正体追逐

Sayan Bhattacharya* Niv Buchbinder† Roie Levin†‡ Thatchaphol Saranurak §

arXiv:2304.01889v2 [cs.DS] 6 May 2024

摘要

我们研究在 ℓ_1 中追逐正体的问题：给定在线揭示的一系列体 $K_t = \{x^t \in \mathbb{R}^n_+ \mid C^t x^t \geq 1, P^t x^t \leq 1\}$，其中 C^t 和 P^t 是非负矩阵，目标是（近似地）维护一个点 $x_t \in K_t$，使得 $\sum_t \|x_t - x_{t-1}\|_1$ 最小化。该问题刻画了任何可表示为混合装箱-覆盖线性规划的问题的全动态低重配置变体，因此也涵盖了动态算法中许多核心问题的分数版本，例如集合覆盖、负载均衡、超边定向、最小生成树和匹配。

我们为该问题给出一个 $O(\log d)$-竞争算法，其中 d 是任意矩阵 C^t的最大行稀疏度。该算法绕过了已知的一般凸体追逐下界 $\sqrt{n}$，并将其指数级改进。我们的算法基于迭代信息投影；与一般凸体追逐算法不同，它完全不依赖记忆。

我们还展示了如何动态舍入我们的解，从而为上述所有问题获得首批具有竞争性重配置的全动态算法；也就是说，对于每个更新序列，其重配置量（允许相差多项式对数因子）都小于任何其他算法的重配置量。这一概念比动态算法文献中的绝对重配置概念强得多。

*华威大学计算机科学系。电子邮件：S.Bhattacharya@warwick.ac.uk。研究获英国工程与自然科学研究委员会（EPSRC）资助，项目编号 EP/S03353X/1。

†特拉维夫大学数学科学学院统计与运筹学系，特拉维夫。电子邮件：niv.buchbinder@gmail.com、roiel@tauex.tau.ac.il。研究部分获以色列科学基金会项目 2233/19 和美以两国双边科学基金会项目 2022418 资助。

‡研究部分获富布赖特以色列博士后奖学金资助。§密歇根大学计算机科学与工程系。电子邮件：thsa@umich.edu。研究获 NSF CAREER 项目 2238138 资助。

1 引言

我们研究正体追逐问题，该问题在 ℓ_1 中定义如下。给定在线揭示的一系列正体 $K_t = \{x^t \in \mathbb{R}^n_+ \mid C^t x^t \geq 1, P^t x^t \leq 1\}$，其中 C^t 和 P^t 是具有非负元素的矩阵。目标是（近似地）维护一个点 $x^t \in K_t$，使总的 ℓ_1 移动量 $\sum_t \|x^t - x^{t-1}\|_1$ 最小，其中 $x^0 = 0$。更一般地，给定权重 $w \in \mathbb{R}^n_+$，我们希望使加权的 ℓ_1 移动量 $\sum_t w_i \sum_{i=1}^n |x_i^t - x_i^{t-1}|$ 最小。这涵盖了任何可表示为混合装箱-覆盖线性规划的问题的全动态变体，因此也涵盖了许多核心全动态问题的分数版本，其中目标是最小化重配置，即以 ℓ_1 移动量衡量的解的变化量。例子包括集合覆盖、负载均衡（超边定向）、最小生成树和匹配。

我们问题的一个更一般版本称为凸体追逐问题，允许 K_t 为任意凸体，目标是对任意 $p \geq 1$ 最小化总 ℓ_p 移动量。Fried- man 和 Linial [FL93] 将凸体追逐引入为众多在线问题的广泛推广。近年来，这一问题一直受到深入研究 [BBE+20、ABC+19、BKL+20]，最终产生了具有 $O(n)$ 竞争比的算法 [AGTG21、Sel20]。这一界几乎是紧的，因为存在 $\Omega(\max\{\sqrt{n}, n^{1-1/p}\})$ 的下界 [BKL+20]。遗憾的是，由于这一强下界，针对这一一般问题的算法并未为具体的组合优化问题带来有意义的应用。

在本文中，我们证明，对于 ℓ_1 中正体追逐这一特殊但仍具表达力的情形，可以绕过并将 $\sqrt{n}$ 下界改进一个指数级的幅度。随后我们证明，正体追逐问题的一个解可以在线舍入，从而为上述所有组合问题导出低重配置的全动态算法。我们的算法具有竞争性重配置 保证，这是一种显著强于通常重配置概念的性质；动态算法文献中通常使用后者，我们将在下文很快讨论。

1.1 我们的结果

我们的主要贡献是以下定理。

定理 1.1。 对于任意 $\epsilon \in (0, 1]$，存在一个 $O(1/\epsilon \text{ 对数 } (d/\epsilon))$-竞争算法，用于在 ℓ_1 中追踪正体；该算法满足：在时间 t 时，$x^t \in K_t^{1+\epsilon} = \{x^t \in R^n_+ \mid C^t x^t \geq 1, P^t x^t \leq 1 + \epsilon\}$，且 d 是覆盖约束中非负系数的最大数量。

请注意，我们的算法采用 ϵ 资源增强，即允许其略微违反装箱约束。或者，通过对解进行缩放，我们可以生成一个完全满足所有装箱约束、但将覆盖约束违反至多 ϵ 倍的解。正如我们将在下文讨论的，ϵ 资源增强在我们的应用中并不重要，因为我们在舍入过程中无论如何都会损失额外的近似因子。我们将算法维持的点在 $K_t^{1+\epsilon}$ 中的总移动量，与最优解维持的点在 K_t 中的总移动量进行比较。在若干满足 $d = O(1)$ 的应用中，例如动态边定向的分数版本以及有界频率的集合覆盖问题，我们的竞争比完全独立于 n。我们还指出，对于形如 $x \leq b$ 的静态盒约束，其中 $b \in R^n_+ \cup \{\infty\}$，我们可以在不产生任何违反的情况下处理它们。更多细节请参见附录 F。

我们通过以下下界补充定理1.1：

定理 1.2. 正体追逐不存在竞争比优于 n

$$\Omega\left(\max\left\{\min\left(\frac{1}{\epsilon\sqrt{\log(1/\epsilon)}}, \sqrt{n}\right), \log n\right\}\right).$$

该$\Omega(\log n)$下界可由覆盖线性规划的已知下界推出 [BN09b、GKL21]。该$\Omega(\min(1/(\epsilon\sqrt{\log(1/\epsilon)}), \sqrt{n}))$下界可通过归约到$\sqrt{n}$一般凸体追逐问题的下界（《[BKL+20]》中的定理 5.4）[1]得到。完整证明见附录E。

这个下界意味着，定理 1.1 中的所有假设对于相较于一般凸体追逐情形取得指数级改进都至关重要。特别是，这表明一般凸体追逐问题中$\Omega(\sqrt{n})$的下界，同样适用于不带资源增强的正体追逐问题。更一般地，我们对$1/\epsilon$的线性依赖基本上是紧的。例如，当$\epsilon = \Theta(n^{-\delta})$对某个常数$0 < \delta < 1/2$成立时，竞争比至少为$\Omega(n^{\delta})$（省略对数项），因此不可能达到$O(\text{poly log } n)$的竞争比。当$\epsilon = O(\frac{\log n}{n})$时，针对一般凸体追逐的最佳算法仍是$O(n)$-竞争算法 [AGTG21, Sel20]。最后，我们指出，对于ℓ_p-范数，当$p > 1$时，已知下界 [FL93] 排除了任何$o(n^{1-1/p})$-竞争算法，即使是追逐正体且具有$\Omega(1)$的资源增强也不例外。

动态问题的竞争性重配置。在线和动态算法的一个总体目标，是在约束随时间变化时，维持组合问题的近似最优解，同时尽量减少对解进行编辑的次数，即重配置。在这方面，大量研究已为以下问题提出了低重配置算法：终端更新下的斯坦纳树 [IW91, GGK16, GK14, ŁOP+15]；作业到达下的负载均衡 [AAPW01, GKS14, KLS23]，；元素插入/删除下的集合覆盖 [GKKP17, AAG+19, BHN19, GL20a, BHNW21, AS21]，；客户端更新下的设施选址 [BLP22, GKLX20]，；以及许多在边更新下的全动态图问题，例如边定向 [BF99, SW20a, BBCG22]、图着色 [SW20b]，、极大独立集 [AOSS18]，和稀疏化器 [BKS12, BSS22]。

上述研究以及大多数其他关于动态算法的研究，都以绝对量衡量重配置。此类界的形式为：“经过T次更新后，算法产生的重配置至多为kT”。我们将其称为绝对重配置界。另一方面，在线算法领域更倾向于采用竞争性分析，即将算法的表现与事后看来最优的算法进行比较。本文给出了竞争性重配置界：若某算法在每个更新序列上的重配置至多为任意其他离线算法的c倍，我们就称该算法具有c-竞争性重配置。

研究竞争性重配置相较于绝对重配置有几项优势。绝对重配置界是针对所有更新序列的最坏情况界，而竞争性重配置界则针对每个更新序列量身定制。举例来说，考虑集合覆盖中的一个更新序列：反复插入并删除同一个元素，却不改变最优解。竞争性重配置算法不会产生任何重配置，而具有绝对重配置保证的算法则可能因这些“无关”更新而产生大量重配置。因此，竞争性重配置界可能远强于绝对界。举一个具体例子，通过消除较短的增广路径，可以直接得到一个针对$(1+\epsilon)$-近似匹配的全动态算法，其绝对重配置为$O(1/\epsilon)$；然而，在我们的工作之前，还没有已知的具有竞争性重配置的非平凡算法。

对于某些问题，绝对重配置可能根本无法做到很小。在全动态负载均衡的背景下，[KLS23]证明，即使掌握作业到达与离开的完整序列，任何算法也无法在绝对重配置少于$\alpha = o\Omega(\log(n))$的情况下达到$\Omega(n^{1/\alpha})$的竞争比。他们写道：

> “要规避这一负面结果，就需要为全动态模型采用不同的重配置衡量方式。”

[1]我们感谢 Mark Sellke 指出了这一归约。

We 本文认为，竞争性重配置是克服这一障碍的自然方式 e.

定理的应用 1.1 于组合问题。 本文首次为动态算法文献中研究的若干核心问题（包括集合覆盖、负载均衡、二分图匹配和最小生成树）提供了具有多对数级竞争性重配置的全动态算法。结果汇总于表 1。表中，我们的算法具有 α 近似保证和 c 竞争性重配置保证，具体保证如下。对于每个 更新序列，我们的算法都能维护一个 α 近似解，且其重配置成本至多为维护最优解所需最优重配置成本的 c 倍。更一般地，我们的每种算法甚至都支持以下权衡。给定参数 β，对于每个 更新序列 σ，算法都能维护一个 $(\alpha\beta)$ 近似解，且总补救成本至多为 $c \cdot \mathsf{Opt}^{\beta}{}_{\textsc{Rec}}(\sigma)$，其中 $\mathsf{Opt}^{\beta}{}_{\textsc{Rec}}(\sigma)$ 表示在经历更新 σ 的情况下，维护一个 β 近似解所需的最优重配置成本。这一点很有用，因为我们自然会预期，允许一定程度的近似时，最优重配置成本会显著降低。

问题	近似	竞争性 重配置	随机化 vs. 确定性	Ref.	更新 type	变量
集合覆盖	$O(\log\ n)$	O(日志 n日志 f)	随机化	参见。 4.2	元素	n 元素， 每个包含 $\leq f$ 个元素的集合
	$O(f)$	$O(f$ 对数 $f)$	确定性	相关。 4.2		
Load 平衡	$2+\epsilon$	$O\left(\frac{1}{\epsilon^5}\text{日志}\ n \cdot \frac{r}{\epsilon}\ \text{日志 日志}\ \frac{1}{\epsilon}\right)$	确定性	相关。 4.6	jobs	n个作业，各自 适用于 $\leq r$ 台机器
	$O\left(\frac{\text{对数 对数 } n \text{ 对}}{\text{数 对数 对数 } n}\right)$	O(对数 r)	随机化	相关。 4.6		
二分图 匹配	$1+\epsilon$	$O\left(\frac{1}{\epsilon^4}\text{对数 } n\text{对数 } \frac{n}{\epsilon}\right)$	随机化	相关系数。 4.4	边	n 顶点
最小 生成树 tree	$2+\epsilon$	$O\left(\frac{1}{\epsilon^3}\text{对数 } n\text{对数 } \frac{n}{\epsilon}\right)$	随机化	柯林多后书。 4.8	边	n 顶点

表 1： 具有竞争性重配置的全动态算法概要。所有随机化算法均假设存在无适应对手，并在调整代价界中产生一个多项式级小的加性项，此处省略。详情请参见第4 节。

我们的分数算法的一个重要特点是，其保证与动态更新的类型无关，因为只要每一步的可行域构成一个正体，定理 1.1 就成立。例如，我们的分数最小生成树和二分图匹配算法适用于顶点和边到达/离开两种情况；我们的分数负载均衡算法适用于作业-机器负载的细粒度更新；我们的分数集合覆盖算法则适用于集合成本可更新的情况。由于我们的舍入方案也适用于这些广义的更新类型，因此最终算法同样适用。

我们强调，我们关注的是最小化重配置，而非更新时间；后者是动态算法文献中通常采用的度量指标。如何同时实现竞争性重配置和较短的更新时间，是一个值得研究的方向。

1.2 与以往研究的联系

正体追逐。 我们关于正体追逐的结果，直接推广了在线覆盖问题领域中一系列颇具影响力的研究 [BN09b、GN14、GL20b]。在这些研究中，凸体彼此嵌套，且仅由覆盖约束定义：对于每个时刻 t，相应的正体为 $K_t = \{x \in R_+^n \mid C^t x \geq 1\}$ 其中 C^t 是一个非负矩阵，且 $K_t \subseteq K_{t-1}$。任务是在最小化 $\langle w$的同时，维持 $x^t \in K_t$ 单调递增 $x\rangle$，其中 $w \in \mathbb{R}_+^n$固定。这个目标等价于最小化 ℓ_1按 w加权的移动量，因为减小任何

变量都无助于满足覆盖约束。即便是正体追逐的这一特殊情形，也极为成功地统一了以往的结果，并解决了竞争分析中的重要开放问题。例如，这包括经典的滑雪租赁问题、在线集合覆盖 [AAA+09]、加权分页及其变体 [BBN12a, BBN12b, CLNT22]，与连通性和割相关的图优化问题 [AAA+06, NPS11]，动态 TCP 确认问题 [KKR01, BJN07],度量任务系统 [BBN10, BCLL21]，以及k-服务器问题 [BCL+18, BGMN19]。

另一类研究 [ABFP13、ABC+16] 以另一种方式推广了 [BN09b、GN14] 。在这里，凸体 K_t 是嵌套覆盖约束，且 $x^t \in K_t$ 只能单调移动。 [ABFP13] 的目标是最小化一组固定装箱约束的最大违反程度，而 [ABC+16] 的目标是最小化一个非递减凸函数。这与我们的目标值不同，因为我们必须处理装箱约束和覆盖约束组成的在线序列，但我们的解 x 可以非单调移动。

[BKS23] 提出了用于维护混合装箱-覆盖线性规划$(1+\epsilon)$近似解的动态算法。他们展示了如何以较短的更新时间处理放宽更新（可行域只会扩大），并提出一个开放问题：对于限制更新（可行域只会缩小），是否也能做到这一点。定理 *1.1* 解决了这个问题的重配置版本：我们可以维护混合装箱-覆盖线性规划的()近似解，且总绝对重配置量为 (log)，其中 是限制更新序列结束时的任意可行解。这是因为离线算法只需从一开始就将解移至 。

