连续单调可塑作业调度的实用型73/50近似算法

Klaus Jansen ✉基尔大学，德国

FelixOhnesorge ✉基尔大学，德国

arXiv:2601.02836v1 [cs.DS] 6 Jan 2026

摘要

在可塑作业调度问题中，给定 m 台同型机器和 n 个作业，每个作业可在数量可变的机器上执行。每个作业的执行时间取决于分配给该作业的机器数量。对于单调可塑作业调度这一特定问题，假设作业的处理时间随机器数量增加而非递增。

此前已知的最佳算法包括：(1) 一种时间复杂度为 $\Omega(n^{g(1/\varepsilon)})$ 的多项式时间近似方案，其中 $g(\cdot)$ 是一个超指数函数 [Jansen and Thöle '08；Jansen 和 Land '18]；(2) 针对 $m \geq 8\frac{n}{\varepsilon}$ [的全多项式近似方案，见 Jansen 和 Land '18]；以及 (3) 一种 $\frac{3}{2}$ 近似算法，时间复杂度为 $O(nm\log(mn))$ [Wu、Zhang 和 Chen '23]。

我们提出了一种新的、实际高效的算法，其近似比为 $\approx(1.4593+\varepsilon)$，时间复杂度为 $O(nm\log\frac{1}{\varepsilon})$。我们的结果也适用于该问题的连续变体。除了理论结果外，我们还实现了所提出的算法，并表明其实际性能显著优于理论最坏情况近似比。

2012年 ACM 学科分类计算理论 → 调度算法；计算理论 → 离散优化；计算理论 → 并行计算模型

关键词和短语计算、机器调度、可塑性、多项式近似

补充材料软件：`https://github.com/Felioh/MoldableJobScheduling`

资助 由德国研究联合会（DFG，德国研究基金会）资助——项目编号 453769249

1 引言

过去，人们研究了许多种调度问题。经典的任务调度问题是将 n 作业调度到 m 同型机器上，其中每个作业j 恰好占用 1 台机器。受许多实际应用的启发，例如高性能计算 [8]，利用并行性来加快作业的执行速度，人们研究了这一问题的许多扩展形式。其中一种是调度可塑型作业。这是一种自然的扩展，允许作业在数量可变的机器上执行。正如 [2, 8, 11] 中所讨论的，这种问题形式有许多实际应用动机。

在可塑作业调度问题中，给定 $J = \{1, \dots, n\}$ 个作业和 m 台同型机器。每个作业可分配到 $k \leq m$ 台机器上，任一作业 j 的处理时间取决于分配给它的机器数量。我们将任一作业 j 的处理时间记为 $t(j, k)$。此外，任一作业 j 的工作量 定义为 $w(j, k) = t(j, k) \cdot k$，可将其描述为作业的面积。对于单调的 可塑作业调度，我们假设对任意作业，其工作量函数和时间函数分别非递减和非递增。

▶ **假设 1.1** (单调性 [29])。 给定任意作业 j，对所有 $k \leq k' \leq m$，有：**(i)** $w(j, k) \leq w(j, k')$ **(ii)** $t(j, k) \geq t(j, k')$

该问题一个实例的解由一个调度构成，其中包含每个作业的开始时间s_j，以及每个作业的分配量 $\alpha(j) \in \{1, \dots, m\}$。若满足以下条件，则称该调度可行：

- 作业在其所有分配的机器上同时开始执行。作业在执行时间内不得中
- 断。每台机器同一时间最多只能执行一个作业。

在该问题的连续 变体中，我们还要求每个作业必须在一个相邻机器集合上处理。在许多实际应用中，采用这种调度方式，让作业在相邻机器上处理很有益处，因为这样可以改善内存局部性，并减少机器之间的通信开销 [2]。正如 [1, 5, 6, 13, 29, 31]中所讨论的，这一模型还有许多其他应用。本文研究的是如何最小化完工时间，即调度中任意作业的最大完工时间。

1.1 相关工作

请注意，在可塑作业调度中，分配给作业的机器数量在作业执行期间是固定的。一些较早的出版物将此问题称为可变作业调度 [4, 12, 13, 21, 24, 27, 29]。不过，最近这一术语已改为可塑作业调度，而可变作业调度现在用于指代另一类问题：分配给作业的机器数量可以在执行过程中发生变化。

已知可塑作业调度及其单调变体都是 NP难问题 [9, 20]。Turek、Wolf 和 Yu [30] 提出了一种适用于可塑作业调度一般情形的 2-近似算法。之后，Ludwig 和 Tiwari [27] 提出了一种近似比相同、但运行时间改进为关于 $\log m$ 多项式的算法。已有证明表明，对于可塑作业调度（不要求单调性！），除非 P = NP [7, 25]，否则不存在保证值低于 $3/2$ 的多项式时间近似算法。不过，对于连续可塑作业调度，存在一种伪多项式（$\frac{5}{4} + \varepsilon$）-近似算法，其时间复杂度为 $O(n \log n) \cdot m^{f(1/\varepsilon)}$，其中 $f(\cdot)$ 是一个可计算函数 [22]。对于该问题的一些特殊情形，存在更好的近似算法：

Jansen 和 Porkolab [21] 针对机器数量为常数的情形 m提出了一种运行时间为线性的近似方案；Decker、Lücking 和 Monien [4] 在额外假设作业相同的条件下，给出了一种 1.25-近似算法。如果所有作业在任意数量机器上的执行时间都相同，则称这些作业相同。

在本文中，我们研究了单调可塑作业调度这一具体问题。Mounié、Rapine 和 Trystram [28, 29] 针对连续情形提出了一种 $(\frac{3}{2}+\varepsilon)$-近似算法，其时间复杂度为 $O(nm \log \frac{1}{\epsilon})$。过去数年中，许多作者改进了这一结果。Jansen 和 Thöle [23, 24] 证明了存在一种近似比任意接近 1.5 的算法，用于连续问题；当 m 受 n的多项式界约束时，他们还针对不受连续性限制的同一问题提出了一个多项式时间近似方案（PTAS）。Jansen 和 Land [19] 通过在 [24]中提出一个全多项式时间近似方案（FPTAS），将该 PTAS 推广到了满足 $m > 8\frac{n}{\varepsilon}$ 假设的连续情形。此外，他们改进了 [29] 中的 $(\frac{3}{2}+\varepsilon)$-近似算法，使其运行时间仅对机器数量具有多对数依赖 m。这些结果随后又被 Grage、Jansen 和 Ohnesorge [16]进一步改进。就在最近，Wu、Zhang 和 Chen [31]提出了一种$\frac{3}{2}$-近似算法，其运行时间为 $O(nm \log nm)$，并提出了一个开放问题：对于该问题，是否存在近似比优于$\frac{3}{2}$且具有类似运行时间的近似算法。

比率	复杂度	作者（年份）
$\frac{3}{2}+\epsilon$	$O(\log(1/\varepsilon)nm)$	Mounié, Rapine, Trystram（2007） [29]
$\frac{3}{2}+\epsilon$	$O(\frac{n}{\varepsilon^2}$对数 $m(\frac{\log m}{\epsilon}+$对数$^3(\varepsilon m)))$	Jansen, Land（2018年） [19]
$\frac{3}{2}+\epsilon$	$O(n\log^2(\frac{1+\log(\varepsilon m)}{\varepsilon})+\frac{n}{\varepsilon}\log(\frac{1}{\varepsilon})\log(\varepsilon m))$	Grage, Jansen, Ohnesorge（2023年） [16]
$\frac{3}{2}$	$O(nm\log(nm))$	Wu、Zhang、Chen（2023年） [31]
$1.4593+\epsilon$	$O(\log(1/\varepsilon)nm)$	本文工作

表1 关于单调可塑作业调度的多项式时间近似算法概述，其中 n、 m 和 $1/\varepsilon$。

1.2 我们的贡献

对于单调可塑作业调度问题（非连续），目前已知的最佳近似比为 $1+\varepsilon$，由 [23, 24] 中的多项式时间近似方案给出。该多项式时间近似方案使用动态规划，将作业安排在 δ^{-2} 个起始点上。令 $\varepsilon := 0.5$ 时，我们通过计算 $\delta := \sigma_{36}$ 得到起始点的数量，使得 $\sigma_1 := \frac{1}{18}$ 且 $\sigma_k := {}^{\sigma_{k-1}^3}(4\cdot 7^2)$。这大约可化简为 $\sigma_{36} \approx 10^{-1.2\cdot 10^{17}}$。随后，整体动态规划的运行时间约为 $nm^{2\cdot 10^{2.4\cdot 10^{17}}}$。将 [22], 中的 $(\frac{5}{4}+\varepsilon)$-近似算法与当前的$\frac{3}{2}$-近似算法进行比较，并令 $\varepsilon := \frac{1}{4}$，可得运行时间为 $\Omega((mn)^{16^{4^{13}}})$。虽然这些算法具有理论意义，但其运行时间在实践中难以接受。受此启发，我们专注于实际高效的近似算法（表1）。

此前，人们认为，对于单调可塑作业调度问题，任意 $\varepsilon > 0$ [31] 都不存在实用且高效的 $(\frac{3}{2}-\varepsilon)$-近似算法。我们打破了这一障碍，提出了一种最坏情况近似比为

$\approx 1.4593 < 1.5$ 的近似算法。值得注意的是，[24] 中提出的多项式时间近似方案无法解决该问题的连续变体，而且目前尚不清楚该问题的这一变体是否存在多项式时间近似方案。令人惊讶的是，我们证明该算法生成的连续调度的完工时间至多为 $1.4593 \cdot OPT$，其中 OPT 是非连续问题最优值的下界。这一结果也界定了该问题连续变体与非连续变体之间的差距。

我们的算法运行时间为 $O(mn \log \frac{1}{\varepsilon})$。由于对 m 的线性依赖，从严格意义上说，该算法对于问题的编码长度而言仅为伪多项式时间。与 [22, 24], 中的算法类似，对于满足 $m > 8\frac{n}{\varepsilon}$ 的实例，我们可以使用 [16, 19] 中提出的全多项式时间近似方案。因此，我们可以假设 $m \leq 8\frac{n}{\varepsilon}$。在这种情况下，我们的运行时间 $O(mn \log \frac{1}{\varepsilon})$ 是全多项式时间。

▶ **定理 1.** 设 $\varepsilon > 0$。对于连续单调可塑作业调度，存在一种算法，其近似比为 $(1.4593 + \varepsilon)$，运行时间为 $O(nm \log \frac{1}{\varepsilon})$。

我们通过改进 Mounié、Rapine 和 Trystram [29] 长期以来的算法，取得了这一结果。尽管该算法的运行时间 [19, 16], 多次得到改进，但提升近似比似乎一直是一项艰巨的任务（除去消除 ε 因子 [31]）。我们通过以下新技术实现了这一目标：(1) 将其算法中使用的标准 0/1 背包问题放宽为多重选择背包问题；(2) 将作业装入更多容器中，这使得修复算法更加复杂；(3) 运用优雅的几何论证，仔细分析一个关键情形。我们分析这一关键情形的方法，使得所谓的朗伯 W 函数出现在近似比中。

我们认为，该结果具有理论意义，因为它打破了这些实际高效算法长期以来的$\frac{3}{2}$ 障碍；所采用的技术，尤其是几何论证，也可能具有更广泛的意义。类似的几何分析或许有益于其他调度和装箱问题 [10, 14, 17]。此外，我们的理论结果得到了实际实验的支持，实验表明，在大多数情况下，可以保证调度长度至多为 $\frac{10}{7}OPT$。这使得该算法成为实际应用中很有前景的候选方案。

1.3 预备知识

与许多先前的工作一样，我们采用 Hochbaum 和 Shmoys [18]提出的对偶近似框架。首先，我们通过常数近似算法 [30, 27] 获得最优完工时间的下界和上界，然后在该区间内进行二分搜索，以在误差为 ε的精度下找到最优完工时间。因此，在本文中，我们假设给定一个完工时间猜测值 d，并希望找到一个完工时间至多为 λd的调度方案；如果 $d < OPT$，则拒绝 d。

与 Mounié、Rapine 和 Trystram [29], 类似，我们将规范机器数定义如下：

▶**定义 2** (规范机器数)。给定一个实数 h，我们将每个作业 j 的规范机器数 $\gamma(j, h)$ 定义为执行 j 所需的最少机器数，使其完成时间不超过 h。如果 j 无法在 m 台机器上于不超过 h 的时间内处理完成，则按约定设 $\gamma(j, h) = \infty$。

