jk}=\left\lfloor\frac{p_{jk}}{4\varepsilon L/5dn}\right\rfloor\cdot\frac{4\varepsilon L}{5dn}\left(k=1,2,\cdots,d\right)$。因为 $p'_{jk}=\left\lfloor\frac{p_{jk}}{4\varepsilon L/5dn}\right\rfloor\cdot\frac{4\varepsilon L}{5dn}$，所以 $p'_{jk}\le p_{jk}\le p'_{jk}+\frac{4\varepsilon L}{5dn}$。因为动态规划可以得到实例 $I'=(\mathrm{M},\mathrm{J},\mathrm{p}',\mathrm{g})$ 的最优方案 $\left(S'_1,S'_2\right)$，我们假设最优值 $OPT(I')=\max_{i,k}\sum_{J_j\in S'_i}p'_{jk}\left(i=1,2;k=1,2,\cdots,d\right)$。假设实例 I 的最优方案是 $\left(O_1,O_2\right)$，则实例 I 的最优值 $OPT(I)=\max_{i,k}\sum_{J_j\in O_i}p_{jk}\left(i=1,2;k=1,2,\cdots,d\right)$。

用实例 I' 的最优调度方案 $\left(S'_1,S'_2\right)$ 求解实例 I，我们可以得到下面的链式不等式

$$\max_{i,k}\sum_{J_j\in S'_i}p_{jk}\le\max_{i,k}\sum_{J_j\in S'_i}\left(p'_{jk}+\frac{4\varepsilon L}{5dn}\right)=\max_{i,k}\sum_{J_j\in S'_i}p'_{jk}+\left|S'_i\right|\frac{4\varepsilon L}{5dn}\le\max_{i,k}\sum_{J_j\in S'_i}p'_{jk}+\frac{4\varepsilon L}{5d}\le\max_{i,k}\sum_{J_j\in O_i}p'_{jk}+\frac{4\varepsilon L}{5d}$$

$$\le\max_{i,k}\sum_{J_j\in O_i}p_{jk}+\frac{4\varepsilon L}{5d}\le OPT(I)+\varepsilon OPT(I)=(1+\varepsilon)OPT(I)$$

当动态规划算法求解实例 I' 时，由于 $OPT(I')\le OPT(I)\le L$，我们只需要考虑负载不超过 L 的调度方案。因为任意机器的负载分量都是 $\frac{4\varepsilon L}{5dn}$ 的整数倍，而且一共有 n 项任务，因此实例 I' 可以在 $O\left(n\left(\frac{5dn}{4\varepsilon}+1\right)^{2d}\right)=O\left(n^{2d+1}\left(\frac{d}{\varepsilon}\right)^{2d}\right)$ 时间内得到最优方案。

6. 结论

在本文，对于两台平行机上的 $P_2|GoS,VC|C_{\max}$ 问题，我们首先设计了一个 5 d/4-近似算法。然后，

我们给出了一个求最优解的动态规划算法和一个完全时间多项式近似方案。对于带等级约束的多维向量调度问题，当机器台数为$m(m \geq 3)$时，我们以后将设计一个近似算法而且给出一个完全多项式时间近似方案。

参考文献

[1] Graham, R.L., *et al*. (1979) Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey. *Annals of Discrete Mathematics*, **5**, 287-326. https://doi.org/10.1016/S0167-5060(08)70356-X

[2] Chekuri, C. and Khanna, S. (2004) On Multidimensional Packing Problems. *SIAM Journal on Computing*, **33**, 837-851. https://doi.org/10.1137/S0097539799356265

[3] Bansal, N., Vredeveld, T. and Zwaan, R.V.D. (2014) Approximating Vector Scheduling: Almost Matching Upper and Lower Bounds. *LATIN* 2014: *Theoretical Informatics*, Springer, Berlin Heidelberg.

[4] Zhu, X., Li, Q., Mao, W., *et al*. (2014) Online Vector Scheduling and Generalized Load Balancing. *Journal of Parallel & Distributed Computing*, **74**, 2304-2309. https://doi.org/10.1016/j.jpdc.2013.12.006

[5] Meyerson, A., Roytman, A. and Tagiku, B. (2013) Online Multidimensional Load Balancing. Approximation, Randomization, and Combinatorial Optimization. *Algorithms and Techniques*, Springer, Berlin, Heidelberg, 287-302.

[6] Im, S., Kell, N., Kulkarni, J., *et al*. (2014) Tight Bounds for Online Vector Scheduling. 525-544.

[7] Hwang, H.C., Chang, S.Y. and Lee, K. (2004) Parallel Machine Scheduling under a Grade of Service Provision. *Computers & Operations Research*, **31**, 2055-2061. https://doi.org/10.1016/S0305-0548(03)00164-3

[8] Glass, C.A. and Kellerer, H. (2007) Parallel Machine Scheduling with Job Assignment Restrictions. *Naval Research Logistics* (*NRL*), **54**, 250-257. https://doi.org/10.1002/nav.20202

[9] Ou, J., Leung, J.Y.T. and Li, C.L. (2008) Scheduling Parallel Machines with Inclusive Processing Set Restrictions. *Naval Research Logistics* (*NRL*), **55**, 328-338. https://doi.org/10.1002/nav.20286

[10] Li, W., Li, J. and Zhang, T. (2012) Two Approximation Schemes for Scheduling on Parallel Machines under a Grade of Service Provision. *Asia-Pacific Journal of Operational Research*, **29**, 1250029. https://doi.org/10.1142/S0217595912500297

[11] Liu, X., Li, W. and Zhang, X. (2018) Strategy-Proof Mechanism for Provisioning and Allocation Virtual Machines in Heterogeneous Clouds. *IEEE Transactions on Parallel & Distributed Systems*, **99**, 1650-1663.