组合问题中的竞争性重配置。 这里，我们只将算法与表 1 中少数已知的、具有竞争性重配置保证的动态算法进行比较。

对于集合覆盖问题， [GN14] 中的在线算法支持元素插入，并保证 $\min\{O(\log n \log f)\ O(f\log f)\}$近似，这是目前可达到的最佳结果。当每个集合的权重均为 1 时，近似比也对应于调整代价界，因为没有集合会从解中移除。我们的算法（推论 4.2）是首个全动态竞争性重配置算法，即使在无权情形下也是如此。

对于负载均衡问题，即使聚焦于特殊情形，我们的算法也带来了有趣的推论；推论 4.6 给出了加权图全动态边定向的首个竞争性重配置算法。[2] 此前，这一结果仅对无权图已知 [BF99]。在无权情形下，我们的算法具有 $O(\log r)$竞争性重配置，是首个在秩 为常数 且 $r = O(1)$时，实现常数竞争性重配置和非平凡近似的全动态超边定向算法。此前 [BBCG22] 的结果在重配置方面存在 $O(\log n)$ 因子。

[GTW14] 提出了一种算法，用于在边具有获取成本和随时间变化的持有成本时维护生成树（更一般地说，拟阵基）。他们的结果可用于获得一种 $O(\log^2 n)$ 竞争性重配置算法，以便在边更新期间维护生成树，但无法保证树在每一步的成本。相比之下，我们的算法（推论 4.8）保证提供 $(2+\epsilon)$ 近似，且具有 $O_\epsilon(\log^2 n)$竞争性重配置。

据我们所知，除此之外，唯一具有竞争性重配置保证的动态算法是[AMPST23] ，用于顶点着色；以及 [ALPS16、ABL+20] ，用于平衡图划分。我们希望通用框架能推动这一方向取得更多成果。

[2]这对应于这样的负载均衡实例：每个作业都适用于两台机器。

1.3 技术与概述

我们的核心技术贡献是一种用于追逐正半空间的算法，也适用于由单个覆盖约束或装箱约束定义的正体（我们在附录A中说明了从追逐正体到追逐半空间的归约）。我们的算法如下：给定上一轮的点 x^{t-1} ，算法将其投影到新的可行域上，所采用的度量是 KL 散度（出于技术原因，还会进行一次小幅加性更新）。这有时也称为信息投影（例如[Csi84]）。许多在线覆盖算法都可以解释为信息投影 [BGMN19]；我们的贡献是将分析扩展到装箱约束，而这此前一直难以实现。有趣的是，与一般凸体追逐的竞争性算法（[AGTG21、Sel20]）相比，该算法完全不依赖记忆，只取决于 x^{t-1} 和当前违反的约束。

为了分析该算法，我们写出一个辅助线性规划，其最优值是一个已知所有未来约束的离线算法的最优重配置，然后为我们的在线算法解拟合一个对偶解。我们利用用于计算投影的凸规划的拉格朗日乘子来构造这个对偶解。我们首先在第 3.1 节至3.3节进行预热分析。这里的证明很简单，包含了主要思想，但会额外损失一个纵横比项。为了得到定理1.1的完整结果，我们需要克服一个重要障碍：可以证明，任何单调对偶解都无法避免对纵横比的依赖（参见例如 [BN09a]）。尽管如此，我们证明，通过谨慎地减小对偶解的某些坐标，可以达到所需的界。详情见第 3.4节。

在第 4节，我们将竞争性分数解舍入，以应用于常见问题。我们的正体追逐框架涵盖了在动态更新下维护集合覆盖（第 4.1节）、二分图匹配（第 4.2节）、负载均衡（第 4.3节）和最小生成树（第 4.4节）的分数解这一任务。我们证明，还可以将这些分数解舍入为真正的组合解，并且在重配置和近似方面的损失均有界。第 4.2 节和4.4 节中的舍入算法采用了一个共同的步骤：我们首先维护一个称为稳定器的对象，其中包含一个自身具有良好竞争性重配置的优质解（详情见附录 C和附录 D ）。然后，我们运行现有的绝对重配置算法，在该稳定器中维护一个真正的解。由于对稳定器的更新次数与最优离线解的重配置相比具有竞争性，而绝对重配置算法相对于稳定器的更新次数也是竞争性的，因此总体而言，这一策略能够产生具有良好竞争性重配置的解。

2 预备知识

数学符号。 本文中所有对数均以 e 为底。对于 $w \in \mathbb{R}$，我们用记号 $w_+ = \max(0, w)$ 表示。我们用符号 $\oplus$ 表示集合的对称差。在以下定义中，设 $x, y \in \mathbb{R}^n_+$ 为向量。 x 与y 之间的标准点积记作 $\langle x, y \rangle = \sum_{i=1}^n x_i y_i$。我们使用 KL 散度的加权推广。给定权重函数 w，定义

$$\mathrm{KL}_w(x \,||\, y) := \sum_{i=1}^{n} w_i \left[x_i \log\left(\frac{x_i}{y_i}\right) - x_i + y_i \right].$$

已知，对于非负向量 x、 y，$\mathrm{KL}_w(x \,||\, y) \geq 0$ （可以逐项检查上面的结果，确认其成立）。本文始终约定，对于任意 a、 $b \geq 0$，表达式

$$(a + \infty) \log\left(\frac{a + \infty}{b + \infty}\right) - a + b = 0. \tag{2.1}$$

这些定义与相应的极限行为一致。

简化正体追逐实例。 在正体追逐问题中，由于$x^0 = 0$ 和 $x^t \geq 0$，因此我们有

$$\sum_{t\in[T]}\sum_{i\in[n]} w_i(x_i^t - x_i^{t-1})_+ \leq \sum_{t\in[T]}\sum_{i\in[n]} w_i \cdot |x_i^t - x_i^{t-1}| \leq 2 \cdot \sum_{t\in[T]}\sum_{i\in[n]} w_i(x_i^t - x_i^{t-1})_+.$$

因此，不失一般性，我们使用向上补救 来衡量我们的算法和最优算法各自的移动量。

此后，不失一般性，我们假设在每个时刻 t，正体 K_t由单个覆盖约束 $\langle c^t、\ x\rangle \geq 1$ 或装箱约束 $\langle p^t、\ x\rangle \leq 1$定义。这种从追逐凸体到追逐半空间的归约是标准做法（例如参见 [BBE+20]），但为完整起见，我们在附录 A 中阐述主要论证。

令 $c_i^{\max} = \max_t c_i^t$, $c_i^{\min} = \min_{t|c_i^t\neq 0} c_i^t$。我们将纵横比定义为 $\Delta = \max_i c_i^{\max}/c_i^{\min}$。令 $d^t = |\{i \mid c_i^t \neq 0\}|$ 为覆盖约束的稀疏度。令 $d = \max_t\{d^t\}$ 为任意覆盖约束的最大行稀疏度。

3 一种追逐正体的算法

本节中，我们设计一种用于追逐正体的分数算法，以证明定理1.1。如第2 节所述，我们不失一般性地假设，在每个时刻 t正体 K_t 均由单个覆盖约束或装箱约束定义。

3.1 算法：

该算法以参数 $\epsilon \in (0, 1]$作为输入。每当遇到一个违反的约束时，它就按照以下步骤投影回该约束。

- 初始时，$x_i^0 = 0$对所有 i均成立。在任意时刻 $t = 1$，2，...，T:
- 当一个违反的覆盖约束 $\langle c^t, x\rangle < 1$ 到达时，令 x^t 为以下方程的解:

$$\min_x \sum_{i|\, c_i^t\neq 0} w_i \cdot \left[\widehat{x}_i \log\left(\frac{\widehat{x}_i}{\widehat{x}_i^{t-1}}\right) - \widehat{x}_i\right]$$
$$\text{s.t.} \qquad \langle c^t, x\rangle \geq 1 \qquad (I).$$

其中 $\widehat{x}_i := x_i + \frac{\epsilon}{4d^t\cdot c_i^t}$、 $\widehat{x}_i^{t-1} := x_i^{t-1} + \frac{\epsilon}{4d^t\cdot c_i^t}$，而 $d^t = |\{i \mid c_i^t \neq 0\}|$ 是覆盖约束的稀疏度。[3] 如果对于某个 x_i^t，系数 $c_i^t = 0$，那么我们令 $x_i^t = x_i^{t-1}$。

- 当给定一个被违反的装箱约束 $\langle p^t$ 时，$x\rangle > 1 + \epsilon$，令 x^t 为以下问题的解:

$$\min_x \sum_{i|\, p_i^t\neq 0} w_i\left(x_i^t \log\left(\frac{x_i}{x_i^{t-1}}\right) - x_i\right)$$
$$\text{s.t.} \qquad \langle p^t, x\rangle \leq 1 + \epsilon \qquad (II).$$

如果对于某个 x_i^t，系数 $p_i^t = 0$，那么我们令 $x_i^t = x_i^{t-1}$。

不失一般性地，我们假设只会遇到违反约束的情况。令 C、 $P \subseteq \{1, \ldots, T\}$分别为覆盖约束和装箱约束出现的时间集合。

[3]注意：在时刻 t，我们重新定义 $\widehat{x}_i^{t-1}$，尽管它已于 $t-1$ 定义为 $x_i^{t-1} + \epsilon/(3d^{t-1}\cdot c_i^{t-1})$。这稍微滥用了记号，但能使我们的证明更简洁，也更清楚地体现与 KL 散度的联系。

3.2 分析框架

令 y^t、 z^t 为上述约束（I）和（II）的拉格朗日乘子。在证明中，对于每个时间步 t，我们使用该程序最优解处的 KKT 条件：

$$\forall t \in C \qquad \left\langle c^t, x^t \right\rangle = 1 \tag{3.1}$$

$$\forall t \in C,\ \forall i, c_i^t \neq 0 \qquad w_i \log\left(\frac{\widehat{x}_i^t}{\widehat{x}_i^{t-1}}\right) = c_i^t y^t \tag{3.2}$$

$$\forall t \in P \qquad \left\langle p^t, x^t \right\rangle = 1 + \epsilon \tag{3.3}$$

$$\forall t \in P,\ \forall i \qquad w_i \log\left(\frac{x_i^t}{x_i^{t-1}}\right) = -p_i^t z^t \tag{3.4}$$

为方便起见，我们定义 $y^0 = z^0 = 0$。

最优向上补救可通过以下线性规划 $(\overline{\mathcal{P}})$ 计算。

$$(\overline{\mathcal{P}}):\ \min \sum_{t=1}^{T}\sum_{i=1}^{n} w_i \overline{\ell}_i^t$$

s.t.

$$\forall t \in C: \quad \left\langle c^t, \overline{x}^t \right\rangle \geq 1$$

$$\forall t \in P: \quad \left\langle p^t, \overline{x}^t \right\rangle \leq 1$$

$$\forall i, t: \quad \overline{x}_i^t - \overline{x}_i^{t-1} \leq \overline{\ell}_i^t$$

$$\overline{x}_i^t, \overline{\ell}_i^t \geq 0$$

$$(\overline{\mathcal{D}}): \max \sum_{t \in C} \overline{y}^t - \sum_{t \in P} \overline{z}^t$$

s.t.

$$\forall i, t \in C: \quad c_i^t \overline{y}^t - \overline{r}_i^t + \overline{r}_i^{t+1} \leq 0$$

$$\forall i, t \in P: \quad -p_i^t z^t - \overline{r}_i^t + \overline{r}_i^{t+1} \leq 0$$

$$0 \leq \overline{r}_i^t \leq w_i$$

$$\forall t: \quad \overline{y}^t, \overline{z}^t \geq 0$$

从现在起，我们的目标是构造对偶规划（$\overline{\mathcal{D}}$）的一个可行解，为我们的在线算法的总成本提供上界。

我们先给出一个比定理 1.1 中的界更弱的界，作为预热。

3.3 预热：一个 $O\left(1/\epsilon \cdot \log(d\Delta/\epsilon)\right)$ 竞争界

本节中，我们给出算法重配置量的第一个界。

定理 3.1 (预热界)。 该算法的总重配置成本至多为目标值的 $O\left(\frac{1}{\epsilon} \cdot \log\left(\frac{d \cdot \Delta}{\epsilon}\right)\right)$倍，其中目标值为$(\overline{\mathcal{D}})$。

回顾一下， $c_i^{\max} = \max_t c_i^t$ 和 $c_i^{\min} = \min_{t | c_i^t \neq 0} c_i^t$， 以及纵横比 $\Delta = \max_i c_i^{\max}/c_i^{\min}$。我们构造如下对偶解。

令 $A = \log\left(1 + \frac{4d \cdot \Delta}{\epsilon}\right)$。我们设

$$\begin{aligned}
\bar{y}^\tau &= y^\tau / A, \\
\bar{z}^\tau &= z^\tau / A, \\
\overline{r}_i^\tau &= w_i \cdot \left(1 - \frac{1}{A} \log\left(1 + \frac{4dc_i^{\max} \cdot x_i^{\tau-1}}{\epsilon}\right)\right)
\end{aligned}$$

（这有助于将其写成）

$$= w_i \cdot \left(1 - \frac{1}{A}\left(\log\left(x_i^{\tau-1} + \frac{\epsilon}{4d \cdot c_i^{\max}}\right) - \log\left(\frac{\epsilon}{4d \cdot c_i^{\max}}\right)\right)\right). \tag{3.5}$$

我们首先证明我们的对偶问题是可行的。

引理 3.2。 变量$(\bar{y},\ \bar{z},\ \bar{r})$ 是$(\overline{\mathcal{D}})$的一个可行对偶解。

证明。 对于 $\tau \in C$，我们有

$$\overline{r}_i^{\tau} - \overline{r}_i^{\tau+1} = \frac{w_i}{A} \cdot \log\left(\frac{x_i^{\tau} + \frac{\epsilon}{4d \cdot c_i^{\max}}}{x_i^{\tau-1} + \frac{\epsilon}{4d \cdot c_i^{\max}}}\right) \geq \frac{w_i}{A} \log\left(\frac{x_i^{\tau} + \frac{\epsilon}{4d^{\tau} \cdot c_i^{\tau}}}{x_i^{\tau-1} + \frac{\epsilon}{4d^{\tau} \cdot c_i^{\tau}}}\right) = c_i^{\tau} \bar{y}^{\tau}.$$

该不等式成立，是因为对于任意 i 和 $\tau \in C$，我们有 $x_i^{\tau} \geq x_i^{\tau-1}$，因此，对于任意 $a \geq b$，不等式 $\frac{x_i^{\tau}+a}{x_i^{\tau-1}+a} \leq \frac{x_i^{\tau}+b}{x_i^{\tau-1}+b}$成立。类似地，对于 $\tau \in P$，我们有

$$\overline{r}_i^{\tau} - \overline{r}_i^{\tau+1} = \frac{w_i}{A} \cdot \log\left(\frac{x_i^{\tau} + \frac{\epsilon}{4d \cdot c_i^{\max}}}{x_i^{\tau-1} + \frac{\epsilon}{4d \cdot c_i^{\max}}}\right) \geq \frac{w_i}{A} \log\left(\frac{x_i^{\tau}}{x_i^{\tau-1}}\right) = -p_i^{\tau} \bar{z}^{\tau}.$$

该不等式成立，是因为对于任意 $\tau \in P$，我们有 $x_i^{\tau} \leq x_i^{\tau-1}$，因此，对于任意 $a \geq 0$，不等式 $\frac{x_i^{\tau}}{x_i^{\tau-1}} \leq \frac{x_i^{\tau}+a}{x_i^{\tau-1}+a}$ 成立。最后，

$$0 \leq \overline{r}_i^{\tau} = w_i \cdot \left(1 - \frac{1}{A} \log\left(1 + \frac{4dc_i^{\max} \cdot x_i^{\tau-1}}{\epsilon}\right)\right) \leq w_i.$$