需要指出的是，由于单调性，可以通过二分搜索在 $O(\log m)$ 时间内找到这个数。

2 算法描述

为了阐明我们算法的核心思想，我们先简要概述 Mounié、Rapine 和 Trystram [29] 提出的算法：首先，找到一个具有某些特定性质的划分$\mathcal{S}_1 \sqcup \mathcal{S}_2 = J_B = \{j \in J\colon t(j,\ 1) > \frac{d}{2}\}$。然后，调度第一组$\mathcal{S}_1$，其总宽度和高度分别为 m 和 d。第二组$\mathcal{S}_2$ 则安排在其上方，高度至多为 $\frac{d}{2}$，从而总高度至多为$\frac{3}{2}d$。之后，对剩余的小作业 $J_S = \{j \in J\colon t(j,\ 1) \leq \frac{d}{2}\}$ 进行贪心调度。为便于表述，我们将仿照 [29, 21, 16], 分别把集合$\mathcal{S}_1$,$\mathcal{S}_2$ 第 *1* 层货架 和第 *2* 层货架称为货架 1 和货架 2。我们将每个货架中作业允许的最大执行时间称为该货架的高度；例如，在 [29] 的算法中，货架 1 的高度为 d，货架 2 的高度为 $\frac{d}{2}$。

尽管该算法针对近似比 $\frac{3}{2}$ 进行了高度定制，我们仍设法对其进行调整，以获得更好的近似比。一个关键见解是，我们可以将作业划分为更多集合，从而更贴近最优调度中的分配量。我们相信，这项技术可能具有更广泛的意义。改进该算法的近似比面临几个主要挑战：(1) 以贪心方式添加作业 $j \in J_S$with $t(j,1) \leq \frac{d}{2}$ will result in an approximation ratio $\geq 3/2$。(2) 我们需要更加谨慎地将作业划分到货架中。

我们通过定义 $J_S = \{j \in J\colon t(j,1) \leq \frac{3}{7}d\}$ 作为小作业集合，并将 $J_B = J \setminus J_S$ 作为大作业集合，克服了第一个挑战。这一选择看似武断，但在后续分析中将会变得清晰。任意调度中小作业的最小工作量记为 $\mathcal{W}_S = \sum_{j \in J_S} w(j,1)$。

对于该划分，我们允许使用第三个货架，其宽度可以超过 m。利用作业的工作量单调性，我们之后可以减少分配给该货架中作业的机器数量。这引出了我们工作的核心定理：

▶ 定理 *3.* 给定作业的划分 $C_1 \sqcup C_2 \sqcup C_3 = J_B$，使得 $C1\ \sum_{j \in C_1} w(j, \gamma(j,d)) + \sum_{j \in C_2} w\left(j, \gamma\left(j, \frac{4}{7}d\right)\right) + \sum_{j \in C_3} w\left(j, \gamma\left(j, \frac{3}{7}d\right)\right) \leq md - W_S C2\ \sum_{j \in C_1} \gamma(j,d) + \sum_{j \in C_2} \gamma\left(j, \frac{4}{7}d\right) \cdot \frac{1}{2} \leq m,$ 对于任意$\lambda > -\frac{1}{3}W_{-1}(-\frac{3}{e^4}) \approx 1.4593,$ 可以在时间 $O(mn)$ 内计算出一个完工时间至多为 λd 的连续调度。

朗伯 W 函数也称为欧米伽函数或乘积对数，其定义使得 $ye^y = x$ 成立，当且仅当 $y = W(x)$。对于 $-\frac{1}{e} \leq x < 0$ [3]，使用 W_{-1} 分支。朗伯 W 函数之所以会出现，是由我们的分析方法决定的：我们通过定义积分来获得调度中作业工作量的上界。虽然这种技术并非新方法，但在调度问题的分析中并不常用。Fotakis、Matuschke 和 Papadigenopoulos [12] 采用了类似的分析技术，但他们的分析中没有出现朗伯 W 函数。

2.1 寻找一个划分

首先，我们证明：给定一个完工时间至多为 d exists 的调度，就存在一个将作业划分为三个类别的划分，如定理 3 所要求。

▶**引理 4.** 给定任意一个完工时间至多为 d 的调度，都存在一个划分 $C_1 \sqcup C_2 \sqcup C_3 = J_B$，满足定理 *3* 的约束 *C1* 和 *C2*。

证明。 考虑实例 I 的任意调度，其完工时间为 $d^* \leq d$。所有小作业的总工作量定义为 $J_S = \{j \in J\colon t(j,1) \leq \frac{3}{7}d\}$，由于工作量单调性，该值至少为 $\mathcal{W}_S = \sum_{j \in J_S} t(j,1)$,

根据工作量单调性的假设，所有其余作业的工作量至多为 $W = md - W_S$。对于任意作业 $j \in J_B$，我们将最优调度中分配给该作业的机器数记为 $OPT(j)$。现在我们说明，J_B 中作业的划分 $C_1 := \{j \in J_B : t(j, OPT(j)) > \frac{4}{7}d\}$、$C_2 := \{j \in J_B : \frac{4}{7}d \geq t(j, OPT(j)) > \frac{3}{7}d\}$ 和 $C_3 := \{j \in J_B : \frac{3}{7}d \geq t(j, OPT(j))\}$ 同时满足约束 C1 和 C2。

由于单调性，我们知道，$\mathcal{C}_3$ 中的任意作业的工作量至少为 $w(j, \gamma(j, \frac{3}{7}d))$。因此：

$$\sum_{j \in \mathcal{C}_3} w\left(j, \gamma\left(j, \frac{3}{7}d\right)\right) \leq \sum_{j \in \mathcal{C}_3} w(j, OPT(j)) \tag{1}$$

$\mathcal{C}_1$ 中的每个作业至少使用 $\gamma(j, d)$ 台机器，且工作量至少为 $w(j, \gamma(j, d))$，因此：

$$\sum_{j \in \mathcal{C}_1} w(j, \gamma(j, d)) \leq \sum_{j \in \mathcal{C}_1} w(j, OPT(j)) \tag{2}$$

$$\sum_{j \in \mathcal{C}_1} \gamma(j, d) \leq \sum_{j \in \mathcal{C}_1} OPT(j) \tag{3}$$

对于最后一个划分 $\mathcal{C}_2$，，首先注意到，在高度至多为 d 的任意最优调度中，来自 $\mathcal{C}_2$ 的作业都不能与来自 $\mathcal{C}_1$ 的作业安排在同一台机器上。因此，$\mathcal{C}_2$ 中的所有作业最多安排在 $m - \sum_{j \in \mathcal{C}_1} OPT(j)$ 台机器上。每个作业 $j \in \mathcal{C}_2$ 至少使用 $\gamma(j, \frac{4}{7}d)$ 台机器，并且由于 $t(j, OPT(j)) > \frac{3}{7}d$，同一台机器上最多可以安排两个这样的作业。因此：

$$\sum_{j \in \mathcal{C}_2} \gamma\left(j, \frac{4}{7}d\right) \cdot \frac{1}{2} \leq m - \sum_{j \in \mathcal{C}_1} OPT(j) \tag{4}$$

对于工作面积，我们给出与上文类似的论证，得到：

$$\sum_{j \in \mathcal{C}_2} w\left(j, \gamma\left(j, \frac{4}{7}d\right)\right) \leq \sum_{j \in \mathcal{C}_2} w(j, OPT(j)) \tag{5}$$

结合方程 (3) 和 (4)，可得约束 C2；结合方程 (1)、(2) 和 (5)，可得约束 C1。由此得证该引理。◀

因此，给定一个完工时间猜测 $d \geq \mathrm{OPT}$，我们知道存在一个作业划分 $\mathcal{C}_1 \sqcup \mathcal{C}_2 \sqcup \mathcal{C}_3 = J_B$，使得定理 3 的约束成立。找到这样的划分等价于求解一个多重选择背包问题 (MCKP)。

▶ **定义 5** (多重选择背包问题 (MCKP))。给定容量 $m \in \mathbb{N}$、$k \in \mathbb{N}$ 个类别和 $n \in \mathbb{N}$ 个物品，每个物品 $j \in [n]$ 都有一个成本 c_{jl} 和一个大小 s_{jl}，适用于每个 $l \in [k]$。求一个划分 $\mathcal{C}_1 \sqcup \cdots \sqcup \mathcal{C}_k = [n]$，满足：

$$\min \sum_{l=1}^{k} \sum_{j \in \mathcal{C}_l} c_{jl}$$

$$s.t. \sum_{l=1}^{k} \sum_{j \in \mathcal{C}_l} s_{jl} \leq m$$

使用动态规划 [26]可以在时间 $O(mn)$ 内求解 MCKP。给定一个完工时间猜测值 $d \geq OPT$，寻找满足定理 3 约束的划分等价于

求解一个容量为 m 且有 $k=3$ 个类别的 MCKP，其中每个作业 $j \in J_B$ 都是一个物品，其成本为 $c_{j1} = w(j, \gamma(j, d))$、$c_{j2} = w(j, \gamma(j, \frac{4}{7}d))$ 和 $c_{j3} = w(j, \gamma(j, \frac{3}{7}d))$，大小为 $s_{j1} = \gamma(j, d)$、$s_{j2} = \frac{1}{2}\gamma(j, \frac{4}{7}d)$ 和 $s_{j3} = 0$。因此，在本文余下部分中，我们假设给定了一个满足定理 3 约束的划分 $\mathcal{C}_1 \sqcup \mathcal{C}_2 \sqcup \mathcal{C}_3 = J_B$。

2.2 构造调度

本节假设给定一个完工时间猜测 $d \geq OPT$ 和一个划分$\mathcal{C}_1 \sqcup \mathcal{C}_2 \sqcup \mathcal{C}_3 = J_B$，且该划分满足定理 3 的约束。我们将通过给出一个具有所需近似比的算法来证明定理 3。算法 1 概述了该算法。

算法 1 对偶近似算法说明

要求： m 台同型机器、n 个可塑作业，以及一个完工时间猜测 d

1: 令 $J_S = \{j \in J | t(j, 1) \leq \frac{3}{7}d\}$ 并令 $J_B = J \setminus J_S$。
2: 通过 MCKP 构造划分 $\mathcal{C}_1 \sqcup \mathcal{C}_2 \sqcup \mathcal{C}_3 = J_B$。 ▷ 引理 4
3: **如果** $W(\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3) > dm - \mathcal{W}_S$，**则**
4: **拒绝** d。
5: **结束 if**
6: 根据引理 6 将作业分配到货架 $\mathcal{C}_1$,$\mathcal{C}_2$,$\mathcal{C}_3$ 上。 ▷ 定义 8
7: 应用算法 3。
8: **如果** $q < m/6$ **则**
9: 应用算法 4。 ▷ 引理 10
10: **否则**
11: 应用算法 2。 ▷ 引理 12
12: **结束 if**
13: 通过算法 5 贪心地添加小作业 J_S。 ▷引理30

现在，我们的目标是为所有作业 $J_B = J \setminus J_S$构造一个中间调度，如图1（左）所示。任何作业 $j \in \mathcal{C}_1$ 均在时间 0 分配到 $\gamma(j, d)$ 台机器上，任何作业 $j \in \mathcal{C}_2$均在时间 0 分配到 $\gamma(j, \frac{4}{7}d)$ 台机器上，而任何作业 $j \in \mathcal{C}_3$则分配到 $\gamma(j, \frac{3}{7}d)$ 台机器上，使其在时间 $\frac{10}{7}d$ 完成。分析所得调度时，我们注意到：（1）$\mathcal{C}_1$ 和$\mathcal{C}_2$ 中的作业需要超过 m 台机器；（2）$\mathcal{C}_3$ 中的作业也可能需要超过 m 台机器。

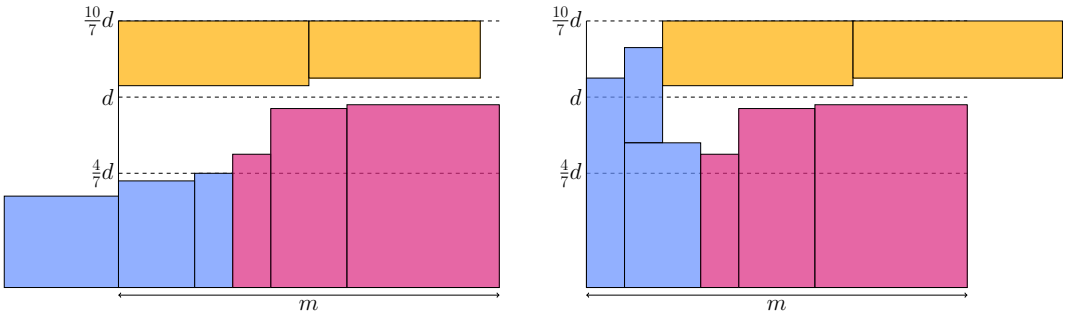

图1 引理 6 之前（左）和之后（右）的调度。粉色作业位于 $\mathcal{C}_1$,，蓝色作业位于 $\mathcal{C}_2$,，黄色作业位于$\mathcal{C}_3$。

8 可塑作业调度

我们通过将作业 $j \in \mathcal{C}_2$ 安排在更少的机器上来解决第一个问题，并在第 2.3 节中关注第二个问题。请注意，所有作业的调度方式均确保约束 C1继续得到满足，且所有大作业的总工作量至多为 $md - \mathcal{W}_S$。由于工作量单调性，只要确保对该调度的任何修改都不会增加分配给任何作业的机器数量，这一点在本文余下部分仍然成立。

▶ **引理 6.** $\mathcal{C}_2$ 中的所有作业都可以安排在 $m - \sum_{j \in \mathcal{C}_1} \gamma(j, d)$ 台机器上，且完工时间不超过 $\frac{10}{7}d$。

证明。首先注意到，根据约束 C2，我们得到 $\sum_{j \in \mathcal{C}_2} \gamma(j, \frac{4}{7}d) \cdot \frac{1}{2} \leq m - \sum_{j \in \mathcal{C}_1} \gamma(j, d)$。因此，我们需要有效地调度每个作业，使其至多需要 $\gamma(j, \frac{4}{7}d) \cdot 1\,2$台机器。

我们根据作业的规范机器数 $\gamma(j, \frac{4}{7}d)$，对作业 $j \in \mathcal{C}_2$ 进行分析。

情形 1 ($\gamma(j, \frac{4}{7}d) \geq 4$)：这些作业可以通过这样的方式进行调度：分配到 $\lfloor \gamma(j, \frac{4}{7}d)/2 \rfloor$ 台机器时，执行时间不会超过 $\frac{10}{7}d$。由于工作量单调性，我们有：

$$
\begin{aligned}
& w\left(j, \gamma\left(j, \frac{4}{7}d\right)\right) \geq w\left(j, \left\lfloor \frac{\gamma\left(j, \frac{4}{7}d\right)}{2} \right\rfloor\right) \\
\Leftrightarrow\ & t\left(j, \gamma\left(j, \frac{4}{7}d\right)\right) \cdot \gamma\left(j, \frac{4}{7}d\right) \geq t\left(j, \left\lfloor \frac{\gamma(j, \frac{4}{7}d)}{2} \right\rfloor\right) \cdot \left\lfloor \frac{\gamma(j, \frac{4}{7}d)}{2} \right\rfloor \\
\Rightarrow\ & \frac{4}{7}d \cdot \frac{\gamma(j, \frac{4}{7}d)}{\left\lfloor \frac{\gamma(j, \frac{4}{7}d)}{2} \right\rfloor} \geq t\left(j, \left\lfloor \frac{\gamma(j, \frac{4}{7}d)}{2} \right\rfloor\right)
\end{aligned}
$$

这意味着，对于任意 $\gamma(j, \frac{4}{7}d) \geq 4$，$t\left(j, \lfloor \gamma(j, \frac{4}{7}d)/2 \rfloor\right)$ 小于或等于 $\frac{10}{7}d$。

情形 2（$\gamma(j, \frac{4}{7}d) = 2$）：这些作业可以安排在 1 台机器上调度，且完工时间不超过 $\frac{10}{7}d$，从而使所需机器数量恰好减半。

情形 3（$\gamma(j, \frac{4}{7}d) = 3$）：如果有多个这样的作业，可以将它们两两叠加调度，且不会产生空闲时间。在本文余下部分中，我们可以将这些作业视为一个更大的作业。因此，我们可以假设此类作业至多有一个，并将其记为 j_3。

情形 4（$\gamma(j, \frac{4}{7}d) = 1$）：这些作业也可以两两叠加调度。在本文余下部分中，我们可以将这些作业视为一个更大的作业。剩下的单个作业（记为 j_1）可能需要采用不同的处理方式。

如果 j_1 和 j_3 都不存在，那么我们就成功地调度了所有作业，使它们在$\mathcal{C}_2$ 中所需的机器不超过 $m - \sum_{j \in \mathcal{C}_1} \gamma(j, d)$ 台。

如果只有 j_1 存在，由于 $m - \sum_{j \in \mathcal{C}_1} \gamma(j, d)$ 是整数值，因此该调度仍符合机器数量限制。

如果只有 j_3 存在，则可以将其安排在 $\gamma(j_3, \frac{10}{7}d) \leq 2$ 台机器上，并且该调度同样符合机器数量限制。

如果 j_1和 j_3都存在，则将 j_3安排在 2 台机器上，高度至多为 $\frac{4}{7}d \cdot \frac{3}{2} = \frac{6}{7}d$，并将 j_1 放置在其中一台机器的顶部。我们将这两个作业视为一个高度为 $t(j_3, 2) + t(j_1, 1)$ 的作业，并将其安排在 1 台机器上；以及一个高度为 $t(j_3, 2)$ 的作业，并将其安排在 1 台机器上。

◀

▶ 备注 7。注意，将作业 j_3 拆分为两部分可能会破坏调度的连续性。不过，我们只需重新排列机器，使该作业安排在相邻机器上即可。在下面的每个证明中，我们将分别重新讨论这个问题，以确保在任何时候， j_3 的两部分都能安排在相邻机器上。

在接下来的章节中，我们将引理 6 中构造的调度称为三层货架调度（定义 8，取 $\lambda = \frac{10}{7}$ ）。为此，我们将作业分配到 $\mathcal{C}_1 \sqcup \mathcal{C}_2 \sqcup \mathcal{C}_3$ 三个货架上，如图 2 所示。$\mathcal{C}_1 \sqcup \mathcal{C}_2$ 中的作业被分配到前两个货架（S_0 和 S_1）上，使得执行时间大于 d 的任何作业都位于 S_0 中，其余作业则位于 S_1 中。$\mathcal{C}_3$ 中的作业被放入 S_2 ，其执行时间至多为 $\frac{3}{7}d$。

–

▶**定义 8**（三层货架调度）。给定完工时间猜测值 d 和 $\lambda \in [\frac{10}{7}, \frac{3}{2})$ ，三层货架调度是对作业 $S_0 \sqcup S_1 \sqcup S_2 = J_B = J \setminus J_S$ 的一种划分，满足：

$\sum_{j \in S_0} w(j, \gamma(j, \lambda d)) + \sum_{j \in S_1} w(j, \gamma(j, d)) + \sum_{j \in S_2} w(j, \gamma(j, (\lambda - 1)d))$

$\leq md - \mathcal{W}_S \sum_{j \in S_0} \gamma(j, \lambda d) + \sum_{j \in S_1} \gamma(j, d) \leq m$

注意，根据引理 6， $\sum_{j \in S_0} \gamma(j, \frac{10}{7}d) + \sum_{j \in S_1} \gamma(j, d) \leq m$。作业的总工作量仍有上界 $md - \mathcal{W}_S$。如图 2 所示，我们现在需要考虑剩下的问题：第 2 层货架可能需要过多的机器。

2.3 修复三层货架调度

本节致力于修复上一节得到的三层货架调度。尽管上述背包 formulation 表明可以实现 $\frac{10}{7}d$ 的完工时间保证，但为了获得可行调度，我们必须放宽这一保证。为此，我们用参数 $\lambda \in [\frac{10}{7}, \frac{3}{2})$ 来参数化算法的近似比，该参数将在后文确定。因此，我们的目标是构造一个高度至多为 λd 的调度，并允许第 2 层货架上的作业执行时间至多为 $(\lambda - 1)d$。

我们将 S_0 和 S_2 使用的机器数量分别记为 $m_0 := \sum_{j \in S_0} \gamma(j, \lambda d)$ 和 $m_2 := \sum_{j \in S_2} \gamma(j, (\lambda - 1)d)$。根据构造，使用 $m - m_0$ 台机器来调度 S_1 和 S_2 中的作业。引理 6 保证，调度 S_1 中的作业最多需要 $m - m_0$ 台机器，且高度不超过 d。我们将货架 1 中的空闲机器数量（即不执行任何作业 $j \in S_1$ 的机器）记为 q。请注意，目前尚不能保证 S_2 中的作业能够在剩余的 $m - m_0$ 台机器上调度，且最大高度为 $(\lambda - 1)d$，对于任意 $\lambda \in [\frac{10}{7}, \frac{3}{2})$ 均如此。本节余下部分中，我们假设给定了一个满足 $m_2 > m - m_0$ 的三层货架调度；否则该调度是可行的（图 2）。

以下证明利用作业总工作量的下界来证明所需的近似比。请记住，根据定理 3 的约束 C1，给定的调度满足 $\mathcal{W}_0 + \mathcal{W}_1 + \mathcal{W}_2 \leq md$，其中 $\mathcal{W}_k$ 表示货架 k 上作业的总工作面积。由于按照构造，货架 0 上作业的总工作量大于 dm_0，因此我们有 $\mathcal{W}_1 + \mathcal{W}_2 \leq d(m - m_0)$。为简化记法，不失一般性，我们忽略货架 0，并假设 $m_0 = 0$ ，如图 2（右）所示。

▶**引理 9.** 给定一个三层货架调度和任意 $\lambda \in [\frac{10}{7}, \frac{3}{2})$，我们可以在时间 $O(mn)$ 内找到一个完工时间至多为 λd 的可行连续调度；否则，以下性质成立：

(i) 至多有一台机器执行高度小于 $j \in S_1$ 的作业 $\frac{\lambda}{2}d$。**(ii)** 中任一作业的工作面积 S_2 大于 λdq。**(iii)** 的工作面积 S_1 大于 $\frac{\lambda}{2}d(m - q)$。

–

证明。 此引理的证明采用了 Mounié、Rapine 和 Trystram 最初提出的技术 [29]。为完整起见，我们在附录 A 中给出完整证明。◀

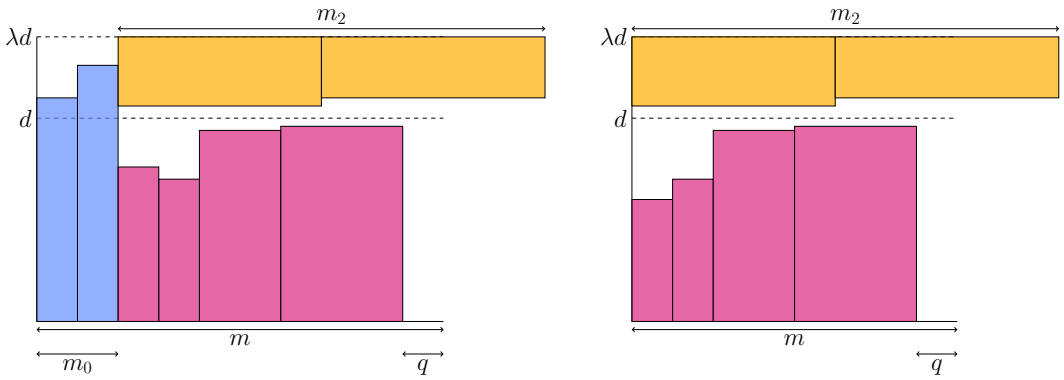

图 2 不可行的三层货架调度。蓝色作业位于 S_0 中，粉色作业位于 S_1 中，黄色作业位于 S_2 中。

本节余下部分的目标是证明，S_2 所需的机器数不超过 m。新的思路是将 S_2 中的作业压缩到高于其所在货架高度的位置（$(\lambda-1)d$），因为 S_1 中的作业不可能利用其所在货架的全部高度。我们据此提出了多种新的修复算法，并证明这些算法能够通过多种方式构造出可行调度。

▶ **引理 10。** 给定一个包含 $q \le m/6$ 的三层货架调度，我们可以在时间 $O(mn)$ 内找到一个完工时间至多为 $\frac{10}{7}d$（如果 $q = 0$）以及 $\frac{13}{9}d$（如果 $0 < q \le m/6$）的可行连续调度。

证明。该引理的证明采用了相当简单的技术，因此我们将其推迟至附录 B。◀

接下来还需证明，对于情况 $q > m/6$，我们也能找到一个可行调度。我们先给出以下观察，为证明做准备：

▶ **观察 11.** 给定一个满足 $q > m/6$ 和 $\lambda \ge \frac{4}{3}$ 的三层货架调度，S_2 中的作业数量至多为 $|S_2| \le 1$。

证明。 假设 $|S_2| > 1$。根据引理 9.(ii)，我们有 $(\mathcal{W}_2 > \lambda dq \cdot |S_2| \ge 2\lambda dq)$。结合引理 9.(iii)，得到：

$$
\begin{aligned}
&\mathcal{W}_1 + \mathcal{W}_2 \le dm \\
\Rightarrow\ &\frac{\lambda}{2}d(m-q) + 2\lambda dq < dm \\
\Leftrightarrow\ &\frac{\lambda}{2}m + \frac{3}{2}\lambda q < m \\
\Leftrightarrow\ &\frac{3}{4}\lambda m < m \qquad\qquad (q > m/6) \\
\Leftrightarrow\ &\lambda < \frac{4}{3}
\end{aligned}
$$