要看出这一点，请注意，对于所有 $\tau \in C$ 和所有 $i \in [n]$，我们都有 $0 \leq x_i^{\tau} \leq \frac{1}{c_i^{\tau}}$；否则，第 τ 轮的约束在到达时就已经满足。因此

$$0 \leq \log\left(1 + \frac{4dc_i^{\max} \cdot x_i^{\tau-1}}{\epsilon}\right) \leq \log\left(1 + \frac{4dc_i^{\max}}{\epsilon c_i^{\tau-1}}\right) \leq \log\left(1 + \frac{4d \cdot \Delta}{\epsilon}\right) = A. \qquad \square$$

接下来，为了将算法的移动与对偶目标值联系起来，我们需要以下两个关键引理。首先，我们仅用 y 项来界定重配置。

引理 3.3（用 y 来界定移动）。 算法在任意时刻 $t \in C$ 的向上移动成本都可以界定为

$$\sum_i w_i \left(x_i^t - x_i^{t-1}\right)_+ \leq \left(1 + \frac{\epsilon}{4}\right) y^t.$$

证明。 由 $1+x \leq e^x$，取 $x = \log(b/a)$，可得以下一个便于使用的不等式（有时称为“穷人版 Pinsker 不等式”）。对所有 a、$b \geq 0$，

$$a - b \leq a \log\left(\frac{a}{b}\right).$$

利用这一事实，我们推导出

$$\begin{aligned}
\sum_i w_i \left(x_i^t - x_i^{t-1}\right)_+ &= \sum_{i|\, c_i^t \neq 0} w_i \left(\widehat{x}_i^t - \widehat{x}_i^{t-1}\right)_+ \\
&\leq \sum_{i|\, x_i^t > x_i^{t-1}} w_i \cdot \widehat{x}_i^t \cdot \log\left(\frac{\widehat{x}_i^t}{\widehat{x}_i^{t-1}}\right) \\
&\overset{(3.2)}{=} \sum_{i|\, x_i^t > x_i^{t-1}} \left(x_i^t + \frac{\epsilon}{4d^t \cdot c_i^t}\right) c_i^t y^t \\
&\leq \frac{\epsilon}{4} y^t + y^t \cdot \sum_i c_i^t x_i^t \\
&\overset{(3.1)}{=} \left(1 + \frac{\epsilon}{4}\right) y^t.
\end{aligned}$$

$\square$

为了将重配置界定为完整对偶目标值的函数，我们还需要论证，减去 z 项不会造成太大损失。

引理 3.4 (关于 z 的界，以 y表示)。 对偶向量 y 和 z 满足

$$\left(1+\frac{\epsilon}{4}\right)\sum_{t\in C} y^t - (1+\epsilon)\sum_{t\in P} z^t \geq 0 \tag{3.6}$$

证明。 由 KL 散度的非负性可得

$$\forall t\in C \quad 0 \leq \sum_{i|\, c_i^t\neq 0} w_i\left[\widehat{x}_i^t \log\left(\frac{\widehat{x}_i^t}{\widehat{x}_i^{t-1}}\right) - \widehat{x}_i^t + \widehat{x}_i^{t-1}\right] = \sum_{i|\, c_i^t\neq 0} w_i\left[\widehat{x}_i^t \log\left(\frac{\widehat{x}_i^t}{\widehat{x}_i^{t-1}}\right) - x_i^t + x_i^{t-1}\right]$$

$$\forall t\in P \quad 0 \leq \sum_{i|\, p_i^t\neq 0} w_i\left[x_i^t \log\left(\frac{x_i^t}{x_i^{t-1}}\right) - x_i^t + x_i^{t-1}\right].$$

对所有时刻 t 求和，得到:

$$\begin{aligned}
0 &\leq \sum_{t\in C}\sum_{i|\, c_i^t\neq 0} w_i\left[\widehat{x}_i^t \cdot \log\left(\frac{\widehat{x}_i^t}{\widehat{x}_i^{t-1}}\right) - x_i^t + x_i^{t-1}\right] + \sum_{t\in P}\sum_{i|\, p_i^t\neq 0} w_i\left[x_i^t \log\left(\frac{x_i^t}{x_i^{t-1}}\right) - x_i^t + x_i^{t-1}\right] \\
&= \sum_{t\in C}\sum_{i|\, c_i^t\neq 0} w_i\widehat{x}_i^t \cdot \log\left(\frac{\widehat{x}_i^t}{\widehat{x}_i^{t-1}}\right) + \sum_{t\in P}\sum_{i|\, p_i^t\neq 0} w_i \cdot x_i^t \log\left(\frac{x_i^t}{x_i^{t-1}}\right) - \sum_{i=1}^n w_i(x_i^T + x_i^0) \qquad (3.7) \\
&\overset{(3.2)\ \&\ (3.4)}{=} \sum_{t\in C} y^t \sum_{i|\, c_i^t\neq 0}\left(c_i^t x_i^t + \frac{\epsilon}{4d^t}\right) - \sum_{t\in P} z^t \sum_{i|\, p_i^t\neq 0} p_i^t x_i^t - \sum_{i=1}^n w_i x_i^T \\
&\overset{(3.1)\ \&\ (3.3)}{\leq} \left(1+\frac{\epsilon}{4}\right)\sum_{t\in C} y^t - (1+\epsilon)\sum_{t\in P} z^t. \qquad (3.8)
\end{aligned}$$

等式 (3.7) 由 (2.1) 得出。不等式 (3.8) 利用了这样一个事实：至多有 d^t 个索引i 满足 $c_i^t \neq 0$。 □

通过仔细结合引理 3.3 和引理 3.4，我们得到以下以对偶目标值表示的重配置上界。我们将引理写成关于更一般的 $\widetilde{y}^\tau$ 的形式，因为稍后会用到。现在，读者应令 $\alpha = 0$，即 $\widetilde{y}^\tau = y^\tau$。

引理 3.5。 设 $\widetilde{y}^\tau$ 满足 $\sum_{\tau\in C}\widetilde{y}^\tau \geq (1-\alpha\epsilon)\sum_{\tau\in C} y^\tau$，其中 $\alpha \in [0, 1/10]$。则

$$\sum_{t=1}^T\sum_i w_i\left(x_i^t - x_i^{t-1}\right)_+ = O\left(\frac{1}{\epsilon}\right)\left[\sum_{t\in C}\widetilde{y}^t - \sum_{t\in P} z^t\right] \tag{3.9}$$

证明。注意到

$$\begin{aligned}
\left(1+\frac{\epsilon}{4}\right)\sum_{t\in C} y^t &\leq \frac{1+\frac{\epsilon}{4}}{1-\alpha\epsilon}\cdot\sum_{t\in C}\widetilde{y}^t \\
&\leq \left(1+\frac{\epsilon}{4}\right)\cdot(1+2\alpha\epsilon)\cdot\sum_{t\in C}\widetilde{y}^t \\
&\leq \left(1+\frac{\epsilon}{4}+2\alpha\epsilon+\frac{2\alpha\epsilon}{4}\right)\cdot\sum_{t\in C}\widetilde{y}^t \\
&\leq \left(1+\frac{\epsilon}{2}\right)\cdot\sum_{t\in C}\widetilde{y}^t, \qquad (3.10)
\end{aligned}$$

其中，第一个不等式由我们的假设得出，第二个则是因为 $1/(1-x) \le 1+2x$对 $x \in [0, \frac{1}{2}]$ 和 $\epsilon \le 1$成立。
求和引理 3.3 对所有时刻求和 t并将 $\frac{2+\epsilon}{\epsilon}$ 引理 3.4的结果相加，我们得到：

$$\begin{aligned}
\sum_i w_i \left(x_i^t - x_i^{t-1}\right)_+ &\le \sum_{t\in C}\left(1+\frac{\epsilon}{4}\right)y^t + \frac{2+\epsilon}{\epsilon}\left[\sum_{t\in C}\left(1+\frac{\epsilon}{4}\right)y^t - \sum_{t\in P}(1+\epsilon)z^t\right] \\
&\le \sum_{t\in C}\left(1+\frac{\epsilon}{2}\right)\widetilde{y}^t + \frac{2+\epsilon}{\epsilon}\left[\sum_{t\in C}\left(1+\frac{\epsilon}{2}\right)\widetilde{y}^t - \sum_{t\in P}(1+\epsilon)z^t\right] \\
&= \frac{(2+\epsilon)(1+\epsilon)}{\epsilon}\left[\sum_{t\in C}\widetilde{y}^t - \sum_{t\in P} z^t\right] \\
&= O\left(\frac{1}{\epsilon}\right)\left[\sum_{t\in C}\widetilde{y}^t - \sum_{t\in P} z^t\right],
\end{aligned}$$

其中第二个不等式来自上面的（3.10）。 □
综合以上结果，我们可以得出本节的定理。
定理 *3.1* 的证明。 根据引理 3.5 我们可以将算法的总移动量界定为

$$\sum_{t=1}^{T}\sum_i w_i\left(x_i^t - x_i^{t-1}\right)_+ = O\left(\frac{1}{\epsilon}\right)\cdot\left[\sum_{t\in C} y^t - \sum_{t\in P} z^t\right] = O\left(\frac{\log\left(\frac{d\cdot\Delta}{\epsilon}\right)}{\epsilon}\right)\cdot\left[\sum_{t\in C}\bar{y}^t - \sum_{t\in P}\bar{z}^t\right].$$

由于 $(\overline{y}, \overline{z}, \overline{r})$ 是一个可行对偶解，根据弱对偶性，定理得证。 □

3.4 移除 Δ： $O\left(1/\epsilon \log\left(d/\epsilon\right)\right)$ 竞争界

接下来我们证明更精细的界。思路仍然是为 $\overline{\mathcal{D}}$拟合一个对偶解，但这次我们需要更精细地构造对偶解。首先，我们需要克服以下重大障碍：对偶解 y 可以单调地在线计算，但可以证明，任何单调对偶解都无法避免对 Δ的依赖，见[BN09a]。尽管如此，我们证明，可以谨慎地减小对偶解的某些坐标，从而得到以下界。

定理 3.6 (Δ-无关界)。 算法的总重配置成本至多为 $O\left(1/\epsilon \cdot \log(d/\epsilon)\right)$乘以$(\overline{\mathcal{D}})$的目标值。

我们将构造一个向量 $\widetilde{y}$，使其具有以下性质。

引理 3.7。 给定 y，存在一个向量 $\widetilde{y}$，使得

$$\forall 1 \le s \le t \le T,\ \forall i \qquad \sum_{\substack{\tau\in C:\\ \tau\in[s,t]}} c_i^\tau \widetilde{y}^\tau - \sum_{\substack{\tau\in P:\\ \tau\in[s,t]}} p_i^\tau z^\tau \le w_i \cdot \log\left(1+\frac{40d^2}{\epsilon^2}\right) \tag{3.11}$$

$$and \qquad \sum_{\tau\in C}\widetilde{y}^\tau \ge \left(1-\frac{\epsilon}{10}\right)\sum_{\tau\in C} y^\tau. \tag{3.12}$$

有了这个，我们可以如下定义对偶解。令 $A = \log\left(1+\frac{40d^2}{\epsilon^2}\right)$。我们设

一旦我们

,

$$\begin{aligned}
\forall s \in C \qquad & \bar{y}^s = \widetilde{y}^s / A, \\
\forall s \in P \qquad & \bar{z}^s = z^s / A,
\end{aligned}$$

$$\overline{r}_i^{T+1} = 0,$$

$$\forall s \in [T] \qquad \overline{r}_i^s = \frac{1}{A} \cdot \max\left(0,\ \max_{t \geq s}\left\{\sum_{\substack{\tau \in C:\\ \tau \in [s,t]}} c_i^\tau \widetilde{y}^\tau - \sum_{\substack{\tau \in P:\\ \tau \in [s,t]}} p_i^\tau z^\tau\right\}\right).$$

在证明引理 3.7之前，我们先来理解为什么这个对偶解是可行的，并且为何它蕴含定理 3.6。

引理 3.8. 变量 $(\bar{y}, \bar{z}, \bar{r})$ 是$(\overline{\mathcal{D}})$的一个可行对偶解。

证明。 我们先证明该对偶解的可行性。约束 $\overline{r}_i^s \geq 0$ 根据构造成立。对于由正数和负数组成的序列 $a_1, a_2, \ldots, a_T$，有如下结论：

$$\max\left(0,\ \max_{t|\, s \leq t \leq T}\left\{\sum_{\tau \in [s,t]} a_\tau\right\}\right) \geq \max\left(a_s,\ a_s + \max_{t|\, s+1 \leq t \leq T}\left\{\sum_{\tau \in [s+1,t]} a_\tau\right\}\right)$$
$$= a_s + \max\left(0,\ \max_{t|\, s+1 \leq t \leq T}\left\{\sum_{\tau \in [s+1,t]} a_\tau\right\}\right)$$

根据这一观察并代入 $a_t = c_i^\tau \overline{y}^\tau$，若将 $t \in C$或 $a_t = -p_i^\tau \bar{z}^\tau$代入 $t \in P$，便可立即得到 $\overline{r}_i^s \geq c_i^\tau \overline{y}^\tau + \overline{r}_i^{s+1}$；同理可得 $\overline{r}_i^s \geq -p_i^\tau \bar{z}^\tau + \overline{r}_i^{s+1}$。
最后，根据引理 3.11 的不等式3.7 ，我们得到：

$$\overline{r}_i^s = \frac{1}{A} \cdot \max\left(0,\ \max_{t \geq s}\left\{\sum_{\substack{\tau \in C:\\ \tau \in [s,t]}} c_i^\tau \widetilde{y}^\tau - \sum_{\substack{\tau \in P:\\ \tau \in [s,t]}} p_i^\tau z^\tau\right\}\right) \leq \frac{1}{A} \cdot w_i \cdot \log\left(1 + \frac{40d^2}{\epsilon^2}\right) \leq w_i,$$

因此，该对偶解是可行的。 □

将所有结果综合起来，即可得到我们的主定理。
定理3.6的证明。由引理（3.12）的3.7可知$\sum_{\tau \in C} \widetilde{y}^\tau \geq \left(1 - \frac{\epsilon}{10}\right) \sum_{\tau \in C} y^\tau$，因此我们可以重用引理3.5 来界定总补救成本：

$$\sum_{t=1}^{T} \sum_i w_i \left(x_i^t - x_i^{t-1}\right)_+ = O\left(\frac{1}{\epsilon}\right) \cdot \left[\sum_{t \in C} \widetilde{y}^t - \sum_{t \in P} z^t\right] = O\left(\frac{\log\left(\frac{d}{\epsilon}\right)}{\epsilon}\right) \cdot \left[\sum_{t \in C} \bar{y}^t - \sum_{t \in P} \bar{z}^t\right].$$

由于 $(\overline{y}, \overline{z}, \overline{r})$ 是一个可行对偶解，根据弱对偶性，定理得证。 □

本节余下部分将用于证明引理3.7。首先，我们需要一个辅助引理。

引理 3.9。 对于任意满足 $1 \leq s \leq t \leq T$ 的 s 和 t、任意 $i \in [n]$ 以及任意 $S \subseteq C \cap [s,t]$，定义 $c_i^{\max}(S)$ = 最大值$_{\tau \in S}\{c_i^\tau\}$。那么

$$\frac{1}{w_i}\left(\sum_{\substack{\tau \in S:\\ \tau \in [s,t]}} c_i^\tau y^\tau - \sum_{\substack{\tau \in P:\\ \tau \in [s,t]}} p_i^\tau z^\tau\right) \leq \log\left(1 + \frac{4d \cdot c_i^{\max}(S)}{\epsilon} \cdot x_i^t\right).$$