这与 $\lambda \ge \frac{4}{3}$ 矛盾。 – ◀

算法 2 RepairS2-2

将 S_2 中的单个作业记为 j_0
按执行时间降序排列 S_1 中的所有作业
设置 $i = 0$
对每个 $i \in \{0, \ldots, m-q-1\}$ **执行**
    将 j_0 放置在负载最轻的 $m-i$ 台机器上
    if $C_{\max} \leq \lambda d$ **然后**
        返回
    结束 if
结束 for

由此，我们可以假设 $|S_2| = 1$，并将 S_2 中的唯一作业记为 j_0。我们可以通过枚举该作业的所有可能分配量来利用这一观察（算法 2）。下面的引理将分析最后一种（也是最复杂的）情况的正确性。

▶ **引理 12.** 给定一个满足 $q > m/6$ 的三层货架调度，对于任意 $\lambda \geq -\frac{1}{3}W_{-1}(-\frac{3}{e^4}) \approx 1.4593$，我们可以找到一个完工时间至多为 λd 的可行调度。

证明。 我们旨在通过证明在机器 $i \in \{0, \ldots, m-1\}$ 上调度的任意作业 S_1 的高度都大于 $\max\left(\lambda d - \frac{dm - \frac{\lambda}{2}d}{(m-q)}\frac{(m-i)}{}, \frac{\lambda}{2}d\right)$，来证明该引理，前提是我们~~无法使用算法~~ 2 找到可行调度（如图 3 所示）。基于这一观察，我们可以分析该调度的总工作面积，并与 $\mathcal{W}_1 + \mathcal{W}_2 \leq dm$ 矛盾。

考虑一个如图 3（左）所示的调度，其中算法 2 无法放置 $j_0 \in S_2$。我们首先证明，当按照处理时间非递增的顺序将货架 1 中的所有作业放置时，每台机器 $i \in \{0, \ldots, m-1\}$ 上的作业的处理时间（负载 L_i）都大于 $g(i) = \lambda d - \frac{dm - \frac{\lambda}{2}d}{(m-q)}\frac{(m-i)}{}$。

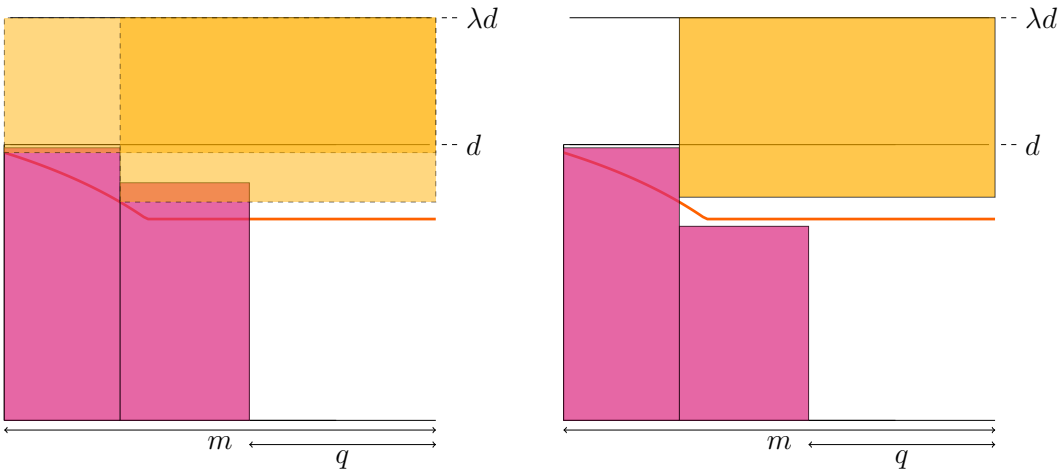

图 3 应用算法 2 后的不可行（左）和可行（右）调度。红线表示最大值 $\left(\frac{\lambda d}{2}, \lambda d - \frac{dm - \frac{\lambda}{2}d}{\binom{m-q}{m-i}}\right)$。粉色作业位于 $\overline{S_1}$，黄色作业位于 S_2。

▶**假设 2.1.** 如果算法 2 未能找到可行调度，则可以将 S_1 中的作业放置为使每台机器 $i \in \{0, \ldots, m-1\}$ 上的负载 L_i 满足：

$$L_i > g(i) = \lambda d - \frac{dm - \frac{\lambda}{2}d(m-q)}{m-i}$$

该假设可以通过简单的反证法证明。我们在附录 D.1 中给出形式化证明。

我们还需要分析在假设 2.1 下该调度的总工作面积，从而与 $\mathcal{W}_1 + \mathcal{W}_2 \leq dm$ 矛盾。我们将分两种情形进行分析。

情形 *1*（$m < 6$）：在这种情形下，我们明确计算 $\mathcal{W}_1 + \mathcal{W}_2$ 的一个上界。更多详情请参阅附录 D.4。

情形 *2*（$m \geq 6$）：这种情形中最重要的观察是，我们可以通过以下积分正确地界定货架 1 上的工作量：

$$\mathcal{W}_1 > \int_0^{m-q} \max\left\{\lambda d - \frac{dm - \frac{\lambda}{2}d(m-q)}{m-i}, \frac{\lambda}{2}d\right\} di \tag{6}$$

这个上界并非显而易见，因为根据引理 9.(i)，货架 1 可能包含一个执行时间小于 $\frac{\lambda}{2}d$ 的作业。我们在附录 D.2 中正式证明，在 $m \geq 6$ 的假设下，这个上界是正确的。

为了得出矛盾，我们需要证明

$$\begin{aligned}
&\mathcal{W}_1 + \mathcal{W}_2 \\
&> \int_0^{m-q} \max\left\{\lambda d - \frac{dm - \frac{\lambda}{2}d(m-q)}{m-i}, \frac{\lambda}{2}d\right\} di + \max\left\{(\lambda-1)dm, \lambda dq\right\} \\
&> dm
\end{aligned}$$

第一个不等式直接由公式 (6)、引理 9(ii) 和 $\gamma(j_0, (\lambda-1)d) > m \Rightarrow \mathcal{W}_2 > (\lambda-1)dm$ 得出。为了证明第二个不等式，我们将此表达式视为 q 中的一个函数，并证明它在 $q = \frac{\lambda-1}{\lambda}m$ 处取得的最小值大于 dm。 ——

▶**观察13**。该函数

$$\mathcal{W}(q) = \int_0^{m-q} \max\left\{\lambda d - \frac{dm - \frac{\lambda}{2}d(m-q)}{m-i}, \frac{\lambda}{2}d\right\} di + \max\left\{(\lambda-1)dm, \lambda dq\right\}$$

对于任意 $\lambda \in [\frac{10}{7}, \frac{3}{2})$，在 ~~$q$~~ $= \frac{\lambda-1}{\lambda}m$ 处取得最小值。

我们在附录 D.3 中给出这一观察的完整证明。

由于 $\mathcal{W}(q)$ 在区间 $[0, m]$ 内于 $q = \frac{\lambda-1}{\lambda}m$ 处取得最小值，我们需要证明：

$$\begin{aligned}
&\mathcal{W}\left(\frac{\lambda-1}{\lambda}m\right) > dm \\
\Leftrightarrow &\int_0^{m-\frac{\lambda-1}{\lambda}m} \max\left\{\lambda d - \frac{\frac{1}{2}dm}{m-i}, \frac{\lambda}{2}d\right\} di + (\lambda-1)dm > dm \\
\Leftrightarrow &\int_0^{m-\frac{\lambda-1}{\lambda}m} \max\left\{\lambda d - \frac{\frac{1}{2}dm}{m-i}, \frac{\lambda}{2}d\right\} di > (2-\lambda)dm
\end{aligned} \tag{7}$$

为了进一步分析这个表达式，我们需要仔细考察积分项。

首先，我们来找出最大函数的切换点 i'，此时 $\lambda d - \frac{\frac{1}{2}dm}{m-i'} = \frac{\lambda}{2}d$。

$$\begin{aligned}
&\lambda d - \frac{\frac{1}{2}dm}{m-i'} = \frac{\lambda}{2}d \\
&\Leftrightarrow \frac{\frac{1}{2}dm}{m-i'} = \frac{\lambda}{2}d \\
&\Leftrightarrow dm = \lambda d(m-i') \\
&\Leftrightarrow \frac{\lambda-1}{\lambda}m = i'
\end{aligned}$$

然后，我们可以在 i' 处将方程（7）中的积分拆分，并分别分析两部分

$$\begin{aligned}
&\int_0^{m-\frac{\lambda-1}{\lambda}m} \max\left\{\lambda d - \frac{\frac{1}{2}dm}{m-i}, \frac{\lambda}{2}d\right\} di \\
&= \int_0^{\frac{\lambda-1}{\lambda}m} \left(\lambda d - \frac{\frac{1}{2}dm}{m-i}\right) di + \int_{\frac{\lambda-1}{\lambda}m}^{m-\frac{\lambda-1}{\lambda}m} \frac{\lambda}{2}d\, di \\
&= \lambda d \int_0^{\frac{\lambda-1}{\lambda}m} 1\, di - \int_0^{\frac{\lambda-1}{\lambda}m} \frac{\frac{1}{2}dm}{m-i}\, di + \left(\left(m - \frac{\lambda-1}{\lambda}m\right) - \frac{\lambda-1}{\lambda}m\right)\frac{\lambda}{2}d \\
&= (\lambda-1)\, dm - \frac{1}{2}dm \int_0^{\frac{\lambda-1}{\lambda}m} \frac{1}{m-i}\, di + \frac{2-\lambda}{2}dm \\
&= (\lambda-1)\, dm - \frac{1}{2}dm \left(\ln(m) - \ln\left(m - \frac{\lambda-1}{\lambda}m\right)\right) + \frac{2-\lambda}{2}dm \\
&= (\lambda-1)\, dm - \frac{1}{2}dm \ln(\lambda) + \frac{2-\lambda}{2}dm
\end{aligned}$$

结合方程（7），我们得到：

$$\begin{aligned}
(\lambda-1)\, dm - \frac{1}{2}dm \ln(\lambda) + \frac{2-\lambda}{2}dm &> (2-\lambda)dm \\
\Leftrightarrow -dm \ln(\lambda) + (2-\lambda)\, dm &> (6-4\lambda)dm \\
\Leftrightarrow \ln(\lambda) &< 3\lambda - 4 \\
\Leftrightarrow \lambda &< e^{3\lambda-4} \\
\Leftrightarrow \lambda e^{-3\lambda} &< e^{-4} \\
\Leftrightarrow -3\lambda e^{-3\lambda} &> -3e^{-4}
\end{aligned}$$

朗伯 W 函数的定义是：当且仅当 $W(x) = y$ 时，$ye^y = x$。注意，我们使用朗伯函数的 W_{-1} 分支，由于该分支单调递减，因此结合 $x := -3e^{-4}$，$y := -3\lambda$，可得以下不等式。

$$\begin{aligned}
&-3\lambda < W_{-1}(-3e^{-4}) \\
&\Leftrightarrow \lambda > -\frac{1}{3}W_{-1}(-3e^{-4})
\end{aligned}$$

因此，对于任意 $\lambda > -\frac{1}{3}W_{-1}(-3e^{-4}) \approx 1.4593$，我们都得出与 $\mathcal{W}_1 + \mathcal{W}_2 > dm$ 矛盾的结论。至此，引理 12 的证明完成。 ◀

▶**观察 14.** 算法 *2* 构造的可行调度可以重新排列为连续调度，而不改变完工时间。

证明。 注意，根据定义，完全位于同一货架中的任何作业都可以安排在一组连续的机器上。至多存在一个作业 j_3（备注 7 中提到），可能会被拆分到两个货架之间。这个证明的一个关键观察是，虽然 S_1 中的作业顺序很重要，但我们可以将 S_1 中的作业划分为两组。第一组包含所有能放在 S_2 中那个单独作业下方的作业，第二组包含所有不能放在该作业下方的作业。这两组内部的顺序无关紧要。因此，我们可以将作业 j_3 放在货架 1 的边缘处，并且在需要时，可以将分配量镜像。这样，我们便能保证作业 j_3 与其在 S_0 中的对应作业安排在相邻机器上。 ◀

引理 10 和 12 以及观察 14 完成了定理 3 的证明，并表明我们可以在 J_B 中为所有作业找到可行连续调度。此前被忽略的 J_S 中的小作业可以贪心地添加到该调度中。我们在附录 E 中对这一过程进行了更准确的描述。

所提出算法的时间复杂度为 $O(nm)$。借助对偶近似框架，我们可以找到完工时间至多为 $(1.4593+\varepsilon)OPT$ 的可行调度，其时间复杂度为 $O(nm\log\ 1/\varepsilon)$。

3 实验评估

除了对当前最先进算法在理论上的改进之外，我们还实现了所提出的算法。下面，我们展示该算法在随机生成的实例上的一些实验结果，并表明该算法能够在合理的时间内解决现实实例。该实现采用 Java 编写，并已公开发布在 GitHub（ https://anonymous.4open.science/r/MoldableJobScheduling-9AE3）上，同时公开的还有测试实例和结果。我们在包含 n 个作业和 m 台机器的随机生成实例上进行了测试，其中每个作业的执行时间在 $[1,100]$（在一台机器上）上均匀分布，其余值也均匀分布，并遵循单调性限制。

所有测试均在英特尔 i5-8400T CPU 的单个核心上运行，并为 Java 进程提供 2GB 随机存取存储器。

我们将测试分为两组：一组固定作业数量为 $n=1000$，改变机器数量 $m\in[500,2000]$；另一组固定机器数量为 $m=1000$，改变作业数量 $n\in[500,2000]$。对于所有测试，我们均设置$\varepsilon=0.05$。

图 4 和图 5 展示了我们的实现对这两组测试的执行时间。尽管我们没有特别优化实现，该算法仍能在不到 30 秒的时间内求解所有实例。对于 $n>>m$，执行时间有所缩短，因为这些实例的最优完工时间更长。因此，更多作业会被归类为小作业，实例也更容易求解。

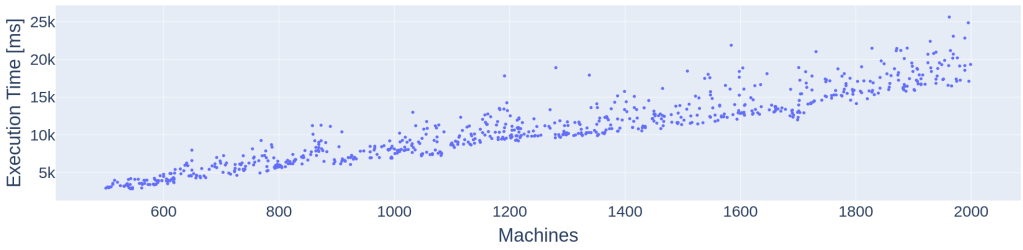

图4 不同机器数量下算法的运行时间。

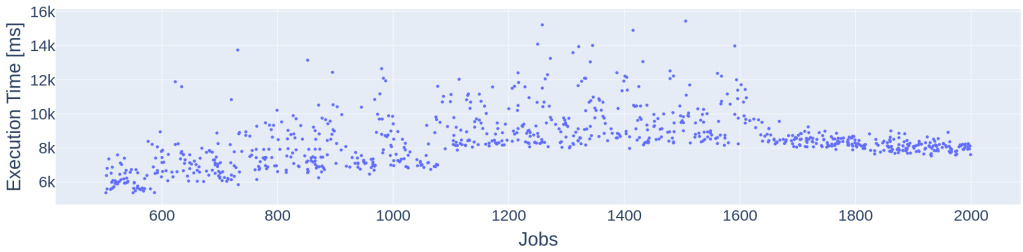

图5 不同作业数量下算法的运行时间。

我们注意到，尽管理论最坏情况近似比为 $1.4593+\varepsilon$，但对于所有随机生成的实例，该算法都能保证完工时间低于 $(\frac{10}{7}+\varepsilon)OPT$。这是因为最坏情况近似比只会出现在非常特殊的情形下。更准确地说，除非货架 1 上的空闲机器数量大于 $m/6$，否则所提出的算法可以保证完工时间至多为 $(\frac{13}{9}+\varepsilon)OPT$（见引理 12）。

然而，我们可以证明，我们的分析在最坏情况下是紧的。考虑如下包含 $m=13$ 和 $n=10$的实例。所有作业都具有如下取值的常数工作函数：

- j_1: $w(j,k)=6.01, \quad \forall k \in \{1,\ldots,m\}$ j_2:
- $w(j,k)=0.99, \quad \forall k \in \{1,\ldots,m\}$
- j_3-j_{10}: $w(j,k)=\frac{3}{4}, \quad \forall k \in \{1,\ldots,m\}$

该实例的最优调度的完工时间为 $C_{\max}=1$，因为所有作业的工作量为 $W=13$，并且通过将所有作业分配到 $m=13$ 台机器上，我们可以实现 $OPT=W/m=1$的完工时间。另一方面，在我们的算法中，MCKP问题的最优解为 $\mathcal{C}_1=\{j_2,\ \ldots,\ j_{10}\}$、$\mathcal{C}_2=\emptyset$、$\mathcal{C}_3=\{j_1\}$。在这种情况下，算法会将 j_1 安排在13台机器上，并将其余作业分别安排在1台机器上。这将产生 $C_{\max}=6.01/13+0.99\approx 1.452$的完工时间。对于更大的 m，我们可以构造类似的实例，并使近似比任意接近我们的理论最坏情况保证 $1.4593+\varepsilon$。由于这些实例的结构非常特殊，它们在实践中极不可能出现。

4 结论

本文提出的算法对于调度单调问题实现了略低于 $(73/50+\varepsilon)$ 的近似比，时间复杂度为 $O(nm\log 1/\varepsilon)$，适用于任意 $\varepsilon>0$

可塑作业调度问题，在时间复杂度为 {v3}({v4}log {v5}) 的任意 {v2} 下，实现了略低于 ({v1}) 的近似比。更令人惊讶的是，我们还展示了如何将该算法应用于问题的连续变体。由于我们为连续变体构造了与非连续变体相同的调度方案，因此该结果界定了这两个变体之间的差距，这是我们算法的一个有趣的附带效果。

我们通过实现证明，该算法在大多数情况下的比率为 $10/7$ ，而最坏情况近似比 $1.4593+\varepsilon$ 只会出现在非常特殊的问题情形中。

4.1 未来工作

所提出的算法打破了长期以来的瓶颈，为单调可塑性作业调度问题提供了一种实用高效且近似比低于 1。5 的算法。

我们希望这项工作能推动单调可塑性作业调度领域取得进一步进展。具体而言，我们将以下开放问题留待未来工作研究：

- 近似比还能进一步改善吗？对于实用高效的算法，目前尚无已知的近似比下界。正如我们所展示的，该算法在大多数情况下已经达到优于最坏情况保证的近似比。对于连续变体，这个问题尤其值得关注，因为目前尚无已知的多项式时间近似方案。

- 能否改进运行时间？ Jansen 和 Land [19] 引入了压缩 的概念，使算法的运行时间对机器数量 m 呈对数依赖。Grage、Jansen 和 Ohnesorge [16]利用（min， +）卷积进一步改进了这一方法。这些技术或许可以应用于我们的算法，以降低运行时间对 m 的依赖。

- 这一结果能否应用于相关问题？ 所提出的算法有望改进其他密切相关的问题，例如 CPU/GPU 调度 [15],、二维背包 [14], 、条带装箱 [17], 或需求条带装箱 [10] （以获得实用高效的算法）。

参考文献

1 Jacek Blazewicz、T. C. Edwin Cheng、Maciej Machowiak 和 Ceyda Oguz。泊位与岸桥分配：一种可塑性任务调度模型。《运筹学会杂志》，62(7):1189–1197，2011年。doi:10.1057/jors.2010.54。2 Raphaël Bleuse、Sascha Hunold、Safia Kedad-Sidhoum、Florence Monna、Grégory Mounié 和 Denis Trystram。在多核处理器上调度带图形处理器的独立可塑任务。《*IEEE*并行与分布式系统汇刊》，28(9):2689–2702，2017年。 doi:10.1109/TPDS.2017.2675891。3 Robert M Corless、Gaston H Gonnet、David EG Hare、David J Jeffrey 和 Donald E Knuth。关于朗伯 W 函数。《计算数学进展》，5:329–359，1996年。doi:10.1007/BF02124750。4 Thomas Decker、Thomas Lücking 和 Burkhard Monien。用于调度同型可伸缩任务的 54 近似算法。《理论计算机科学》，361(2-3):226–240，2006年。doi:10.1016/j.tcs.2006.05.012。5 Xavier Delorme、Alexandre Dolgui、Sergey Kovalev 和 Mikhail Y. Kovalyov。使定节拍混合型号装配线的工人数量最少化。《欧洲运筹学杂志》，272(1):188–194，2019年。 doi:10.1016/j.ejor.2018.05.072。

6 Alexandre Dolgui、Sergey Kovalev、Mikhail Y. Kovalyov、Sergey Malyutin 和 A meur Soukhal。定节拍装配线作业的最优劳动力分配。《欧洲运筹学杂志》，264(1):200–211，2018年。 doi:10.1016/j.ejor.2017.06.017。7 Maciej Drozdowski。多处理器任务调度的复杂度。《波兰科学院技术科学通报》，43:381–392，1995年。8 Maciej Drozdowski。《并行处理调度》，第18卷。Springer，2009年。 doi:10.1007/978-1-84882-310-5。9 Jianzhong Du 和 Joseph Y-T Leung。并行任务系统调度的复杂度。《SIAM离散数学杂志》，2(4):473–487，1989年。 doi:10.1137/0402042。10 Franziska Eberle、Felix Hommelsheim、Malin Rau 和 Stefan Walzer。用于需求条带装箱的紧确 $(3/2 + \in)$ 近似算法。载于 SODA，第641–699页。SIAM，2025年。doi:10.1137/1.9781611978322.20。11 Dror G. Feitelson 和 Larry Rudolph。迈向并行超级计算机作业调度器的趋同。载于 JSSPP，《计算机科学讲义》第1162卷，第1–26页。Springer，1996年。 doi:10.1007/BFb0022284。12 Dimitris Fotakis、Jannik Matuschke 和 Orestis Papadigenopoulos。超越同型机器的可伸缩调度。载于 APPROX-RANDOM，LIPIcs 第145卷，第17:1–17:14页。达格施图尔城堡—莱布尼茨信息学中心，2019年。 doi:10.4230/LIPIcs.APPROX-RANDOM.2019.17。13 Dimitris Fotakis、Jannik Matuschke 和 Orestis Papadigenopoulos。在次可加或次模处理速度下分配和调度广义可伸缩作业。《运筹学》，73(3):1598–1614，2025年。 doi:10.1287/opre.2022.0168。14 Waldo Gálvez、Fabrizio Grandoni、Sandy Heydrich、Salvatore Ingala、Arindam Khan 和 Andreas Wiese。通过 l-装箱近似几何背包。载于 FOCS，第260–271页。IEEE计算机学会，2017年。 doi:10.1109/FOCS.2017.32。15 Bernhard Sebastian Germann、Klaus Jansen、Felix Ohnesorge 和 Malte Tutas。CPU/GPU 调度的 3/2 对偶近似。载于第22届国际实验算法研讨会（SEA 2024）。达格施图尔城堡—莱布尼茨信息学中心，2024年。 doi:10.4230/LIPIcs.SEA.2024.13。16 Kilian Grage、Klaus Jansen 和 Felix Ohnesorge。利用压缩与卷积改进单调可塑作业调度算法。载于 Euro-Par，《计算机科学讲义》第14100卷，第503–517页。Springer，2023年。 doi:10.1007/978-3-031-39698-4_34。17 Rolf Harren、Klaus Jansen、Lars Prädel 和 Rob van Stee。条带装箱的 $(5/3 + \epsilon)$ 近似算法。载于 WADS，《计算机科学讲义》第6844卷，第475–487页。Springer，2011年。 doi:10.1007/978-3-642-22300-6_40。18 Dorit S. Hochbaum 和 David B. Shmoys。在调度问题中使用对偶近似算法：理论与实践结果。《ACM杂志》，34(1):144–162，1987年。 doi:10.1145/7531.7535。19 Klaus Jansen 和 Felix Land。线性时间调度单调可塑作业。载于2018年IEEE国际并行与分布式处理研讨会（IPDPS 2018），第172–181页。IEEE计算机学会，2018年。 doi:10.1109/IPDPS.2018.00027。20 Klaus Jansen、Felix Land 和 Kati Land。界定调度与装箱问题算法的运行时间。《SIAM离散数学杂志》，30(1):343–366，2016年。 doi:10.1137/140952636。21 Klaus Jansen 和 Lorant Porkolab。可伸缩并行任务调度的线性时间近似方案。载于第十届ACM-SIAM离散算法年度研讨会论文集（SODA 1999），第490–498页。ACM/SIAM，1999年。URL: http://dl.acm.org/citation.cfm?id=314500.314870。22 Klaus Jansen 和 Malin Rau。缩小伪多项式条带装箱的差距。载于第27届欧洲算法研讨会（ESA 2019），LIPIcs 第144卷，第62:1–62:14页。达格施图尔城堡—莱布尼茨信息学中心，2019年。 doi:10.4230/LIPIcs.ESA.2019.62。