证明。 我们有

$$\frac{1}{w_i}\left(\sum_{\substack{\tau\in S:\\ \tau\in[s,t]}} c_i^\tau y^\tau - \sum_{\substack{\tau\in P:\\ \tau\in[s,t]}} p_i^\tau z^\tau\right) = \sum_{\substack{\tau\in S:\\ \tau\in[s,t], c_i^\tau\neq 0}} \log\left(\frac{x_i^\tau + \frac{\epsilon}{4d^t\cdot c_i^t}}{x_i^{\tau-1} + \frac{\epsilon}{4d^t\cdot c_i^t}}\right) + \sum_{\substack{\tau\in P:\\ \tau\in[s,t]}} \log\left(\frac{x_i^\tau}{x_i^{\tau-1}}\right) \tag{3.13}$$

$$\leq \sum_{\tau=s}^{t} \log\left(\frac{x_i^\tau + \frac{\epsilon}{4d\cdot c_i^{\max}(S)}}{x_i^{\tau-1} + \frac{\epsilon}{4d\cdot c_i^{\max}(S)}}\right) \tag{3.14}$$

$$= \log\left(\frac{x_i^t + \frac{\epsilon}{4d\cdot c_i^{\max}(S)}}{x_i^{s-1} + \frac{\epsilon}{4d\cdot c_i^{\max}(S)}}\right)$$

$$\leq \log\left(1 + \frac{4d\cdot c_i^{\max}(S)}{\epsilon}\cdot x_i^t\right).$$

等式（3.13）由（3.2）和（3.4）得出。要看出不等式（3.14）为何成立，注意对于任意 i 和 $\tau\in C$，都有 $x_i^\tau \geq x_i^{\tau-1}$因此，对于任意 $a\geq b$，有 $\frac{x_i^\tau+a}{x_i^{\tau-1}+a}\leq\frac{x_i^\tau+b}{x_i^{\tau-1}+b}$。对于任意 $\tau\in P$，$x_i^\tau \leq x_i^{\tau-1}$因此，对于任意 $a\geq 0$，可得 $\frac{x_i^\tau}{x_i^{\tau-1}}\leq\frac{x_i^\tau+a}{x_i^{\tau-1}+a}$。最后，对于任意 $\tau\in(C\cap[s,t])\backslash S$，有 $x_i^\tau \geq x_i^{\tau-1}$因此相应的求和项非负。

□

有了这一点，我们终于可以证明该引理了。

引理的证明3.7。 我们通过 $\widetilde{y}$ 迭代地递减由算法生成的解的坐标 y 来得到这一结论。我们对 ℓ 进行归纳，证明不等式 (3.11) 和 (3.12) 一直成立到时刻 ℓ，即

$$\forall 1\leq s\leq t\leq \ell,\ \forall i \qquad \sum_{\substack{\tau\in C:\\ \tau\in[s,t]}} c_i^\tau \widetilde{y}^\tau - \sum_{\substack{\tau\in P:\\ \tau\in[s,t]}} p_i^\tau z^\tau \leq w_i\cdot\log\left(1+\frac{40d^2}{\epsilon^2}\right) \tag{3.15}$$

$$\text{and} \qquad \sum_{\substack{\tau\in C\\ \tau\leq\ell}} \widetilde{y}^\tau \geq \left(1-\frac{\epsilon}{10}\right)\sum_{\substack{\tau\in C\\ \tau\leq\ell}} y^\tau. \tag{3.16}$$

归纳基础是 $\ell=0$，这一点显然成立。现在假设根据归纳假设，对于所有 $\widetilde{y}^\tau$，我们都有一个 $\tau\leq\ell-1$，使得不等式 (3.15) 和 (3.16) 一直成立到时刻 $\ell-1$（并且 $\widetilde{y}^\tau=0$ 对于 $\tau\geq\ell$）。我们说明如何构造一个更新后的分配 $\widetilde{y}^\tau_{new}$，使这些不等式在时刻 ℓ也成立。

如果时间 $\ell\in P$，这些不等式已经成立，因此无需更改 $\widetilde{y}^\tau$。因此，假设 $\ell\in C$。我们按如下方式构造 $\widetilde{y}_{new}$。

算法 1构造 $\widetilde{y}_{new}$

1: 对于 $i = 1, \ldots, n$， 执行 2: 初始化$R_i \leftarrow \left\{ \tau \in [\ell - 1] \;\middle|\; c_i^\tau \geq \frac{10 \cdot d^\ell \cdot c_i^\ell}{\epsilon},\ \widetilde{y}^\tau > 0 \right\}$, // 待减少候选项$B \leftarrow c_i^\ell \cdot y^\ell$. // 预算 3: 当 $B > 0$ 且 $R_i \neq \emptyset$ 时，执行 4: 令 τ^* 为 R_i中的最新时间。5: 更新$B \leftarrow B - c_i^{\tau^*} \Delta\widetilde{y}_i^{\tau^*}$.6: 对每个 $\tau \in [\ell - 1]$ 执行 7: 更新$\Delta\widetilde{y}_i^\tau \leftarrow \min\{\widetilde{y}^\tau, B/c_i^{\tau *}\}$.8: 如果 $\Delta\widetilde{y}_i^\tau = \widetilde{y}^\tau$，则同时更新 $R_i \leftarrow R_i \setminus \{\tau^*\}$。 9: 最后，设置$\widetilde{y}_{new}^\ell \leftarrow y^\ell$，并且对于所有 $t \in [\ell - 1]$ $\widetilde{y}_{new}^\tau \leftarrow \widetilde{y}^\tau - \max_{i|\, c_i^t \neq 0} \{\Delta\widetilde{y}_i^\tau\}$.

根据构造，在总会终止的 while 循环结束时，要么 $R_i = \emptyset$，要么已花费的总预算为 $\Delta B = c_i^\ell y_i^\ell = \sum_{\tau \leq \ell - 1} c_i^\tau \Delta\widetilde{y}_i^\tau$。在整个执行过程中，我们始终保持

$$c_i^\ell y_i^\ell - \sum_{\tau \leq \ell - 1} c_i^\tau \Delta\widetilde{y}^\tau \geq 0$$

并且对于所有 $\tau \in [\ell - 1]$，都有 $\Delta\widetilde{y}_i^\tau \leq \widetilde{y}^\tau$。

Sinc**e** $c_i^\ell y_i^\ell - \sum_\tau c_i^\tau \Delta\widetilde{y}_i^\tau \geq 0$对于每个 $\tau \in R_i$，根据定义 $c_i^\tau \geq 10 \cdot d^\ell c_i^\ell / \epsilon$，我们得到 th at

$$\sum_{\tau \leq \ell - 1} \Delta\widetilde{y}_i^\tau \leq \frac{\epsilon}{10 \cdot d^t} \cdot \sum_{\tau \leq \ell - 1} \frac{c_i^\tau \cdot \Delta\widetilde{y}_i^\tau}{c_i^t} \leq \frac{\epsilon}{10 \cdot d^t} \cdot y^\ell. \tag{3.17}$$

因此，即使在算法 1的减量操作之后，我们添加到 $\widetilde{y}$ 中的总质量至少为

$$\begin{aligned}
(\widetilde{y}_{new}^\ell - \widetilde{y}^\ell) + \sum_{\tau \leq t - 1} (\widetilde{y}_{new}^\tau - \widetilde{y}^\tau) &= y^\ell - \sum_{\tau \leq \ell - 1} \max_{i|\, c_i^t \neq 0} \{\Delta\widetilde{y}_i^\tau\} \\
&\geq y^\ell - \sum_{i|\, c_i^t \neq 0} \sum_{\tau \leq \ell - 1} \Delta\widetilde{y}_i^\tau \\
&\geq y^\ell - \frac{\epsilon}{10 \cdot d^\ell} \sum_{i|\, c_i^\ell \neq 0} y^\ell \\
&= \left(1 - \frac{\epsilon}{10}\right) y^\ell.
\end{aligned}$$

上文中，第一个等号是因为我们将 $\widetilde{y}^\ell$ 定义为 0。第二个不等式来自（3.17），第三个则是因为 c^ℓ 至多有 d^ℓ 个坐标非零。

结合归纳假设（3.16），

$$\sum_{\tau\le\ell}\widetilde{y}^{\tau}_{new}=(\widetilde{y}^{\ell}_{new}-\widetilde{y}^{\ell})+\sum_{\tau\le\ell-1}(\widetilde{y}^{\tau}_{new}-\widetilde{y}^{\tau})+\sum_{\tau\le\ell-1}\widetilde{y}^{\tau}\ge\left(1-\frac{\epsilon}{10}\right)\sum_{\substack{\tau\in C\\ \tau\le\ell}}y^{\ell},$$

因此，不等式（3.16）得以保持。

接下来证明，对于任意 $i\in[n]$ 和 $s\le t=\ell$，不等式（3.15）成立。固定一个指标 i，并令 $\tau_i^{\min}$为满足 τ 的最早时刻 $\Delta\widetilde{y}_i^{\tau}>0$。我们分情况讨论。

情况 1： $\mathbf{c_i^{\ell}=0}$。在这种情况下

$$\sum_{\substack{\tau\in C:\\ \tau\in[s,\ell]}}c_i^{\tau}\widetilde{y}^{\tau}_{new}-\sum_{\substack{\tau\in P:\\ \tau\in[s,\ell]}}p_i^{\tau}z^{\tau}=\sum_{\substack{\tau\in C:\\ \tau\in[s,\ell-1]}}c_i^{\tau}\widetilde{y}^{\tau}_{new}-\sum_{\substack{\tau\in P:\\ \tau\in[s,\ell-1]}}p_i^{\tau}z^{\tau}\le\sum_{\substack{\tau\in C:\\ \tau\in[s,\ell-1]}}c_i^{\tau}\widetilde{y}^{\tau}-\sum_{\substack{\tau\in P:\\ \tau\in[s,\ell-1]}}p_i^{\tau}z^{\tau}.$$

根据归纳假设，不等式（3.15）成立。在其余情况下，假设$c_i^{\ell}>0$。

情况 2： 在循环结束时 $\mathbf{R_i\neq\emptyset}$，**且** $\mathbf{s\le\tau_i^{\min}}$。在这种情况下，预算 B 已全部用尽，即

$$c_i^{\ell}y^{\ell}=\sum_{\substack{\tau\in C\\ \tau\in[\tau_i^{\min},\ell-1]}}c_i^{\tau}\Delta\widetilde{y}_i^{\tau},$$

这意味着

$$\sum_{\substack{\tau\in C:\\ \tau\in[\tau_i^{\min},\ell]}}c_i^{\tau}\widetilde{y}^{\tau}_{new}-\sum_{\substack{\tau\in P:\\ \tau\in[\tau_i^{\min},\ell]}}p_i^{\tau}z^{\tau}\le\sum_{\substack{\tau\in C:\\ \tau\in[\tau_i^{\min},\ell-1]}}c_i^{\tau}\widetilde{y}^{\tau}-\sum_{\substack{\tau\in P:\\ \tau\in[\tau_i^{\min},\ell-1]}}p_i^{\tau}z^{\tau}.$$

那么，对于任意 $s\le\tau_i^{\min}$，

$$\sum_{\substack{\tau\in C:\\ \tau\in[s,\ell]}}c_i^{\tau}\widetilde{y}^{\tau}_{new}-\sum_{\substack{\tau\in P:\\ \tau\in[s,\ell]}}p_i^{\tau}z^{\tau}\le\sum_{\substack{\tau\in C:\\ \tau\in[s,\ell-1]}}c_i^{\tau}\widetilde{y}^{\tau}-\sum_{\substack{\tau\in P:\\ \tau\in[s,\ell-1]}}p_i^{\tau}z^{\tau},$$

并且根据归纳假设，不等式（3.15）再次成立。

情况 3： $\mathbf{R_i=\emptyset}$ **在循环结束时，或** $\mathbf{s>\tau_i^{\min}}$。在此情况下 $[s,\ell]\cap R_i=\emptyset$，因此，对于每个 $\tau\in[s,\ell]$，要么 $\widetilde{y}^{\tau}_{new}=0$，要么 $c_i^{\tau}\le10\cdot d^{\ell}c_i^{\ell}/\epsilon$。定义

$$S=C\cap\{\tau\in[s,\ell],\widetilde{y}^{\tau}>0\},$$
$$c_i^{\max}(S)=\max_{\tau\in S}\{c_i^{\tau}\}\le10\cdot d^{\ell}c_i^{\ell}/\epsilon.$$

应用引理 3.9 及以上集合 S，可得：

$$\begin{aligned}\frac{1}{w_i}\left(\sum_{\substack{\tau\in C\\ \tau\in[s,t]}}c_i^{\tau}\widetilde{y}^{\tau}-\sum_{\substack{\tau\in P:\\ \tau\in[s,t]}}p_i^{\tau}z^{\tau}\right)&=\frac{1}{w_i}\left(\sum_{\tau\in S}c_i^{\tau}y^{\tau}-\sum_{\substack{\tau\in P:\\ \tau\in[s,t]}}p_i^{\tau}z^{\tau}\right)\\&\le\log\left(1+\frac{4d\cdot c_i^{\max}(S)}{\epsilon}\cdot x_i^{t}\right)\\&\le\log\left(1+\frac{4d}{\epsilon}\cdot\frac{10\cdot d^{\ell}c_i^{\ell}}{\epsilon}\cdot\frac{1}{c_i^{\ell}}\right)\\&\le\log\left(1+\frac{40d^2}{\epsilon^2}\right).\end{aligned}$$

第二个不等式由 $c_i^{\max}(S)$ 的定义得出。最后一个不等式源于这样一个事实：$x_i^{\ell}\le1/c_i^{\ell}$；否则，时间 ℓ 的约束在时间 $\ell-1$ 就已经满足了。由此证明完毕。

□

4 舍入

本节中，我们研究追逐正体问题的若干应用。对于这些问题中的每一个，我们都证明，其恰当的分数版本符合我们的框架。随后，我们说明如何将分数解舍入为整数解。

4.1 动态集合覆盖

在集合覆盖问题中，给定一个由 n 个元素 U 和一个集合族 m 集合 S_1、.....、$S_m \subseteq U$ 组成的全集，其中每个集合 S_i 的成本为 $c(S_i) \geq 0$。目标是找到一个成本最低的集合族，使其覆盖所有元素。

在动态场景中，在每个时间步 $t \in [T]$，一个元素会被插入到全集或从全集中删除 U。令 U^t 表示 U在时间 t时的状态[4]，并令 OPT^t表示最优 分数 集合覆盖在 U^t下的成本。我们希望维护一个 $\mathcal{S}^t$对 U^t的集合覆盖，其成本$c(\mathcal{S}^t)$: $= \sum_{S\in\mathcal{S}^t} c(S)$ 始终不超过 $\beta \geq 1$ 倍的 OPT^t，同时使总补救成本 $\sum_{t\in[T]} |\mathcal{S}^t \oplus \mathcal{S}^{t-1}|$尽可能小。动态集合覆盖问题的分数版本自然适用于我们的框架

$$K_t := \left\{ x^t \in \mathbb{R}_+^m \;\middle|\; \begin{array}{ll} \sum_{\substack{i\in[m]:\\ u\in S_i}} x_i^t \geq 1 & \forall u \in U^t, \\ \\ \sum_{i\in[m]} c(S_i)\cdot x_i^t \leq \beta \cdot \text{OPT}^t & \end{array} \right\}.$$

定理 4.1 (集合覆盖舍入)。 对于动态集合覆盖的任意分数解 x^t ， 我们有 :