23 Klaus Jansen 和 Ralf Thöle。并行作业调度的近似算法：突破 2 的近似比。载于 ICALP (1)，《计算机科学讲义》第 5125 卷，第 234–245 页。Springer，2008 年。 doi:10.1007/978-3-540-70575-8_20。24 Klaus Jansen 和 Ralf Thöle。并行作业调度的近似算法。《SIAM J. Comput.》，39(8):3571–3615，2010 年。 doi:10.1137/080736491。25 Berit Johannes。以最小化完工时间为目标的并行作业调度。《J. Sched.》，9(5):433–452，2006 年。 doi:10.1007/s10951-006-8497-6。26 Hans Kellerer、Ulrich Pferschy 和 David Pisinger。背包问题。Springer，2004 年。doi:10.1007/978-3-540-24777-7。27 Walter Ludwig 和 Prasoon Tiwari。可伸缩与不可伸缩并行任务的调度。载于 Daniel Dominic Sleator 编，《第五届 ACM-SIAM离散算法年度研讨会论文集》（SODA 1994），第 167–176 页。ACM/SIAM，1994 年。URL： http://dl.acm.org/citation.cfm?id=314464.314491。28 Grégory Mounié、Christophe Rapine 和 Denis Trystram。可伸缩任务调度的高效近似算法。载于《第十一届并行算法与体系结构年度研讨会论文集》（SPAA 1999），第 23–32 页。ACM，1999 年。 doi:10.1145/305619.305622。29 Gregory Mounie、Christophe Rapine 和 Denis Trystram。用于调度独立单调可伸缩任务的 3/2-近似算法。《SIAM J. Comput.》，37(2):401–412，2007 年。 doi:10.1137/S0097539701385995。30 John Turek、Joel L Wolf 和 Philip S Yu。并行化任务调度的近似算法。载于《第四届ACM并行算法与体系结构年度研讨会论文集》（SPAA 1992），第 323–332 页，1992 年。 doi:10.1145/140901.141909。31 Fangfang Wu、Xiandong Zhang 和 Bo Chen。单调可塑任务调度的改进近似算法。《欧洲运筹学杂志》，306(2):567–578，2023年。 doi:10.1016/j.ejor.2022.08.034。

引理 9 的证明

我们首先对三层货架调度应用变换（算法 3），这有助于减少 S_2 使用的机器数量。这些变换基于 Mounié、Rapine 和 Trystram 的思路 [29]，但以 λ 为参数。

T1 若作业 $j \in S_1$ 的执行时间至多为 $\frac{\lambda}{2}d$ 且分配给 $p > 1$ 台机器，则将 j 分配给 $\gamma(j, \lambda d)$ 台机器，位于 S_0。
T2 若 j，$j' \in S_1$ 的执行时间小于 $\frac{\lambda}{2}d$ 且各自分配给 1 台机器，则将 j 和 j' 分配给同一台机器，位于 S_0。
T3 令 q 表示 S_1 中的空闲机器数量。若存在作业 $j \in S_2$，其在 λd 台机器上的执行时间小于 q，则将 j 分配到 $\gamma(j, \lambda d)$ 台机器中的任意一台，具体取决于其执行时间：S_1 或 S_0。

算法 3 修复货架调度

要求： 三层货架调度；$\lambda \in [\frac{10}{7}, \frac{3}{2})$。1: **当**可能**时**2: 应用变换 T1、T2 或 T3。3: **结束循环**

接下来，我们将分析应用算法 3 后得到的调度。请记住，为了便于表示，我们假设 $m_0 = 0$。这不会影响证明的正确性，因为我们与调度的总工作量矛盾，而货架 0 的总工作量大于 dm_0。

▶ **引理 9**。给定一个三层货架调度以及任意 $\lambda \in [\frac{10}{7}, \frac{3}{2})$，我们可以在时间 $O(mn)$ 内找到一个完工时间至多为 λd 的可行连续调度，或者以下性质成立：

(i) 至多有一台机器执行高度小于 $\frac{\lambda}{2}d$ 的作业 $j \in S_1$。**(ii)** 任意作业的工作面积 S_2 大于 λdq。**(iii)** 工作面积 S_1 大于 $\frac{\lambda}{2}d\,(m - q)$。

证明。我们先证明应用算法 3 后性质 (i) 和性质 (ii) 成立，再证明调度具有连续性，并分析该算法的时间复杂度。

我们先证明引理中的三个性质。根据算法 3 的定义，任意作业 $j \in S_1$ 都会被调度到 $\gamma(j, d)$ 台机器上。这意味着，根据 $\gamma(j, d)$ 的定义和单调性，执行时间小于 $\frac{\lambda}{2}d$ 的任意作业都会被调度到恰好一台机器上。调度中至多包含一个这样的作业，否则我们就可以应用变换 T2，这与算法的终止条件矛盾。性质 (ii) 直接成立，因为无法应用变换 T3。

对于最后一个性质，我们采用反证法：假设性质 (iii) 不成立。这意味着存在一个作业 $j \in S_1$，其执行时间为 $t \in (\frac{3}{7}d, \frac{\lambda}{2}d]$。根据性质 (i)，这样的作业至多有一个，且它被调度到恰好一台机器上。我们分三种情况讨论：

情形 1 ($S_1 \setminus \{j\} = \emptyset$ 和 $q = 0$)：在此情形下，$m = 1$。我们将所有作业安排在这台机器上，从而得到一个可行调度，若 $\mathcal{W}_1 + \mathcal{W}_2 \leq d$。

情形 2 ($S_1 \setminus \{j\} = \emptyset$ 和 $q > 0$)：在此情形下，$q = m - 1$。由于 S_2 包含至少一个作业 $\mathcal{W}_2 > \lambda dq = \lambda d(m-1)$，根据性质 (ii)。当$\mathcal{W}_1 + \mathcal{W}_2 \le dm$时，我们得到：

$$\begin{aligned}
&\frac{3}{7}d + \lambda d(m-1) < dm \\
&\Leftrightarrow \frac{3}{7} + \lambda m - \lambda < m \\
&\quad\Leftrightarrow (\lambda - 1)m < \lambda - \frac{3}{7} \\
&\qquad\Leftrightarrow m < \frac{\lambda - \frac{3}{7}}{\lambda - 1}
\end{aligned}$$

对于 $\lambda \in [\frac{10}{7}, \frac{3}{2})$，这意味着 $m \le 2$。由于 $q \ge 1$和 $m = q + 1$，我们必有 $m = 2$和 $q = 1$。首先证明 S_2 至多包含一个作业。为此，设 S_2 中的作业数为 k，那么根据性质 (ii) 和 $q = 1$，S_2 的总工作面积为 $\mathcal{W}_2 > \lambda dqk = \lambda dk$。由于 $\mathcal{W}_1 + \mathcal{W}_2 \le dm$，我们得到：

$$\begin{aligned}
&\frac{3}{7}d + \lambda dk < 2d \\
&\quad\Leftrightarrow k < \frac{11}{7\lambda} < \frac{11}{10} < 2 \qquad\qquad (\lambda \ge \frac{10}{7})
\end{aligned}$$

由此得到最终矛盾。如果我们找不到可行调度，那么S_2 中的单个作业在两台机器上的处理时间至少为 $\lambda d - t$。这意味着总工作面积为 $2(\lambda d - t) + t \ge \frac{3}{2}\lambda d > 2d$，其中第一个不等式由 $t \le \frac{\lambda}{2}d$得出，第二个由 $\lambda > \frac{10}{7}$ 得出。由于 $\mathcal{W}_1 + \mathcal{W}_2 \le 2d$，因此产生矛盾。

情形 3 ($S_1 \setminus \{j\} \neq \emptyset$)：如果性质 (iii) 不成立，我们可以假设存在另一个作业 $j' \in S_1 \setminus \{j\}$，其执行时间小于 $\lambda d - t$。然后，我们可以将 j调度在 j' 的上方。注意，$t(j, d) + t(j', d) > d$，因此我们将 j（以及 j' 的一部分）放入 S_0。新的调度满足 (iii)。

接下来需要证明所得调度是连续的。我们必须确保没有作业被分割到不相邻的机器上。注意，这种情况只可能发生在被分割到两个货架之间的作业中。如果存在这样的作业 j_3（见备注 7），我们可以选择 j（违反性质 (iii) 的作业）为 $j := j_3$。我们考虑上面的三种情形：在情形 1 和情形 2 中，作业 $j\,(= j_3)$ 是 S_1 中唯一的作业。我们可以将其调度在货架 1 和货架 0 的边界处，从而确保其占据一段连续的机器。在情形 3 中，$j\,(= j)$ 被整体移入货架 0，因此不再被分割。另一个作业 j' 可以选为 S_1 中最小的作业，并放置在边界处，从而保持连续性。这保证了所有作业都被调度在相邻机器上。

时间复杂度由以下论证得出：设大作业的数量为 n_B。由于这些变换会将作业从第 2 层货架移到货架 1 或 0，并将作业从货架 1 移到货架 0，因此算法 3 中的 while 循环最多执行 $O(n_B)$ 次。变换 T1 和 T2 可在 $O(n_B)$ 时间内实现。扫描 S_2 中的作业以实现变换 T3 需要 $O(n_B)$，而为所选作业 $j \in S_2$确定 $\gamma(j, \lambda d)$ 需要 $O(\log m)$。由此得到时间复杂度为 $O(n_B^2 + n_B \log m)$。由于 n_B同时受 n 和 m限制，我们可以得出算法 3 的时间复杂度为$O(nm)$。◀

B 引理 10 的证明

▶ **引理 10**。给定一个满足 $q \le m/6$ 的三层货架调度，我们可以找到一个可行连续调度，其完工时间至多为 $\frac{10}{7}d$（如果 $q = 0$），且为 $\frac{13}{9}d$（如果 $0 < q \le m/6$），所需时间为 $O(mn)$。

算法 4 RepairS2-1

1: **当** $m_2 > m$ **时**2: 令 j 为 S_2 中高度最小的作业。3: 令 p 为分配给 j 的机器数量。4: 将 j 分配给 $p-1$ 台机器。5: **结束当**6: 按执行时间降序排列 S_1 中的作业。7: 按执行时间升序排列 S_2 中的作业。

通过应用算法 4 构造可行调度。在该过程中，我们迭代地将 S_2 中高度最小的作业压缩一台机器，直到 S_2 中的机器数量至多为 m。我们将证明，按货架 1（货架 2）上的执行时间降序（升序）排列时， S_1 和 S_2 中的作业不会相交。算法 4 的终止性在附录 C 中证明。根据 q 的取值，我们将引理 10 的证明分为两种情形。

情形 *1*（$q = 0$）：这是最简单的情形，在此我们将证明，对于任意 $\lambda \geq \frac{10}{7}$，若 $\mathcal{W}_1 + \mathcal{W}_2 \leq dm$，则可以得到完工时间至多为 λd 的可行调度。

▶ **观察 15**。应用算法 *4* 后，任意作业 $j \in S_2$ 的高度至多为 $\frac{4}{7}d$。

证明。假设存在一个作业 j，其执行时间严格大于 $\frac{4}{7}d$，且属于 S_2。由于工作量单调性，在算法 4 中最后一次压缩之前，该作业的高度至少为 $\frac{2}{7}d$。由于算法会选择最小的作业进行压缩， S_2 中的所有其他作业的高度都至少为 $\frac{2}{7}d$，因此 $\mathcal{W}_2 > \frac{2}{7}dm_2$。根据引理 9.(iii)，可得：$dm \geq \mathcal{W}_2 + \mathcal{W}_1 > \frac{2}{7}dm_2 + \frac{5}{7}dm > dm$。矛盾。◀

▶ **引理 16**。若 $q = 0$，算法 *4* 将给出一个完工时间至多为 $\frac{10}{7}d$ 的可行调度，除非 $W_1 + W_2 > dm$。

证明。反设算法 4 给出了一个不可行调度。这意味着存在一台机器 i，其负载为 $L_i > \frac{10}{7}d$，其中我们将 S_2 在该机器上的高度记为 t。

我们首先证明 $\frac{3}{7}d \leq t \leq \frac{4}{7}d$。事实上，下界显然成立，因为该机器上的负载大于 $\frac{10}{7}d$，而 S_1 上的任何作业的执行时间至多为 d。上界见观察 15。

现在我们可以继续分析该调度的工作量。 S_2中位于 i 右侧的每个作业的高度至少为 t，因为它们按处理时间升序排列，并且根据引理 9.(i)， S_1 中的每个作业（至多一个除外）的高度至少为 $\frac{\lambda}{2}d = \frac{5}{7}d$。由此可得：

$$L_{i'} \geq t + \frac{5}{7}d \geq \frac{8}{7}d \quad \forall i' \in \{i+1, \ldots, m-1\}$$

S_1 中位于 i 左侧的每个作业的高度至少为 $\frac{10}{7}d - t$，因为它们按处理时间降序排列，并且 S_2 中的每个作业的执行时间都大于$\frac{3}{14}d$，这是因为它们被放置在 $\gamma(j, \frac{3}{7}d)$ 台机器上。由此可得：

$$L_{i'} \geq \frac{10}{7}d - t + \frac{3}{14}d \geq \frac{15}{14}d \quad \forall i' \in \{1, \ldots, i-1\}$$

这些机器中的任意一台，其负载都严格大于 d。接下来只需考虑遗漏的两台机器， i 和 m。

机器 m 上的负载至少为 $t + \frac{3}{7}d \geq \frac{6}{7}d$，因为 J_B 中的任意作业在一台机器上的顺序高度至少为 $\frac{3}{7}d$。机器 i 上的负载至少为 $\frac{10}{7}d$。因此，它们的平均负载也严格大于 d，这与 $\mathcal{W}_1 + \mathcal{W}_2 \leq dm$ 矛盾。◀

▶ **观察 17**。 如果 $q = 0$，那么可以将算法 *4* 构造的可行调度修改为连续的。

证明。 我们回顾一下，至多有一个作业被拆分到两个货架上。这个作业 j_3，如备注 7 中所述，被拆分到 S_0 和 S_1 上，高度至少为 $\frac{5}{7}d$ 且至多为 $\frac{6}{7}d$。由于 S_2 中任何作业的高度至多为 $\frac{4}{7}d$（根据观察 15），因此货架 1 上高度至多为 $\frac{4}{7}d$ 的作业顺序无关紧要。因此，我们可以将 j_3 移至货架 1 最右侧的机器，并将其安排在货架 1 的边缘。其余部分的 j_3 将安排在货架 1 最左侧的机器上。这保证了所有作业都安排在相邻机器上。◀

情形 *2*（$0 < q \leq m/6$）：同样，我们先陈述几个将在本情形证明过程中用到的观察。

▶ **观察 18.** 给定 $\lambda \geq \frac{13}{9}$ 和 $q \leq m/6$，S_1 的总工作面积大于 $\frac{65}{108}dm$。而 S_2 的总工作面积小于 $\frac{43}{108}dm$。

证明. 根据引理 9.(iii) 和 $\lambda \geq \frac{13}{9}$，可得 $\mathcal{W}_1 > \frac{13}{18}d(m-q)$。根据 $q \leq \frac{1}{6}m$，可得：

$$\mathcal{W}_1 > \frac{13}{18}d(m-q) \geq \frac{13}{18}dm - \frac{13}{108}dm = \frac{65}{108}dm$$

$\mathcal{W}_2$ 的上界可直接由 $\mathcal{W}_1 + \mathcal{W}_2 \leq dm$ 得出。◀

▶ **观察 19.** 对于任意 $\lambda \geq \frac{13}{9}$，应用算法 *4* 后，任意作业 $j \in S_2$ 至少分配到3台机器上。

证明. 我们首先说明，在应用算法 4 之前，任意作业 $j \in S_2$ 至少分配到 3 台机器上，然后再证明在应用该算法的过程中这一性质始终成立。根据引理 9.(ii) 和 $q \geq 1$，$j \in S_2$ 中的每个作业的工作面积都大于 $\frac{13}{9}d$。因此，根据单调性和 γ 的定义，$\gamma(j, \frac{4}{9}d) > 3$。

现在假设，在算法执行过程中的任意时刻，作业 $j \in S_2$ 从三台机器压缩到两台机器。根据工作量单调性，我们知道 $t(j, 3) > \frac{13}{9}d/3 = \frac{13}{27}d$。由于算法选择最小的作业，其他所有作业都至少与它一样大。这意味着 S_2 的工作面积大于 $\frac{13}{27}dm > \frac{43}{108}dm$，这与观察 18 矛盾。◀

▶ **观察 20**。 S_2 中的所有作业的执行时间都大于 $\frac{12}{35}d$。

证明。 考虑任意作业 $j \in S_2$，根据 $q \geq 1$ 和引理 9.(ii)，该作业的工作面积至少为 $\frac{13}{9}d$。我们根据分配的机器数量将证明分为 2 种情况。假设 $\gamma(j, \frac{3}{7}d) \leq 4$。那么它的高度至少为 $\frac{13}{9}d/4 = \frac{13}{36}d > \frac{12}{35}d$。如果 $\gamma(j, \frac{3}{7}d) > 4$，那么根据 γ 的定义和单调性：

$$t\left(j, \gamma\left(j, \frac{3}{7}\right)\right) > \frac{3}{7}d \cdot \frac{\gamma(j, \frac{3}{7}d) - 1}{\gamma(j, \frac{3}{7}d)} \geq \frac{3}{7}d \cdot \frac{4}{5} = \frac{12}{35}d$$

◀

▶ **观察 21.** 应用算法 *4* 后，任意作业 $j \in S_2$ 的高度至多为 $\frac{43}{81}d$。

证明. 假设 S_2 包含一个高度大于 $\frac{43}{81}d$的作业 j。令 $\alpha(j)$ 表示分配给 j的机器数量。根据观察 19，我们知道 $\alpha(j) \geq 3$。按照观察 20 证明中的相同论证，该作业在额外一台机器上的高度为 $t(j, \alpha(j)+1) \geq \frac{43}{81}d \cdot \frac{\alpha(j)\alpha(j)+1}{} \geq \frac{43}{81}d \cdot \frac{3}{4} = \frac{43}{108}d$。由于 S_2中的任意作业都必须至少有这么高，这与观察 18 矛盾。◀

▶ **引理 22.** 如果 $0 < q \leq m/6$，算法 *4* 会生成一个可行调度，其完工时间至多为 $\frac{13}{9}d$，除非 $W_1+W_2 > dm$。

证明。 反设算法 4 生成了一个不可行调度。这意味着存在一台机器 i，其负载为 $L_i > \frac{13}{9}d$，我们将S_2在该机器上的高度记为 t。注意，$t \in (\frac{4}{9}d, \frac{43}{81}d)$。

现在我们可以开始分析该调度的工作量。S_1中位于 i左侧的每个作业，其高度都大于 $\frac{13}{9}d - t$，因为它们按处理时间降序排列；根据观察 20，S_2 中的每个作业的高度至少为 $\frac{12}{35}d$。由此可得：

$$L_{i'} \geq \frac{13}{9}d - t + \frac{12}{35}d > \frac{73}{63}d \quad \forall i' \in \{1, \ldots, i-1\} \tag{8}$$

S_2 中最后 $m-i$ 台机器上的任意作业，其高度至少为 $t > \frac{4}{9}d$，因为它们按处理时间升序排列。根据引理 9.(i)，S_1 上的每个作业（至多一个例外）的执行时间至少为 $\frac{13}{18}d$，由此可得除最后 $q+1$ 台机器外所有机器负载的下界：

$$L_{i'} \geq \frac{4}{9}d + \frac{13}{18}d = \frac{73}{63}d \quad \forall i' \in \{i+1, \ldots, m-q-1\} \tag{9}$$

根据选择，机器 i 的负载严格大于 $\frac{13}{9}d$，而 S_1 中负载最低的机器q 可能包含一个高度仅大于 $\frac{3}{7}d$的作业。因此，这两台机器的平均负载大于：$(\frac{13}{9}d + \frac{4}{9}d + \frac{3}{7}d)/2 = \frac{73}{63}d$。由于最后的 q台机器的负载都严格大于 $\frac{4}{9}d$，我们得到以下不等式：

$$\frac{73}{63}d \cdot (m-q) + \frac{4}{9}dq < dm$$
$$\Leftrightarrow q > \frac{2}{9}m$$

这与 $q \leq \frac{m}{6}$ 矛盾。◀

▶ **观察 23。** 如果 $0 < q \leq m/6$，则可以将算法 *4* 构造的可行调度修改为连续的。

证明。 我们回顾一下，最多只有一个作业会被拆分到两个货架上。这个作业 j_3，如备注 7 中所述，被拆分到 S_0 和 S_1，其高度至少为 $\frac{5}{7}d$ 且至多为 $\frac{6}{7}d$。由于 S_2中的任何作业高度至多为 $\frac{43}{81}d$（根据观察 21），货架 1 中高度至多为 $\frac{13}{9}d - \frac{43}{81}d = \frac{74}{81}d$ 的作业顺序并不重要。因此，我们可以将 j_3 移到货架 1 中最右侧的机器上，并将其安排在货架 1 的边缘。j_3 的剩余部分将安排在货架 1 中最左侧的机器上。这样可确保所有作业都安排在相邻机器上。◀

引理16和22以及观察17和23完成了引理 10 的证明。

C 算法 4 的终止性证明

为了证明算法 4 会终止，我们将证明 S_2 中的机器数量有界于 $2m$。首先注意，根据定义，J_B 中的任意作业的顺序执行时间都严格大于 $\frac{3}{7}d$。此外，只要 $m_2 > m$，S_2 中就必定存在一个作业被分配到至少两台机器上；否则，我们将直接得到$\mathcal{W}_2 > \frac{3}{7}dm$。因此，我们只需证明以下观察：

▶ **观察 24.** 给定一个满足 $q \leq \frac{m}{6}$ 的三层货架调度，第 2 层货架使用的机器数量 m_2 小于 $3m$。

证明。 假设 $m_2 \geq 3m$。注意，第 2 层货架上的任一作业的高度至少为 $\frac{\lambda-1}{2}d$，因为它被安排在 $\gamma(j,(\lambda-1)d)$ 台机器上。由 $\mathcal{W}_1+\mathcal{W}_2 \leq dm$ 以及引理 9.(iii)，可得：

$$\begin{aligned}\mathcal{W}_1+\mathcal{W}_2 &\geq \frac{\lambda}{2}d(m-q)+\frac{(\lambda-1)}{2}dm_2 \\ &\geq \frac{5}{12}\lambda dm+\frac{3(\lambda-1)}{2}dm && (q\leq\frac{m}{6};m_2\geq 3m)\\ &> dm && (\lambda\geq\frac{10}{7})\end{aligned}$$

矛盾。 ◀

根据这一观察，我们可以得出结论：算法 4 在 $O(mn)$ 次迭代内终止 .

D 引理 12 的省略证明

为了证明以下陈述，我们先给出一个观察。请注意，这个观察本质上是引理 9 的另一种表述。虽然我们会在该观察的证明中提到一些变换，但这些变换已在算法 3 中应用。因此，为使该观察成立，我们实际上不必改变调度。

▶ **观察 25.** 在 S_1 中，任意两台非空闲机器的平均负载都大于 $\frac{\lambda}{2}d$.

证明。 根据 $\frac{\lambda}{2}d$ 在 S_1 中的定义，执行时间小于或等于 γ 的任意作业都必须分配给一台机器。我们还注意到，这种作业至多有一个，否则就可以应用变换 T2。

现在假设存在一个作业 j，其执行时间小于 $\frac{\lambda}{2}d$，且存在另一个作业 j'，使得 $t(j,\gamma(j,d))+t(j',\gamma(j',d)) \leq \lambda d$。那么我们可以将 j 放在 j' 上方，并将它们放入 S_0，同时让 j' 部分留在 S_1 中。◀

D.1 假设 2.1 的证明

▶**假设 2.1。** 如果算法 2 未找到可行调度，则 S_1 中的作业可以进行放置，使得每台机器 $i \in \{0,\ldots,m-1\}$ 上的负载 L_i 满足：

$$L_i > g(i) = \lambda d - \frac{dm-\frac{\lambda}{2}d(m-q)}{m-i}$$

证明。 假设存在一台机器 i 不满足这一条件。现在我们将证明，S_2 中的唯一作业可以放置在 $m-i$ 负载最小的机器上，且不会使 λd的总完工时间超出上限，如图 3（右）所示。

根据工作量单调性，$\mathcal{W}_2 \leq dm - \mathcal{W}_1$，以及引理 9.(iii)，我们得到：

$$t(j_0, m-i) \leq \frac{w(j_0, m_2)}{m-i} < \frac{dm - \frac{\lambda}{2}d(m-q)}{m-i} \tag{10}$$

将 j_0 放置到 $m-i$ 台负载最轻的机器上时，没有任何机器的负载会超过 $g(i) + t(j_0, m-i)$，因为机器 i 在这 $m-i$ 台负载最轻的机器中负载最高。根据 g 的定义和公式 (10)，可得：

$$g(i) + t(j_0, m-i) < \lambda d$$

因此，算法 2 会将 j_0 调度到 $m-i$ 台机器上。在本证明的剩余部分中，我们可以假设每台机器 $i \in \{0, \ldots, m-1\}$ 都被调度了高度至少为 $g(i) = \lambda d - {}^{dm-\frac{\lambda}{2}d}\!/_{(m-q)\,m-i}$ 的作业（这些作业属于 S_1）。

D.2 公式 (6) 的证明

▶**观察 26**。 以下表达式是货架 *1* 上工作量的一个正确上界。

$$\mathcal{W}_1 > \int_0^{m-q} \max\left\{\lambda d - \frac{dm - \frac{\lambda}{2}d(m-q)}{m-i}, \frac{\lambda}{2}d\right\} di$$