(1) 存在一个确定性动态舍入方案，使得

$$\sum_{t\in[T]} |\mathcal{S}^t \oplus \mathcal{S}^{t-1}| \leq O(f)\cdot \sum_{t\in[T]} \|x^t - x^{t-1}\|_1,$$
$$\sum_{S\in\mathcal{S}^t} c(S) \leq O(f)\cdot \sum_{i\in[m]} c(S_i)\cdot x_i^t \qquad \forall t \in [T].$$

这里， f 是任意时刻任一元素的最大频率的上界，即 .,

$$f \geq \max_{u\in U^t} |\{i \in [m] : u \in S_i\}| \quad \forall t \in [T].$$

(2) 对于每个 $\alpha \geq 1$，存在一个随机动态舍入方案，使得

$$\mathbb{E}\left[\sum_{t\in[T]} \left|\mathcal{S}^t \oplus \mathcal{S}^{t-1}\right|\right] \leq (\alpha \log n)\cdot \sum_{t\in[T]} \|x^t - x^{t-1}\|_1 + O\left(\frac{T}{n^\alpha}\right), \tag{4.1}$$

$$\mathbb{E}\left[\sum_{S\in\mathcal{S}^t} c(S)\right] \leq O(\alpha \log n)\cdot \sum_{i\in[m]} c(S_i)\cdot x_i^t \qquad \forall t \in [T]. \tag{4.2}$$

此处， $n \geq \max_t |U^t|$ 表示任意时刻全集中元素数量的上界。

[4]我们假设 U^t 中的每个元素都至少被 $\{S_1, \ldots, S_m\}$中的一个集合覆盖。这确保了有效集合覆盖的存在。

证明推迟至附录 B，所采用的方法是将标准的离线舍入技术应用于动态场景。注意，在该问题中，K_t 的覆盖约束的行稀疏度至多为 $f \geq \max_{u \in U^t} |\{i \in [m] : u \in S_i\}|$。因此，通过在定理 1.1 中设置 $\epsilon = 1$，我们可以维护一个分数解 x^t，使其在任意时刻 t 均满足覆盖约束、成本至多为 $O(\beta) \cdot \text{Opt}^t$，并且相对于任何维护成本为 $\leq \beta \cdot \text{Opt}^t$的分数解的离线算法，具有 $O(\log f)$-竞争的总补救成本。另请注意，如果将 α 设为常数，则（4.1）中的加性项将变为每个时间步 $1/\text{poly}(n)$。将其与定理 4.1 结合可得：

推论 4.2. 对于任意 $\beta \geq 1$ 以及任意由 T 次更新组成的更新序列 σ，存在：
(1) 一种确定性动态算法，可维护集合覆盖 $\mathcal{S}^t$，其成本为 $O(\beta \cdot f \cdot \text{Opt}^t)$，并始终成立；其总重配置成本至多为 $O(f \log f) \cdot \text{Opt}^{\beta}{}_{\text{REC}}(\sigma)$。*(2)* 一种随机化动态算法，可维护集合覆盖 $\mathcal{S}^t$，其期望成本为 $O(\beta \cdot \log n \cdot \text{Opt}^t)$，并始终成立；其期望总重配置成本至多为 $O(\log f \cdot \log n) \cdot \text{Opt}^{\beta}{}_{\text{REC}}(\sigma) + \frac{T}{\text{poly}(n)}$。

这里，$\text{OPT}^{\beta}{}_{\text{REC}}(\sigma)$ 是指对于任意离线算法，在整个更新序列 σ 中维护成本不超过 $\beta \cdot \text{OPT}^t$ 的分数集合覆盖时，其最优总重配置成本。

我们注意到，可以很容易地将分析扩展到允许其他形式的更新，例如更改某个集合的成本。

4.2 动态二分匹配

在最大二分图匹配问题中，给定一个含有$|V| = n$ 个节点的二分图 $G = (V, E)$。匹配 $M \subseteq E$ 是边集中任意两条边都不共享端点的子集，目标是找到一个规模最大的匹配 M。在动态场景中，每个时间步$t \in [T]$ 都会向集合 E中插入一条边或将其删除。令 E^t 表示时间t时 E 的状态，令 Opt^t 表示时间 t时最大分数匹配的值。我们希望在 $G^t = (V, E^t)$中维护一个匹配 M^t，使其规模始终在 Opt^t的常数因子 $\beta \leq 1$ 范围内，同时尽可能降低总补救成本 $\sum_{t \in [T]} |M^t \oplus M^{t-1}|$。动态最大匹配问题的分数版本自然地适用于我们的框架，具体如下

$$K_t := \left\{ x^t \in \mathbb{R}_+^{\binom{|V|}{2}} \;\middle|\; \begin{array}{cc} \sum_{e \in \partial(v)} x_e^t \leq 1 & \forall v \in V, \\ x_e^t = 0 & \forall e \in \binom{V}{2} \setminus E^t, \\ \sum_{e \in E^t} x_e^t \geq \beta \cdot \text{OPT}^t. & \end{array} \right\}. \tag{4.3}$$

注意，向量 x^t 的每个潜在边 （即节点的无序对）$e \in \binom{V}{2}$都对应一个条目。如果潜在边 e 当前不存在于图中，则令 $x_e^t = 0$。符号 $\partial(v)$表示所有至少有一个端点属于 $v \in V$的潜在边的集合。

本节的主要结果总结如下。定理4.3的证明将在附录C中给出，它基于将一种标准的 [ACC+18] 舍入技术应用于我们的设定。

定理 4.3 (最大匹配舍入)。对于任意动态二分图匹配的分数解 x^t 以及任意 $\delta \in (0,1)$，$\alpha \geq 1$，都存在一种随机动态舍入方案，使得：

$$\mathbb{E}\left[\sum_{t \in [T]} \left|M^t \oplus M^{t-1}\right|\right] \leq O\left(\frac{\alpha \log n}{\delta^3}\right) \cdot \sum_{t \in [T]} \|x^t - x^{t-1}\|_1 + O\left(\frac{T}{\delta \cdot n^{\alpha}}\right). \tag{4.4}$$

$$\mathbb{E}\left[\left|M^t\right|\right] \geq (1-\delta)\cdot \sum_{e\in E^t} x_e^t \qquad \forall t\in[T]. \tag{4.5}$$

注意，在此问题中，覆盖约束的行稀疏度

K_t $\binom{n}{2}$ = 至多为 ()。

Θ n^2 1 1,

因此，使用定理 .中的在线算法，我们可以维持一个分数匹配 x^t，使其始终满足 $\sum_{e\in E^t} x_e^t \geq (1-\epsilon)\beta\cdot \textsc{Opt}^t$，并且 t相对于任意维持值为 $O((1/\epsilon)\cdot\log(n/\epsilon))$的分数匹配的离线算法，其总重配置成本具有 $\geq \beta\cdot \textsc{Opt}^t$。令 $\delta=\epsilon$ 和 α 为某个足够大的常数，并将其与定理 4.3结合，便可得到：

推论 *4.4.* 对于任意 $\beta,\epsilon\in(0,1]$, 以及任意包含 T 次更新的更新序列 σ，存在一种随机化动态算法，能够始终维护一个匹配 M^t，其期望规模至少为 $(1-\epsilon)\beta\cdot Opt^t$，且期望总重配置成本为 $O((1/\epsilon^4)\cdot \log n\cdot \log(n/\epsilon))\cdot Opt^{\beta}\, Rec(\sigma) + \epsilon\cdot \overset{T}{} poly(n)$。这里，$Opt^{\beta}\, Rec(\sigma)$ 表示任意离线算法在整个更新序列 ~~σ中~~始终维护价值至少为 $\beta\cdot Opt^t$ 的分数匹配时所需的最优总重配置成本。

我们指出，将定理 4.3 和推论 4.4 推广到带权二分图是直接的，只需采用 Bernstein 等人提出的经典归约（参见 [BDL21]中的定理 3.1），但重配置成本会产生关于 $(1/\epsilon)$ 的指数级开销因子。我们也可以轻松地将分析推广到允许顶点更新的情形。

4.3 无关机器上的动态负载均衡

在负载均衡问题中，给定一组 M 个 m 机器，以及一组 J 个 作业。每个作业 $j\in J$ 只能由机器的一个非空子集 $M(j)\subseteq M$ 处理。若将作业j 分配给机器 $i\in M(j)$，则该机器的负载 增加 p_{ij}。我们的目标是找到一种有效的分配 $\psi: J\to M$，将作业分配给机器，从而最小化“最大完工时间”。最大完工时间定义为任意机器上的最大负载，其表达式为 $\textsf{Obj}(\psi) := \max_{i\in M}\left(\sum_{j\in J:\psi(j)=i} p_{ij}\right)$。

在动态场景中，每个时间步 t 都会有一个作业插入到集合 J中或从中删除。（我们的分析也适用于其他类型的更新，例如更改作业 j的集合 $M(j)$。）令J^t 表示时间 t的作业集合。令 $\textsf{Opt}^t_{int}$ 表示时间t的最优整数 最大完工时间。我们希望维护一个有效分配 $\psi^t: J^t\to M$，使得 $\textsf{Obj}(\psi^t)\leq \beta\cdot \textsf{Opt}^t_{int}$，其中 $\beta\geq 1$为某个常数。每当一个作业从一台机器重新分配到另一台机器，或有作业插入集合 J或从中删除时，我们都会产生一次“重配置”。沿用记号$\oplus$表示两个集合的对称差，我们将时间 t 产生的重配置成本定义为 $\psi^{t-1}\oplus\psi^t$:
$= |J^t\oplus J^{t-1}| + |\{j\in J^t\cap J^{t-1}: \psi^t(j)\neq\psi^{t-1}(j)\}|$。我们的目标是使总重配置成本（记为 $\sum_{t\in[T]}\psi^t\oplus\psi^{t-1}$）尽可能小。令 $\mathcal{J}$表示所有可能出现的作业的集合，因此对于所有 t，均有 $J^t\subseteq\mathcal{J}$；并令 $n := |\mathcal{J}|$。该动态问题的分数版本符合我们的框架，其中集合 K_t 由满足以下约束的所有 $x^t\in\mathbb{R}_+^{|\mathcal{J}|\times|M|}$ 构成:

$$\sum_{i\in M} x_{ij}^t \geq 1 \qquad \forall j\in J^t. \tag{4.6}$$

$$x_{ij}^t = 0 \qquad \forall i\in M, j\in J^t \text{ with } p_{ij} > \textsc{Opt}^t_{int}. \tag{4.7}$$

$$x_{ij}^t = 0 \qquad \forall i\in M, j\in\mathcal{J}\setminus J^t. \tag{4.8}$$

$$x_{ij}^t = 0 \qquad \forall j\in\mathcal{J}, i\in M\setminus M(j). \tag{4.9}$$

$$\sum_{j\in J^t} p_{ij}\cdot x_{ij}^t \leq \beta\cdot \textsc{Opt}^t_{int} \qquad \forall i\in M. \tag{4.10}$$

本节的主要结果总结如下。定理4.5 的证明几乎

可直接由 Krishnaswamy 等人的一项近期结果推出。 [KLS23]下述定理的第 1 和第 2 部分分别由附录 B 和 [KLS23],的第 4 节推出。[5]

定理 4.5 (由 [KLS23] 提出的负载均衡舍入）。考虑无关机器上的动态负载均衡问题的任意分数解 x^t 。那么有：

(1) 对任意 $\delta > 0$，存在一种确定性动态舍入方案，使得：

$$\sum_{t\in[T]} \psi^t \oplus \psi^{t-1} \leq O\left(\frac{1}{\delta^4}\log\frac{1}{\delta}\cdot\log n\right)\cdot\sum_{t\in[T]}\left\|x^t - x^{t-1}\right\|_1, \tag{4.11}$$

$$\texttt{Obj}(\psi^t) \leq (2+\delta)\beta\cdot OPT^t_{int} \qquad \forall t\in[T]. \tag{4.12}$$

(2) 存在一种随机动态舍入方案，使得：

$$\mathbb{E}\left[\sum_{t\in[T]} \psi^t \oplus \psi^{t-1}\right] \leq O(1)\cdot\sum_{t\in[T]}\left\|x^t - x^{t-1}\right\|_1 + \frac{T}{\text{poly}(n)}, \tag{4.13}$$

$$\mathbb{E}\left[\texttt{Obj}(\psi^t)\right] = O\left(\frac{\log\log n}{\log\log\log n}\right)\cdot OPT^t_{int} \qquad \forall t\in[T]. \tag{4.14}$$

在 K_t 中，覆盖约束的行稀疏度为 $r := \max_{j\in\mathcal{J}}|M(j)|$。因此，根据定理 1.1，我们可以始终维护一个分数解 x^t ，使其满足四个约束 (4.6)-(4.9)，以及约束 $\sum_{j\in J^t} p_{ij}\cdot x^t_{ij} \leq (1+\epsilon)\beta\cdot \text{Opt}^t_{int}$ t。此外，对于任意维护满足全部五个约束 (4.6)-(4.10) 的可行分数解的离线算法，该解的总补救成本为 $O((1/\epsilon)\log(r/\epsilon))$ 倍竞争比。令 $\delta = \epsilon$ ，并将其与定理 4.5 结合，可得：

推论 4.6. 对于任意 $\beta \geq 1$, $\epsilon \in (0,1]$, 以及任意包含 T 次更新的更新序列 σ ，存在：

(1) 一种确定性动态算法，始终维护一个最大完工时间至多为 $(2+\epsilon)\beta\cdot Opt^t_{int}$ 的分配 ψ^t ，总补救成本为 $O((1/\epsilon^5)\log(1/\epsilon)\log n\log(r/\epsilon))\cdot Opt^\beta Rec(\sigma)$。*(2)* 一种随机化动态算法，对于所有 t，维护的分配 ψ^t 的期望最大完工时间为 $O(\log\log n\log\log\log n)\cdot Opt^t_{int}$，期望总补救成本~~为 $O(\log(r))\cdot Opt^\beta Rec(\sigma) +$~~ $T\,poly(n)$ 。

这里，OPT^β $_{REC}(\sigma)$ 是任意离线算法在整个更新序列 σ期间，维护满足所有五个约束 的可行分数解的最优总补救成本 (4.6)-(4.10)$r := \max_{j\in\mathcal{J}}|M(j)|$ 是愿意处理任意给定作业的机器数量上限。

4.4 动态最小生成树

在最小生成树（MST）问题中，给定一个包含 $|V| = n$ 个节点的连通图 $G = (V, E)$，每条边都关联一个成本 $c_e \geq 0$ 。我们的目标是计算图 G 的一棵生成树 $\mathcal{T} = (V, E_\mathcal{T})$，使其总成本 $\sum_{e\in E_\mathcal{T}} c_e$ 最小。

在动态场景中，每个时间步 $t \in [T]$ 都会向集合中插入一条边或从集合中删除一条边E，同时要求输入图 $G = (V, E)$ 保持连通。令 E^t 表示 E 在时间 t时的状态，并令 Opt^t 表示$G^t = (V, E^t)$的最小成本生成树的值。我们希望在 G^t 中维护一棵生成树 $\mathcal{T}^t = (V, E^t_\mathcal{T})$，使其成本始终

[5]关于负载均衡的最终结果[KLS23] 由于在线分数算法的局限性，仅处理作业到达。不过，他们的舍入方案适用于完全动态环境。

在 Opt^t 的 $\beta \geq 1$ 倍以内，同时使总补救成本 $\sum_{t\in[T]} \left|E_{\mathcal{T}}^t \oplus E_{\mathcal{T}}^{t-1}\right|$ 尽可能小。动态最小生成树问题的分数版本自然地契合我们的框架，具体如下

$$K_t := \left\{ x^t \in \mathbb{R}_+^{\binom{|V|}{2}} \;\middle|\; \begin{array}{cc} \sum_{e\in\partial^t(S)} x_e^t \geq 1 & \forall\, \emptyset \subset S \subset V, \\ x_e^t = 0 & \forall e \in \binom{V}{2} \setminus E^t, \\ \sum_{e\in E^t} c_e \cdot x_e^t \leq \beta \cdot \mathrm{OPT}^t. & \end{array} \right\}. \tag{4.15}$$