证明。 根据引理 9.(i)，货架 1 上可能存在一个小于 $\frac{\lambda}{2}d$ 的作业。不过，根据观察 25，当至少两台机器的负载由最大函数中的第二项界定时，该不等式仍然成立，因为它们的平均负载大于 $\frac{\lambda}{2}d$。现在我们将证明 $m \geq 6$的情况确实如此。

设 x 为 $\lambda d - {}^{dm-\frac{\lambda}{2}d}\!/_{(m-q)\,m-i}$ 与 ${}^{\lambda d}\!/_{2}$ 的交点：

$$\begin{aligned}
&\lambda d - \frac{dm - \frac{\lambda}{2}d(m-q)}{m-x} = \frac{\lambda}{2}d \\
\Leftrightarrow\ &\frac{dm - \frac{\lambda}{2}d(m-q)}{m-x} = \frac{\lambda}{2}d \\
\Leftrightarrow\ &dm - \frac{\lambda}{2}d(m-q) = \frac{\lambda}{2}d(m-x) \\
\Leftrightarrow\ &\frac{2dm - \lambda d(m-q)}{\lambda d} = m - x \\
\Leftrightarrow\ &\frac{-2dm + \lambda d(m-q)}{\lambda d} + m = x \\
\Leftrightarrow\ &-\frac{2m}{\lambda} + 2m - q = x
\end{aligned}$$

注意，最大函数中的第二项所界定的机器数量为

$m-q-x$。现在我们要证明 $m-q-x\geq 2$:

$$\begin{aligned}
& m-q-x\geq 2 \\
\Leftrightarrow\ & m-q+\frac{2m}{\lambda}-2m+q\geq 2 \\
\Leftrightarrow\ & -m+\frac{2m}{\lambda}\geq 2 \\
\Leftrightarrow\ & m\left(-1+\frac{2}{\lambda}\right)\geq 2 \\
\Leftrightarrow\ & m\frac{-\lambda+2}{\lambda}\geq 2 \\
\Leftrightarrow\ & m\left(2-\lambda\right)\geq 2\lambda \\
\Leftrightarrow\ & 12-6\lambda\geq 2\lambda \qquad (m\geq 6) \\
\Leftrightarrow\ & \frac{3}{2}\geq\lambda
\end{aligned}$$

因此，当 $\lambda<\frac{3}{2}$ 时，该观察对 $m\geq 6$ 成立。– ◀

D.3 观察13的证明

▶**观察13**。函数

$$\mathcal{W}(q)=\int_0^{m-q}\max\left\{\lambda d-\frac{dm-\frac{\lambda}{2}d(m-q)}{m-i},\frac{\lambda}{2}d\right\}di+\max\left\{(\lambda-1)dm,\lambda dq\right\}$$

在 $q=\frac{\lambda-1}{\lambda}m$ 处取得最小值，对任意 $\lambda\in[\frac{10}{7},\frac{3}{2})$ 均成立。

为此，我们将证明以下观察:

▶**观察 27**。函数

$$a(q)=\int_0^{m-q}\max\left\{\lambda d-\frac{dm-\frac{\lambda}{2}d(m-q)}{m-i},\frac{\lambda}{2}d\right\}di+(\lambda-1)dm \tag{11}$$

在区间 $[0,\frac{\lambda-1}{\lambda}m]$ 内关于 q 单调递减。 ——

▶**观察28**。函数

$$b(q)=\int_0^{m-q}\max\left\{\lambda d-\frac{dm-\frac{\lambda}{2}d(m-q)}{m-i},\frac{\lambda}{2}d\right\}di+\lambda dq \tag{12}$$

在区间 $[\frac{\lambda-1}{\lambda}m,m]$ 内关于 q 单调递增。 ——

首先，我们将分析方程（11）和（12）的积分部分 $I(q)=\int_0^{m-q}\max\left\{\lambda d-\frac{dm-\frac{\lambda}{2}d(m-q)}{m-i},\frac{\lambda}{2}d\right\}di$。为简单起见，定义 $f(i,q):=\max\left\{\lambda d-\frac{dm-\frac{\lambda}{2}d(m-q)}{m-i},\frac{\lambda}{2}d\right\}$。然后，利用莱布尼茨法则，可得:

$$\begin{aligned}
\frac{dI(q)}{dq}&=f(m-q,q)\cdot\frac{d}{dq}(m-q)-\int_0^{m-q}\frac{\partial f(i,q)}{\partial q}\,di \\
&=-f(m-q,q)-\int_0^{m-q}\frac{\partial f(i,q)}{\partial q}\,di
\end{aligned} \tag{13}$$

对于积分项，由于最大值函数的存在，我们需要进行分段分析。令 $g(i,q)$：$= \lambda d - {}^{dm-\frac{\lambda}{2}d}{}_{(m-q)\,m-i}$ 和 $h = \frac{\lambda}{2}d$。

$$\frac{\partial g(i,q)}{\partial q} = \frac{\lambda}{2} \cdot \frac{d}{m-i}$$

由于 $f(i,\ q) = h$ 相对于 q 是常数（当 $g(i,\ q) \leq h$ 时），因此我们得到：

$$\frac{\partial f(i,q)}{\partial q} = \begin{cases} 0, & \text{if } g(i,q) \leq h \\ \frac{\lambda}{2} \cdot \frac{d}{m-i}, & \text{otherwise} \end{cases} \tag{14}$$

因此，$f(i,\ q)$ 的值及其导数均为正。结合这一点和公式（13），我们得到：

$$\frac{dI(q)}{dq} = -f(m-q,q) - \int_0^{m-q} \frac{\partial f(i,q)}{\partial q}\, di < 0$$

因此，$I(q)$ 随 q 单调递减。由此可直接看出，公式（11）随 q 单调递减，因为附加项 $(\lambda - 1)dm$ 相对于 q 是常数。

要证明观察28，我们必须证明公式（12）的导数为正。

$$\frac{dI(q)}{dq} + \lambda d > 0 \tag{15}$$

我们继续对第一部分进行进一步分析。

$$\begin{aligned} \frac{dI(q)}{dq} &= -f(m-q,q) - \int_0^{m-q} \frac{\partial f(i,q)}{\partial q}\, di \\ &\geq -f(m-q,q) - \int_0^{m-q} \frac{\lambda}{2} \cdot \frac{d}{m-i}\, di && \text{(Equation (14))} \\ &= -f(m-q,q) - \frac{\lambda}{2} d \cdot \ln\left(\frac{m}{q}\right) \\ &\geq -f(m-q,q) - \frac{\lambda}{2} d \cdot \ln\left(\frac{\lambda}{\lambda-1}\right) && (q \geq \frac{\lambda-1}{\lambda} m;\ \text{Observation 28}) \end{aligned} \tag{16}$$

让我们更仔细地分析 $f(i,\ q)$。

子情形 4.0.1 ($\frac{\lambda}{2}d \geq \lambda d - {}^{dm-\frac{\lambda}{2}d}{}_{(m-q)\,m-i}$)：在此情形下，$f(m-q, _q) = \frac{\lambda}{2}d$。因此，根据方程 (16)，我们得到：

$$\begin{aligned} \frac{dI(q)}{dq} + \lambda d &> -\frac{\lambda}{2}d - \frac{\lambda}{2} d \cdot \ln\left(\frac{\lambda}{\lambda-1}\right) + \lambda d && \text{(Equation (16))} \\ &= \frac{\lambda}{2}d - \frac{\lambda}{2} d \cdot \ln\left(\frac{\lambda}{\lambda-1}\right) \\ &> 0 && (\lambda \leq \frac{3}{2}) \end{aligned}$$

因此，我们证明了这种情况下的公式（15）。

子情况 4.0.2（$\frac{\lambda}{2}d \leq \lambda d - \frac{dm - \frac{\lambda}{2}d}{(m-q)\,m-i}$）：在这种情况下：

$$\begin{aligned}
\frac{dI(q)}{dq} + \lambda d &> -f(m-q,q) - \frac{\lambda}{2}d \cdot \ln\left(\frac{\lambda}{\lambda-1}\right) + \lambda d && \text{(Equation (16))} \\
&= -\left(\lambda d - \frac{dm - \frac{\lambda}{2}d(m-q)}{q}\right) - \frac{\lambda}{2}d \cdot \ln\left(\frac{\lambda}{\lambda-1}\right) + \lambda d \\
&= \frac{dm\,(2-\lambda)}{2q} + \frac{\lambda}{2}d - \frac{\lambda}{2}d \cdot \ln\left(\frac{\lambda}{\lambda-1}\right) \\
&> \frac{dm\,(2-\lambda)}{2q} \\
&> 0 && (\lambda < 2)
\end{aligned}$$

因此，在此情形下，公式（15）同样成立。

由于在这两种情形下，导数在区间 $[\frac{\lambda-1}{\lambda}m, m]$ 内均为正，观察28成立。

根据观察 27 和观察28，我们知道函数 $W(q) = \max(a(q)\ b(q))$ 在第二个 max 项的切换点处取得最小值，该点为 $q = \frac{\lambda-1}{\lambda}m$。

D.4 情况 1 ($m < 6$)

在此证明情形下，我们假设 $m < 6$。对于这些实例，我们明确计算整体调度工作量的下界，以证明其与 $\mathcal{W}_1 + \mathcal{W}_2 \leq dm$ 相矛盾。

为简化本情形中数值的计算，我们将使用 $\lambda = 1.45$；这比我们需要证明的结论稍强一些。

我们先通过以下观察为证明做准备：

▶ **观察 29**。若 $\lambda \geq 1.45$，则 S_1 中的空闲机器总数少于$q < \frac{11}{29}m$。

证明。注意，根据$\mathcal{W}_2 > \frac{145}{100}dq$，由引理 9.(ii) 可知。又由$\mathcal{W}_1 + \mathcal{W}_2 \leq dm$ 以及引理 9.(iii)，我们得到：

$$\begin{aligned}
&\mathcal{W}_1 + \mathcal{W}_2 \leq dm \\
\Leftrightarrow\ &\frac{145}{200}d(m_1 - q) + \frac{145}{100}dq < dm \\
\Leftrightarrow\ &\frac{145}{200}q < \frac{55}{200}m \\
\Leftrightarrow\ &q < \frac{55}{145}m = \frac{11}{29}m
\end{aligned}$$

◀

若 j_0 无法放置在 $m-i$ 台机器上，则机器 $i \in \{0, \ldots, m-1\}$ 上的负载 L_i 至少为

$$L_i > \begin{cases} g(i) & \text{if } g(i) \geq \frac{\lambda}{2}d \\ \max\{\lambda d - L_{i-1}, g(i)\} & \text{otherwise} \end{cases}$$

表 2 展示了所有 $m \in \{3,4,5\}$ 情况下的最小工作量，其中 $\mathcal{W}_1 = \sum_i L_i$，且 $\mathcal{W}_2 = \max\{0.45dm, 1.45dq\}$。注意，$q = 1$ 和 $m > 2$，因为 $m/6 < q < \frac{11}{29}m$（观察 29）。

m, q	最小机器负载 (L_i)	$(\mathcal{W}_1 + \mathcal{W}_2)/m$
3, 1	['0.93333', '0.67500', '0.00000']	1.01944
4, 1	['0.99375', '0.84167', '0.60833', '0.00000']	1.06094
5, 1	['1.03000', '0.92500', '0.75000', '0.70000', '0.00000']	1.13100

表 2 $m < 6$的数值分析。所有数值均为 d 的倍数，矛盾值以带下划线标出。

E 添加小作业

为完成对偶近似，我们需要加入之前忽略的小作业 J_S。为此，我们回顾一下，所得层架调度中的总工作量至多为 $md - \mathcal{W}_S$。

算法 5 添加小作业

要求： 的货架调度1: **对于每个** $j \in J_S$ 2: 调度作业 j 到负载最小的机器上。3: **结束循环**

▶**引理30**。 给定一个针对 J_B 的可行连续调度，其长度至多为 λd ，且工作面积至多为 $md - W_S$，对于任意 $\lambda \geq \frac{10}{7}$ ，算法 *5* 都能为整个实例 $J_B \cup J_S$生成一个可行连续调度，其完工时间至多为 λd 。 —

证明。 考虑一个如第2节所构造的调度。我们将货架 1 中的所有作业安排在0 开始，并安排第 2 层货架中的所有作业，使其恰好在 λd时完成。这样，每台机器上的空闲时间都是连续的。

每台机器的负载定义为调度到该机器上的作业的执行时间之和。根据定义，任意机器 i 的空闲时间等于 λd 减去该机器上的负载。

现在假设算法 5 未能生成可行的 λd 调度。在这种情况下，添加一个小作业后，必然存在某台机器 i 的负载严格大于 λd 。这意味着，在添加该小作业之前，该机器上的负载大于 d，因为根据定义，所有小作业在一台机器上的执行时间最多为 $\frac{3}{7}d$ 。由于 i 是负载最小的机器，这与该调度的总工作面积至多为 md的假设相矛盾。◀