注意，向量 x^t 的每个条目都对应一个潜在边 $e \in \binom{V}{2}$。如果潜在边 e 当前不在图中，则设定 $x_e^t = 0$。记号 $\partial^t(S)$ 表示 G^t 中所有跨越割 S 的边的集合。我们称 $x^t \in \mathbb{R}^{\binom{V}{2}}_+$ 是 G^t 中的分数最小生成树，其成本至多为 $\beta \cdot \mathrm{Opt}^t$，当且仅当 $x^t \in K_t$ 如上面的（4.15）式所示。本节的主要结果总结如下方定理。定理 4.7 的证明推迟至附录 D。

定理 4.7 (最小生成树舍入)。 考虑动态最小生成树问题的任意分数解 x^t，以及任意 $\delta > 0$, $\alpha \geq 1$。则存在一种随机动态舍入方案，维护 G^t 的一棵生成树 $\mathcal{T}^t = (V, E_{\mathcal{T}}^t)$，使得:

$$\mathbb{E}\left[\sum_{t\in[T]} \left|E_{\mathcal{T}}^t \oplus E_{\mathcal{T}}^{t-1}\right|\right] \leq O\left(\frac{\alpha \log n}{\delta^2}\right) \cdot \sum_{t\in[T]} \left\|x^t - x^{t-1}\right\|_1 + \frac{T}{O(n^\alpha)}, \tag{4.16}$$

$$\sum_{e\in E_{\mathcal{T}}^t} c_e \leq (2+\delta) \cdot \sum_{e\in E^t} c_e \cdot x_e^t \qquad \forall t \in [T]. \tag{4.17}$$

在这个问题中，覆盖约束的行稀疏度至多为 $\binom{n}{2} = \Theta(n^2)$。因此，根据定理 1.1，我们可以维护一个分数最小生成树 x^t，其 G^t 成本为 $\sum_{e\in E^t} c_e \cdot x_e^t \leq (1+\epsilon)\,\beta \cdot \mathrm{OPT}^t$，并且相对于任何始终维护成本至多为 $\beta \cdot \mathrm{OPT}^t$ 的离线算法，具有 $O\left((1/\epsilon)\log(n/\epsilon)\right)$-竞争性的总重配置。令 $\delta = \epsilon$和 α 为某个足够大的常数，并将其与定理 4.7结合，可得:

推论 4.8。 对于任意 $\beta \geq 1$、 $\epsilon \in (0, 1]$, 以及任意包含 T 次更新的更新序列 σ，存在一种随机化动态算法，始终维护一棵成本至多为 $(2+\epsilon)\,\beta \cdot OPT^t$ 的生成树，其期望总重配置为 $O\left((1/\epsilon^3) \cdot \log n \cdot \log(n/\epsilon)\right) \cdot OPT^{\beta}{}_{REC}(\sigma) + T\,\mathrm{poly}(n)$。其中，
$OPT^{\beta}{}_{REC}(\sigma)$ 是任何在整个更新序列 σ 中始终维护成本不超过 $\beta \cdot OPT^t$ 的分数最小生成树的离线算法所需的最优总补救成本。
我们指出，可以轻松地将分析扩展到其他形式的更新，例如更改正在插入的边的成本，或进行顶点插入/删除。

将正体规约为正半空间

本节将说明，只需假设在每个时刻 t，每个正体 K_t 都由单个覆盖约束或装箱约束定义。

假设我们有一个算法 $\mathcal{A}$，用于追逐半空间，该算法完全满足覆盖约束，并将装箱约束满足到 ϵ: 该算法生成 x^t，使得对所有 $t \in C$，都有 $\langle c^t, x\rangle \geq 1$；对所有 $t \in P$，都有 $\langle p^t, x\rangle \leq 1+\epsilon$。

以下是算法 $\mathcal{A}'$，它在违反程度不超过 δ 的情况下满足装箱约束。给定一个体 $K_t \neq \emptyset$，只要存在一个被违反至少 $\delta/10$ 的约束 K_t，就将该约束交给以参数 $\epsilon = \delta/20$ 运行的算法 $\mathcal{A}$。每次进行这样的修正时，算法 $\mathcal{A}$ 至少移动 $\Omega(\delta/w)$，其中 w 是该约束中最大的非零系数。由于存在

一个满足所有约束的有限移动最优解，而且追逐半空间的算法 $\mathcal{A}$ 的竞争比也是有限的，因此这种迭代的次数有限。

因此，经过有限轮后，K_t 中的所有约束都满足至多相差 $\delta/10$。通过将解按 $(1-\delta/10)^{-1}$（在所有时刻 t）放大，我们可以确保所有覆盖约束都完全满足，而装箱约束最多违反 δ。此外，如果 $\mathcal{A}$ 对于正半空间追逐是 c-竞争的，那么 $\mathcal{A}$ 输出的解至多为追逐这些正体的最优算法所得解的 c K_t。因此，算法 $\mathcal{A}'$ 对于正体追逐也是 $O(c)$-竞争的。

B 定理证明 4.1

定理 4.1 (集合覆盖舍入). 对于动态集合覆盖的任意分数解 x^t，我们有：

(1) 存在一种确定性动态舍入方案，使得

$$\sum_{t\in[T]} |\mathcal{S}^t \oplus \mathcal{S}^{t-1}| \le O(f)\cdot \sum_{t\in[T]} \|x^t - x^{t-1}\|_1,$$
$$\sum_{S\in\mathcal{S}^t} c(S) \le O(f)\cdot \sum_{i\in[m]} c(S_i)\cdot x_i^t \qquad \forall t\in[T].$$

这里，f 是任意元素在任意时刻最大频率的上界，即

$$f \ge \max_{u\in U^t} |\{i\in[m] : u\in S_i\}| \quad \forall t\in[T].$$

(2) 对于每个 $\alpha \ge 1$，存在一种随机动态舍入方案，使得

$$\mathbb{E}\left[\sum_{t\in[T]} \left|\mathcal{S}^t \oplus \mathcal{S}^{t-1}\right|\right] \le (\alpha\log n)\cdot \sum_{t\in[T]} \|x^t - x^{t-1}\|_1 + O\left(\frac{T}{n^\alpha}\right), \tag{4.1}$$
$$\mathbb{E}\left[\sum_{S\in\mathcal{S}^t} c(S)\right] \le O(\alpha\log n)\cdot \sum_{i\in[m]} c(S_i)\cdot x_i^t \qquad \forall t\in[T]. \tag{4.2}$$

这里，$n \ge \max_t |U^t|$ 是在任意时刻全集中元素数量的上界。

证明。 我们分别证明每一部分。

（1）的证明： 初始化 $\mathcal{S}^0 = \emptyset$。随后，在处理时间 t 的更新时，初始化$\mathcal{S}^t = \mathcal{S}^{t-1}$。接下来，每当对于某个 $S_i \notin \mathcal{S}^t$ 有 $x_i^t \ge 1/f$ 时，就将 S_i 添加到 $\mathcal{S}^t$ 中。相反，每当对于某个 $S_i \in \mathcal{S}^t$ 有 LP 值 $x_i^t \le 1/2f$ 时，就从 $\mathcal{S}^t$ 中移除 S_i。
因此，我们始终有 $\{S_i : x_i \ge 1/2f\} \supseteq \mathcal{S} \supseteq \{S_i : x_i \ge 1/f\}$。由于 x 是一个有效的分数集合覆盖，这确保了 $\mathcal{S}$ 是一个成本至多为 $2f\cdot\sum_{i=1}^m c(S_i)\cdot x_i^t$ 的有效整数集合覆盖。
为了界定重配置，请注意，我们只有在集合 S_i 的 LP 值发生至少 $1/2f$ 的变化后，才会在 $\mathcal{S}$ 中插入/删除该集合（从而产生一次重配置）。换句话说，我们的取整方案产生的重配置至多是维护分数解 x 的底层动态算法所产生的重配置的 $2f$ 倍。

（2）的证明： 我们维护两个集合族 $\mathcal{R}^t$ 和 $\mathcal{B}^t$，并将其输出作为我们的解$\mathcal{S}^t := \mathcal{R}^t \cup \mathcal{B}^t$。
（读者可以将 $\mathcal{R}$ 视为随机采样的集合，而将 $\mathcal{B}$ 视为

备用集合。）对于每个 $i \in [m]$，集合 S_i 都有一个参数为$\lambda := \alpha\log\ n$ 的指数随机变量 χ_i，该变量仅采样一次，并在动态过程开始时固定。在整个更新序列中，对于每个时间步，我们定义 $\mathcal{R}^t := \{S_i \in \mathcal{S} : x_i^t \geq \chi_i\}$。对于每个元素 $u \in U$，令 $S(u)$ 表示 $\{S_1, \ldots, S_m\}$ 中包含 u 的成本最低的集合。定义$\mathcal{B}^t = \{S(u)$为未被 $\mathcal{R}^t\}$覆盖的。

显然， $\mathcal{R}^t \cup \mathcal{B}^t$ 是一个可行的集合覆盖；接下来需要分析近似比和重配置。我们分别对 $\mathcal{R}^t$ 和 $\mathcal{B}^t$进行分析。

为了分析 $\mathcal{R}^t$，注意到对于所有 $i \in [m]$，我们有

$$\Pr[S_i \in \mathcal{R}^t] = \Pr[x_i^t \geq \chi_i] = 1 - \exp\left(-x_i^t \cdot \alpha \log n\right) \leq x_i^t \cdot \alpha \log n.$$

对 $i \in [m]$求和，可将 $\mathcal{R}^t$的期望成本上界为 $\mathbb{E}[c(\mathcal{R}^t)] \leq \sum_{i\in[m]} c(S_i) \cdot x_i^t \cdot \alpha\log\ n$。

对于 $\mathcal{R}$的重配置，注意到由于 t^{th} 更新，集合 S_i 产生一次重配置，当且仅当 χ_i位于长度为 $|x_i^t - x_i^{t-1}|$的区间内。由于指数分布具有无记忆性，该事件发生的概率至多为 $1 - \exp\left(-\left|x_i^t - x_i^{t-1}\right| \cdot \alpha\log\ n\right) \leq \left|x_i^t - x_i^{t-1}\right| \cdot \alpha\log\ n$。对所有 $i \in [m]$求和，由 t^{th}更新导致的 $\mathcal{R}$期望重配置为：

$$\mathbb{E}\left[\left|\mathcal{R}^t \oplus \mathcal{R}^{t-1}\right|\right] \leq \alpha \log n \cdot \left\|x^t - x^{t-1}\right\|_1.$$

接下来考虑 $\mathcal{B}^t$ 中的任意元素 $u \in U^t$。我们有：

$$\begin{aligned}\Pr\left[u \text{ not covered by } \mathcal{R}^t\right] &= \prod_{i\in[m]:\, u\in S_i} \Pr[S_i \notin \mathcal{R}^t] = \prod_{i\in[m]:\, u\in S_i} e^{-x_i^t \cdot \alpha \log n} \\ &= \exp\left\{-\alpha \log n \cdot \left(\sum_{i\in[m]:\, u\in S_i} x_i^t\right)\right\} \leq \frac{1}{n^\alpha},\end{aligned}$$

其中，最后一个不等式成立是因为 x_i^t 是一个可行的分数集合覆盖。由于 $|U^t| \leq n$，可得 U^t 中未被覆盖（相对于 $\mathcal{R}^t$）的元素的期望数量至多为 $n \cdot 1/n^\alpha \leq 1/n^{\alpha-1}$。这些未覆盖的元素各自会在 $\mathcal{B}^t$ 中贡献一个集合。因此，对于 t^{th} 更新， $\mathcal{B}$ 的期望重配置成本至多为 $\mathbb{E}\left[\mathcal{B}^t \oplus \mathcal{B}^{t-1}\right] \leq \mathbb{E}\left[|\mathcal{B}^t| + |\mathcal{B}^{t-1}|\right] = O(1/n^{\alpha-1})$。

为了界定 $\mathcal{B}^t$ 的期望成本，注意到每个集合 $B \in \mathcal{B}^t$ 都是因为存在一个未被 $\mathcal{R}^t$ 覆盖的元素 $u \in U^t$ 而被加入的。由于 $B = S(u)$ 是覆盖 u 的成本最低的集合，且 x_i^t 是针对 U^t 的一个可行分数集合覆盖，因此我们有 $c(B) \leq \sum_{i\in[m]} c(S_i) \cdot x_i^t$。根据上述论证，我们现在可得：

$$\mathbb{E}\left[c(B^t)\right] \leq \mathbb{E}\left[|\mathcal{B}^t| \cdot \sum_{i\in[m]} c(S_i) \cdot x_i^t\right] \leq \frac{1}{n^{\alpha-1}} \cdot \sum_{i\in[m]} c(S_i) \cdot x_i^t = O(\alpha \log n) \sum_{i\in[m]} c(S_i) \cdot x_i^t.$$

总而言之， $\mathcal{S}^t = \mathcal{R}^t \cup \mathcal{B}^t$ 的期望总成本至多为 $O(\alpha\log\ n) \cdot \sum_{i\in[m]} c(S_i) \cdot x_i^t$，每次更新的期望重配置成本至多为 $\alpha\log\ n \cdot \|x^t - x^{t-1}\|_1 + O(1/\alpha)$。 □

C 定理证明4.3

定理 4.3（最大匹配舍入）。 对于动态二分图匹配的任意分数解 x^t，以及任意 $\delta \in (0, 1)$，$\alpha \geq 1$，都存在一种随机动态舍入方案，使得：

$$\mathbb{E}\left[\sum_{t\in[T]} \left|M^t \oplus M^{t-1}\right|\right] \leq O\left(\frac{\alpha \log n}{\delta^3}\right) \cdot \sum_{t\in[T]} \|x^t - x^{t-1}\|_1 + O\left(\frac{T}{\delta \cdot n^\alpha}\right). \tag{4.4}$$

$$\mathbb{E}\left[\left|M^t\right|\right] \geq (1-\delta)\cdot\sum_{e\in E^t} x_e^t \qquad \forall t\in[T]. \tag{4.5}$$

我们的取整方案分为两个步骤。第一步，我们维护 G^t 的一个子图 H^t，称为稳定器，使得相对于 x^t 的 ℓ_1 变化， H^t 的重配置有界，并且 H^t 包含一个大小非常接近 $\mathsf{Obj}(x^t) := \sum_{e\in E} x_e^t$ 的匹配。第二步，我们在 H^t 中维护一个 $(1-\delta)$-近似最大匹配，其重配置量相对于 H^t 的更新次数较小。

第一步，我们证明：

引理 C.1. 我们可以维护 $H=(V,E_H)$ 的一个子图 G^t，使得：

(1) $\mathbb{E}[\mu(H^t)] \geq (1-\delta)\cdot \mathtt{Obj}(x^t)$ *at all times* t*, where* $\mu(H^t)$ *is the maximum matching size in* H^t*.*

(2) $\mathbb{E}\left[\sum_t \left|E_H^{t-1}\oplus E_H^t\right|\right] \leq O\left(\frac{\alpha\log n}{\delta^2}\right)\cdot\sum_t \left\|x^t-x^{t-1}\right\|_1 + O\left(\frac{T}{n^\alpha}\right).$

我们暂且推迟引理 C.1 的证明，先说明如何完成定理的证明。

引理 C.2. 我们可以在 H^t 中维护一个$(1-\delta)$-近似最大匹配 M^t，每次在 H^t 中更新时的绝对重配置为 $O(1/\delta)$。

证明. 我们维护一个匹配 M^t，它不包含长度小于 $1+\Theta(1/\delta)$ 的增广路径。这样的匹配 M^t 是在 H^t 中的 $(1-\delta)$-近似最大匹配。如果在 H^t 中的一次更新导致产生一条相对于 M^t、长度小于 $1+\Theta(1/\delta)$ 的增广路径，那么我们就沿着该路径增广 M^t。这意味着在 H^t 中每次更新的绝对重配置为 $O(1/\delta)$。 □

定理4.3现在可由引理C.2 和引理C.1得出。

引理*C.1*的证明。我们按如下方式构造稳定器 H^t。固定参数

$$\kappa := \lceil (100(\alpha+4)\log n)/\delta^2 \rceil. \tag{C.1}$$

在预处理阶段，对于每个 $e=(u,v)\in\binom{V}{2}$ 和每个 $i\in[\kappa]$，我们均匀且独立地随机抽取一个值 $\chi_{e,\,i}\in[0,1]$。定义多重边的集合

$$F^t := \left\{ e_i : e\in\binom{V}{2},\ i\in[\kappa],\ \text{and } x_e > \chi_{e,i} \right\},$$

（注意， F^t 始终是 G^t的边集的子集），并令 z 为加权多重图，其中 F^t 中的每条边赋值为 $((1+\delta)\kappa)^{-1}$。定义 Γ^t 为 z^t 在 F^t中表示一个有效分数匹配的事件。最后，若 Γ^t 成立，则将 H^t 的边集设为 F^t；否则，将 H^t 设为 G^t中的任意最大匹配。

在证明引理C.1之前，我们先证明 z^t 以较高概率是一个高权值分数匹配。为方便起见，令

$$\mathtt{Obj}(z^t) := \sum_{e_i\in F^t} z_{e_i}^t.$$

此外，对于每条边 $e\in E^t$ 和 $i\in[\kappa]$，令 $X_{e,\,i}\in\{0,1\}$ 为事件$e_i\in E_F^t$的指示变量。注意到

$$\mathbb{E}[X_{e,i}] = \Pr\left[x_e > \chi_{e,i}\right] = x_e^t. \tag{C.2}$$

断言 C.3. 对于所有时间步 t，事件 Γ^t 以至少 $1-1/n^{\alpha+3}$ 的概率成立。

证明。 任取一个节点 $v \in V$。令 $\deg^t(v, F)$ 表示在 v 中与其关联的多重边数量 F。由于 $z_{e_i} = (1+\delta)^{-1} \cdot (1/\kappa)$ 对所有 $e_i \in E_F^t$ 均成立，只需证明 $\deg^t(v, F) \le (1+\delta)\kappa$ 以足够高的概率成立。注意到 (C.2) 意味着

$$\mathbb{E}[\deg^t(v, F)] = \sum_{e \in E^t(v),\, i \in [\kappa]} \mathbb{E}[X_{e,i}] = \sum_{e \in E^t(v)} \kappa \cdot x_e^t \le \kappa. \tag{C.3}$$

回顾 κ（见 C.1）。应用切尔诺夫界，

$$\Pr\left[\deg^t(v, F) \le (1+\delta) \cdot \kappa\right] \ge 1 - 1/n^{\alpha+4}. \tag{C.4}$$

对 G^t 中所有 n 个节点应用并集界，即可得出该结论。 □

断言 C.4。对于所有时间步 t，有 $\mathbb{E}[\texttt{Obj}(z^t) \mid \Gamma^t] \ge (1 - \Theta(\delta)) \cdot \texttt{Obj}(x^t) - 1/n^{\alpha+1}$.证明。由（C.2）以及 $z_{e_i}^t = ((1+\delta)\kappa)^{-1}$，

$$\mathbb{E}[\texttt{Obj}(z^t)] = \sum_{e \in E^t} \mathbb{E}\left[\sum_{i \in [\kappa]:\, e_i \in E_F^t} z_{e_i}^t\right] = \sum_{e \in E^t} (1+\delta)^{-1} \cdot x_e^t. \ge (1 - \Theta(\delta)) \cdot \texttt{Obj}(x^t) \tag{C.5}$$

接下来，由于 $|E_F^t| \le \kappa n^2$，可知其以概率 1 成立

$$\texttt{Obj}(z^t) = |E_F^t| \cdot (1+\delta)^{-1}(1/\kappa) \le n^2. \tag{C.6}$$

根据全概率公式

$$\mathbb{E}[\texttt{Obj}(z^t) \mid \Gamma^t] \ge \mathbb{E}[\texttt{Obj}(z^t) \mid \Gamma^t] \cdot \Pr\left[\Gamma^t\right] = \mathbb{E}[\texttt{Obj}(z^t)] - \mathbb{E}\left[\texttt{Obj}(z^t) \;\middle|\; \overline{\Gamma^t}\right] \cdot \Pr\left[\overline{\Gamma^t}\right].$$

利用（C.5）、（C.6）以及断言 C.3，我们得出

$$\mathbb{E}[\texttt{Obj}(z^t) \mid \Gamma^t] \ge (1 - \Theta(\delta)) \cdot \texttt{Obj}(x^t) - \frac{n^2}{n^{\alpha+3}} = (1 - \Theta(\delta)) \cdot \texttt{Obj}(x^t) - 1/n^{\alpha+1}.$$ □

现在我们可以证明引理 C.1 的第 (1) 部分。若 G^t 是空图，则该结论显然成立。否则，根据全概率公式，

$$\begin{aligned}
\mathbb{E}[\mu(H^t)] &= \mathbb{E}[\mu(F^t) \mid \Gamma^t] \cdot \Pr\left[\Gamma^t\right] + \mathbb{E}[\mu(G^t) \mid \overline{\Gamma^t}] \cdot \Pr\left[\overline{\Gamma^t}\right] \\
&\ge \mathbb{E}[\texttt{Obj}(z^t) \mid \Gamma^t] \cdot \Pr\left[\Gamma^t\right] + \texttt{Obj}(x^t) \cdot \Pr\left[\overline{\Gamma^t}\right] && \text{(C.7)} \\
&\ge (1 - \Theta(\delta)) \cdot \texttt{Obj}(x^t) - 1/n^{\alpha+1} && \text{(C.8)} \\
&\ge (1 - \Theta(\delta)) \cdot \texttt{Obj}(x^t). && \text{(C.9)}
\end{aligned}$$

步骤 (C.7) 成立，是因为二分图匹配多面体没有整数间隙；步骤 (C.8) 来自断言 C.4；步骤 (C.9) 成立，是因为 G^t 非空，因此 $1/n^{\alpha+1} \le \delta \cdot \texttt{Obj}(x^t)$。

最后，我们证明第 (2) 部分。对于每个不满足 Γ^t 的时间步，H^t 的重配置至多比 F^t 的重配置多 n^2（因为此时我们需要将图替换为 G^t 中的一个最大匹配，再切换回来，这至多需要 $2 \cdot \binom{n}{2} \le n^2$ 次边插入）。由于 Γ^t 以概率 $1 - 1/n^{\alpha+3}$ 成立，因此每个时间步切换到 G^t 再切换回来的预期代价至多为 $1/n^{\alpha}$。

为了界定 F^t 的重配置成本，注意到边 $e_i \in E_F^t \oplus E_F^{t-1}$ 的每个副本当且仅当随机变量 $\chi_{e,\,i}$（在 $[0,1]$ 上均匀分布）落在长度为 $\left|x_e^t - x_e^{t-1}\right|$ 的区间内时，才会出现，因此

$$\mathbb{E}[\left|E_F^t \oplus E_F^{t-1}\right|] = \kappa \cdot \sum_{e \in \binom{V}{2}} \left|x_e^t - x_e^{t-1}\right| = \kappa \cdot \left\|x^t - x^{t-1}\right\|_1.$$

对所有时间步 t 求和并综合以上结果，H^t 的总期望重配置成本至多为

$$\mathbb{E}\left[\sum_t \left|E_H^t \oplus E_H^{t-1}\right|\right] \le \kappa \cdot \sum_t \left\|x^t - x^{t-1}\right\|_1 + \sum_t 1/n^\alpha. \qquad \square$$

D 定理 4.7 的证明

定理 4.7 (最小生成树舍入)。 考虑动态最小生成树问题的任意分数解 x^t ，以及任意 $\delta > 0$, $\alpha \ge 1$。那么，存在一种随机动态舍入方案，可维护 G^t 的一棵生成树 $\mathcal{T}^t = (V, E_{\mathcal{T}}^t)$，使得：

$$\mathbb{E}\left[\sum_{t\in[T]} \left|E_{\mathcal{T}}^t \oplus E_{\mathcal{T}}^{t-1}\right|\right] \le O\left(\frac{\alpha \log n}{\delta^2}\right) \cdot \sum_{t\in[T]} \left\|x^t - x^{t-1}\right\|_1 + \frac{T}{O(n^\alpha)}, \tag{4.16}$$

$$\sum_{e\in E_{\mathcal{T}}^t} c_e \le (2+\delta) \cdot \sum_{e\in E^t} c_e \cdot x_e^t \qquad \forall t \in [T]. \tag{4.17}$$

我们沿用附录 C 中的相同范式，即维护一个稳定器；在此情形下，它是一个图，其中包含一棵低成本生成树，其重配置相对于 ℓ_1 的移动 x有界；然后针对稳定器的更新，给出一种绝对重配置算法。

首先，我们证明：

引理 D.1。 我们可以维护 G^t 的一个子图 $H = (V, E_H)$，使得：
(1) $\mathbb{E}\left[\mu(H^t)\right] \le (2+\delta)\sum_{e\in E^t} c_e \cdot x_e^t$，其中 $\mu(H^t)$ 是 H^t 的最小生成树的期望成本。

(2) $\mathbb{E}\left[\sum_t \left|E_H^{t-1} \oplus E_H^t\right|\right] \le O\left(\frac{\alpha \log n}{\delta^2}\right) \cdot \sum_t \left\|x^t - x^{t-1}\right\|_1 + O\left(\frac{T}{n^\alpha}\right)$.

再次，我们先完成证明，然后再给出稳定器的构造 n.

引理 D.2. 我们可以维护 F 的一棵最小生成树 $\mathcal{T}$，在 F 中每次边的插入/删除仅需承担每条边 $O(1)$ 的绝对重配置。

证明. 当边 e 从 F 中删除时，如果 $e \notin E_{\mathcal{T}}$，我们不对 $\mathcal{T}$ 做任何更改；否则，将 e 替换为 $E_{\mathcal{T}}$ 中跨越其基本割的最便宜边。当一条边 e 被加入 F 时，将 e 加入 $E_{\mathcal{T}}$，然后从 $E_{\mathcal{T}}$ 中删除与 e 相关的基本环上最便宜的边。在这两种情况下，更新后 $\mathcal{T}$ 仍然是最小生成树，且产生绝对重配置 $O(1)$。 $\square$

结合引理 D.1 和引理 D.2 即可得到定理 4.7。接下来，我们说明如何构造稳定器。

证明 引理 *D.1*。 设 $\gamma > 0$ 为一个足够大的常数。对于每条边 $e \in \binom{V}{2}$，定义 e

$$p_e^t := \min\left\{1, \frac{100\gamma(\alpha+3)\log n \cdot x_e^t}{\delta^2}\right\} \tag{D.1}$$

在预处理时，对于每条边 $e \in \binom{V}{2}$，我们均匀且独立地随机抽取一个值 $\chi_e \in [0,1]$，并定义边集 $F^t := \left\{e \in \binom{V}{2} : p_e^t > \chi_e\right\}$（注意， F^t 始终是活跃边 E^t的子集）。令 Γ^t 表示事件： F^t 包含一棵成本不超过

$(2+\delta)\cdot\sum_{e\in E^t} c_e\cdot x^t_e$。最后，定义 H^t 的边集为：若 Γ^t 成立，则取 F^t；否则令 H^t为 G^t 的一棵任意最小生成树。

引理 D.1 的第 (1) 部分可由构造立即得出（因为最小生成树的割 LP 表述的整数间隙至多为 2），接下来只需证明第 (2) 部分。为此，我们证明 H^t 以较高概率包含一棵成本低廉的生成树。

命题 D.3。对于所有时间步 t，事件 Γ^t 成立的概率至少为 $1-1/n^{\alpha+3}$.

证明。我们沿用 [CQ18] 中引理 3.8 的证明。设 (G^t,x^t) 为边权按如下方式定义的加权图 x^t。定义边 e 的强度，记作 κ^t_e，为满足以下条件的 k 的最大值：图 k 的极大 G^t -连通诱导子图包含 e。

图 (G^t,x^t) 中的最小割值为 ≥ 1，因为 x^t 满足（4.15），因此对于所有 $\kappa^t_e\geq 1$ $e\in E^t$。现在，对每条边 $e\in E^t$，我们定义

$$q^t_e := \min\left\{1, \frac{\gamma(\alpha+3)\log n\cdot c_e\cdot x^t_e}{\delta^2\cdot\sum_{e'\in E^t} c_{e'}\cdot x^t_{e'}}\right\} \text{ and } r^t_e := \max\{p^t_e, q^t_e\}. \tag{D.2}$$

现在，考虑加权图 (Z^t,z^t)，其边集为 $E^t_Z := \left\{e\in\binom{V}{2}:\ r^t_e>\chi_e\right\}$，其中边 $e\in E^t_Z$ 的权重为 $z^t_e := (1+\delta)\cdot x^t_e/r^t_e$。令 $\mathcal{E}^t$ 表示以下事件：加权图 (Z^t,z^t) 的最小割至少为 1，且具有 $\sum_{e\in E^t_Z} c_e\cdot z^t_e\leq(1+\delta)\cdot\sum_{e\in E^t} c_e\cdot x^t_e$。由于最小生成树的割 LP 表述的整数间隙至多为 2，事件 $\mathcal{E}^t$ 进而意味着，Z^t 包含一棵整数生成树 $\mathcal{T}'$，其成本至多为 $2(1+\delta)\cdot\sum_{e\in E^t} c_e\cdot x^t_e$。

按照引理 3.8 证明中的分析，在 [CQ18], 中，我们推导出 $\mathcal{E}^t$ 以至少 $1-1/n^{\alpha+3}$的概率成立。在 $\mathcal{E}^t$成立的条件下，考虑任意一条边 $e^*\in E^t_Z\setminus E^t_F$。我们有 $p^t_{e^*}\leq\chi_{e^*}<r^t_{e^*}$。由于 $\chi_{e^*}\in[0,1]$，回顾 $p^t_{e^*}$ 和 $r^t_{e^*}$ 在（D.1）和（D.2）中的取值，我们推断出

$$\frac{100\gamma(\alpha+3)\log n\cdot x^t_{e^*}}{\delta^2}\leq\frac{\gamma(\alpha+3)\log n\cdot c_{e^*}\cdot x^t_{e^*}}{\delta^2\cdot\sum_{e\in E^t} c_e\cdot x^t_e}.$$

重新整理上述不等式中的各项，可得 $c_{e^*}\geq 100\cdot\sum_{e\in E^t} c_e\cdot x^t_e$，这意味着 e^* 不在 $\mathcal{T}'$中，即成本为 $2(1+\delta)\cdot\sum_{e\in E^t} c_e\cdot x^t_e$ 在 Z^t中的生成树。换句话说，$\mathcal{T}$的边也包含在 F^t中，这正是我们要证明的结论。 □

我们最后证明引理 D.1 的第 (2) 部分。的重配置至多为 F^t的重配置，再加上 n^2 每个不成立的时间步所需的成本，因为在这种情况下，我们需要将图替换为生成树，切换到 G^t 再切回，这至多需要 $2\cdot\binom{n}{2}\leq n^2$ 次边插入）。由于 Γ^t 以概率 $1-1/n^{\alpha+3}$成立，因此切换到 G^t 再切回的预期贡献至多为 $1/n^\alpha$ 每个时间步。

为了界定 F^t 的重配置，注意到每条边 $e\in E^t_F\oplus E^{t-1}_F$ 当且仅当随机变量 χ_e（其在 $[0,1]$ 上服从均匀分布）落在长度为 $\left|p^t_e-p^{t-1}_e\right|=O(\alpha\log n/\delta^2)$ $\left|x^t_e-x^{t-1}_e\right|$的区间内，因此

$$\mathbb{E}[\left|E^t_F\oplus E^{t-1}_F\right|]=O\left(\frac{\alpha\log n}{\delta^2}\right)\cdot\sum_{e\in\binom{V}{2}}\left|x^t_e-x^{t-1}_e\right|=O\left(\frac{\alpha\log n}{\delta^2}\right)\cdot\left\|x^t-x^{t-1}\right\|_1.$$

对所有时间步求和 t并将各部分合并，H^t 的总期望重配置成本至多为

$$\mathbb{E}\left[\sum_t\left|E^t_H\oplus E^{t-1}_H\right|\right]\leq O\left(\frac{\alpha\log n}{\delta^2}\right)\cdot\sum_t\left\|x^t-x^{t-1}\right\|_1+\sum_t O(1/n^\alpha).$$ □

E 下界

本节将证明，要在正体追逐中获得次多项式竞争比，资源增强是必不可少的。

定理 1.2. 对于正体追逐，不存在竞争比优于

$$\Omega\left(\max\left\{\min\left(\frac{1}{\epsilon\sqrt{\log(1/\epsilon)}},\sqrt{n}\right),\log n\right\}\right).$$

我们下界证明的关键是以下引理。

引理 E.1. 对于正体追逐，不存在竞争比优于

$$\max\left\{\frac{\sqrt{n}(1-10\epsilon\cdot\sqrt{n\log n})}{10},0\right\}.$$

我们先来看这如何推出定理 1.2。

定理 *1.2* 的证明。 首先注意，$\Omega(\log n)$ 分数型在线集合覆盖的下界（例如参见 [BN09b]）意味着正体追逐存在一个 $\Omega(\log n)$ 下界，即使在 ϵ 资源增强下也是如此，其中 $\epsilon\in(0,1]$。

接下来只需证明该下界的剩余部分。若 $\epsilon\le 1/(20\sqrt{n\log n})$，则引理 E.1 给出 $\Omega(\sqrt{n})$ 的下界。另一方面，若 $\epsilon > 1/(20\sqrt{n\log n})$，则在维数为 $n'=1/(900\cdot\epsilon^2\log(1/\epsilon))$ 的子空间上应用引理 E.1 中的相同构造，$\le n$ 可推出 ~~$\Omega(\sqrt{n'})$~~ $=\Omega(1/\epsilon\sqrt{\log(1/\epsilon)})$ 的下界。 □

最后，我们证明主要引理。

引理 E.1 的证明。假设对于足够大的 k，$n=2^k$ 成立。此外，假设 $\epsilon\le 1/10\sqrt{n\log n}$，否则无需证明。令 h^i 为阶数为 n 的 Hadamard 矩阵的第 i^{th} 行。定义 $S=\sqrt{n}(2\sqrt{\log n}+1)$，以及 $(x^0,s^0)=(2\sqrt{(\log n)/n}\cdot\vec{1},\sqrt{n})$。注意，$S=\|(x^0,s^0)\|_1$。

我们将以下实例执行 M 个阶段。在每个阶段，对手均匀随机地选择一个向量 $b\in\{-1,1\}^n$。一个阶段由 n 个时间步组成，其中时间 $t\in[n]$ 时的身体定义为

$$K_t=\left\{(x,s)\in R_+^{n+1}\;\middle|\;\langle h^t,x\rangle+s+\sum_{i=1}^n x_i\ge b^t+S+\langle h^t,x^0\rangle\right\}\qquad\text{if } b^t=1$$

$$K_t=\left\{(x,s)\in R_+^{n+1}\;\middle|\;\langle h^t,x\rangle+s+\sum_{i=1}^n x_i\le b^t+S+\langle h^t,x^0\rangle\right\}\qquad\text{if } b^t=-1.$$

我们声称这是一个正体追逐实例。首先检查每个 K_t 的约束是否为覆盖/装箱约束。对于每个约束的左侧，每个变量的系数都至少为零，因为 h^t 的元素属于 $\{+1,-1\}$，因此 x_i 的系数为 $h_i^t+1\ge 0$。此外，s 的系数为 1。对于右侧，我们有 $b^t+S+\langle h^t、x^0\rangle\ge 0$，因为根据 Hadamard 矩阵的定义，当 $t>0$ 时，$\langle h^t、x^0\rangle$ 等于 0；而当 $t=0$时，它们等于 $\|x^0\|$。剩下要证明的是 $K_t\ne\emptyset$。注意，$(x^0,s^0+b^t)\in K_t$，事实上，该约束是紧的：$\langle h^t,x_0\rangle+(s^0+b^t)+\sum_{i=1}^n x_i=b^t+S+\langle h^t,x^0\rangle$，因为 $S=\sum_i x_i+s_0$。

Alg 的移动成本$\textsc{Alg}$： 我们认为，任何在线算法每个阶段的预期成本至少为 $n/2 \cdot (1 - 10\epsilon \cdot \sqrt{n \log n})$。

为方便起见，定义 $L^t = \langle h^t, x^t \rangle + s + \sum_i x_i$ 为上述约束的左侧，并令 $R^t = S + \langle h^t, x^0 \rangle$。算法需要完全满足覆盖约束，并满足至多为 $(1+\epsilon)$ 的装箱约束；如果 $b^t = 1$，算法必须选择 x^t，使得

$$L^t \geq R^t + 1,$$

如果 $b^t = -1$，算法必须选择 x^t，使得

$$L^t \leq (R^t - 1)(1 + \epsilon).$$

项 L^t 的变化幅度在期望上必须至少达到 $R^t + 1$ 与 $(R^t - 1)(1+\epsilon)$ 之间距离的一半，即

$$\mathbb{E}\left[|L^t - L^{t-1}|\right] \geq \frac{(R^t + 1) - (R^t - 1)(1+\epsilon)}{2} = 1 - \frac{\epsilon(R^t - 1)}{2}.$$

由于 $L^t = \langle l^t, (x^t, s^t) \rangle$，其中 l^t 是一个系数取自 $[0, 2]$, 的向量，因此这意味着

$$\mathbb{E}\left[\|(x^t, s^t) - (x^{t-1}, s^{t-1})\|_1\right] \geq \frac{1}{2} - \frac{\epsilon(R^t - 1)}{4} = \frac{1 - 10\epsilon \cdot \sqrt{n \log n}}{2}.$$

因此，任何算法在一个包含 n 个时间步的阶段中的期望成本至少为 $n/2 \cdot (1 - 10\epsilon \cdot \sqrt{n \log n})$。

Opt 的移动$\textsc{Opt}$： 我们论证 Opt 在每个阶段的期望成本至多为$5\sqrt{n}$。首先假设 Opt 始终在阶段结束时返回(x^0, s^0)，因为这只会增加 Opt 的成本。

首先注意到，由于$(x^0, s^0 + b^t) \in K_t$，因此最优解可以在每个时刻 t移动到该点，并在最后返回(x^0, s^0)。该解的总移动成本上界为 $4\sum_{i=1}^{n} |b^t| = 4n$。然而，我们声称 Opt 以高概率可以获得一个更优的解。

令 $\mathcal{E}$ 表示事件 $\|H^{-1}b\|_\infty \leq 2\sqrt{(\log n)/n}$。由于

$$\|H^{-1}b\|_2 \leq \|H^{-1}\| \cdot \|b\|_2 \leq \frac{1}{\sqrt{n}} \cdot \sqrt{n} = 1,$$

我们有 $\|H^{-1}b\|_1 \leq \sqrt{n} \cdot \|H^{-1}b\|_2 \leq \sqrt{n}$。定义

$$x^* = x^0 + H^{-1}b = 2\sqrt{\frac{\log n}{n}} \cdot \vec{1} + H^{-1}b,$$
$$s^* = S - \sum_i x_i^* \geq \sqrt{n} - \|H^{-1}b\|_1$$

现在，对于所有 $t \in [n]$，根据 x^*的定义，由于 $\langle h^t, x^* \rangle = \langle h^t, x^0 \rangle + b^t$，我们有按设计

$$\langle h^t, x^* \rangle + s^* + \sum_{i=1}^{n} x_i^* = b^t + S + \langle h^t, x^0 \rangle.$$

此外，如果 $\mathcal{E}$ 成立，那么利用 $\|H^{-1}b\|_1 \leq \sqrt{n}$ 这一事实，点 $(x^*, s^*) \in R_+^{n+1}$，因此 $(x^*, s^*) \in \bigcap_{t=1}^{n} K_t$。因此，Opt 可以直接从 (x^0, s^0) 移动到 (x^*, s^*)，并在阶段结束时返回，代价至多为 $4\sqrt{n}$。

接下来需要界定 $\mathcal{E}$ 的概率。矩阵 H^{-1} 的元素属于 $\{-1/n, 1/n\}$，因此 $H^{-1}b$ 的每个元素都是步长为 $1/n$ 的一维随机游走。

事实 E.2 (切尔诺夫界)。 如果 $Z = \sum_{i=1}^{n} X_i$，其中 X_i 是独立随机变量，且 概率 $[X_i = 1] =$ 概率 $[X_i = -1] = 1/2$，那么

$$\Pr\left[|Z| \geq \sqrt{an \log n}\right] \leq 2n^{-a/2}.$$

根据这一事实并结合并集界，事件 $\|H^{-1}b\|_\infty \leq 2\sqrt{(\log n)/n}$ 以概率$1-2/n$发生。因此，当 n 足够大时，Opt 在整个阶段的期望成本为 $(1-2/n)\cdot 4\sqrt{n}+2/n\cdot 4n \leq 5\sqrt{n}$。

总之，对 M 个阶段求和，并考虑到 Opt 在第一个阶段之前移动到 (x^0, s^0)，任何算法的竞争比至少为

$$\frac{\mathbb{E}[\text{ALG}]}{\mathbb{E}[\text{OPT}]} \geq \frac{M \cdot n/2 \cdot (1 - 10\epsilon \cdot \sqrt{n \log n})}{5 \cdot M \cdot \sqrt{n} + \|(x^0, s^0)\|_1} \geq \frac{\sqrt{n}(1 - 10\epsilon \cdot \sqrt{n \log n})}{10} \cdot (1 - o(1)). \qquad \square$$

F 处理盒约束

本节将说明，我们可以无违例地处理盒约束这一类特殊的装箱约束。盒约束是形式为 $x_j \leq b_j$ 的“静态”装箱约束。
一般而言，我们允许在时刻 t 向多面体中添加盒约束 $x \leq b$，其中 $b \in R_+^n \cup \{\infty\}$。因此，时刻 t 更一般的多面体为：

$$K_t^{1+\epsilon} = \left\{x^t \in R_+^n \mid x \leq b,\ C^t x^t \geq 1,\ P^t x^t \leq 1 + \epsilon\right\}$$

我们声称，只需添加额外的覆盖约束，就可以消除这些新约束。对于每个覆盖约束 $\sum_j c_{ij} x_j \geq 1$，以及变量的每个子集 S，我们添加约束：

$$\sum_{j \notin S} c_{ij} x_j \geq \max\left\{0, 1 - \sum_{j \in S} c_{ij} \cdot b_j\right\}$$

很容易验证，如果 $x \in R_+^n$ 满足这些新约束，那么向量 x'（其中 $x'_j = \min\{x_j, b_j\}$）就满足原有约束以及盒约束。另一方面，任何满足盒约束的 x'' 都满足所有新的覆盖约束。

因此，我们的算法为 $K_t^{1+\epsilon}$ 维护一个可行点，但移除了盒约束，转而加入新的覆盖约束。算法在时刻 t 维护向量 x'，其中 $x'_j = \min\{x_j, b_j\}$，其总移动成本不超过 x 的移动成本。

最后需要指出，尽管此类约束的数量呈指数级增长，但只需检查 x 是否满足带有 $S = \{j \mid x_j \geq b_j\}$ 的覆盖约束，就很容易验证解 x 是否满足这些新约束。

参考文献

[AAA+06] N. Alon、B. Awerbuch、Y. Azar、N. Buchbinder 和 J. Naor。在线网络优化问题的一般方法。*ACM* 算法汇刊，2(4):640–660，2006。 → 引用于第 4AAA+09] Noga Alon、Baruch Awerbuch、Yossi Azar、Niv Buchbinder 和 Joseph Naor。在线集合覆盖问题。*SIAM J. Comput.*，39(2):361–370，2009。 → 引用于第 4[AAG+19] Amir Abboud、Raghavendra Addanki、Fabrizio Grandoni、Debmalya Panigrahi 和 Barna Saha。动态集合覆盖：改进的算法与下界。在 Moses Charikar 和 Edith Cohen（编辑）编，第*51*届 *ACM SIGACT* 年度计算理论研讨会论文集，*STOC 2019*，美国亚利桑那州菲尼克斯，*2019*年*6*月*23*日至*26*日，第114–125页。ACM，2019。 → 引用于第 2

[AAPW01] Baruch Awerbuch、Yossi Azar、Serge A. Plotkin 和 Orli Waarts。未知持续时间虚拟电路的竞争性路由。J. Comput. Syst. Sci.，62(3):385–397，2001。 → 引用于第2页[ABC+16] Yossi Azar、Niv Buchbinder、T.-H. Hubert Chan、Shahar Chen、Ilan Reuven Cohen、Anupam Gupta、Zhiyi Huang、Ning Kang、Viswanath Nagarajan、Joseph Naor 和 Debmalya Panigrahi。具有凸目标的覆盖与打包问题的在线算法。在 IEEE 第57届计算机科学基础年度研讨会，FOCS 2016，第148–157页，2016年。 → 引用于第4页[ABC+19] C. J. Argue、Sébastien Bubeck、Michael B. Cohen、Anupam Gupta 和 Yin Tat Lee。追逐嵌套凸体的近线性界。在第三十届 ACM-SIAM 离散算法研讨会论文集，SODA 2019，美国加利福尼亚州圣迭戈，2019年1月6日至9日，第117–122页。SIAM，2019年。 → 引用于第1页[ABFP13] Yossi Azar、Umang Bhaskar、Lisa Fleischer 和 Debmalya Panigrahi。在线混合打包与覆盖。在第二十四届 ACM-SIAM 离散算法研讨会论文集，SODA 2013，第85–100页，2013年。 → 引用于第4页[ABL+20] Chen Avin、Marcin Bienkowski、Andreas Loukas、Maciej Pacut 和 Stefan Schmid。动态平衡图划分。SIAM 离散数学杂志，34(3):1791–1812，2020年。 →引用于第4页[ACC+18] Moab Arar、Shiri Chechik、Sarel Cohen、Cliff Stein 和 David Wajc。动态匹配：将整数算法约化为近似最大分数算法。在第45届自动机、语言与编程国际会议（ICALP），第

107 7 1 7 16, 卷，LIPIcs，2018 →

第：–: 页。引用于第[页[AGTG21] C. J. Argue、Anupam Gupta、Ziye Tang 和 Guru Guruganesh。以线性竞争比追逐凸体。J. ACM，68(5):32:1–32:10，2021年。 → 引用于第1、2、5页[ALPS16] Chen Avin、Andreas Loukas、Maciej Pacut 和 Stefan Schmid。在线平衡再分区。在《分布式计算：第30届国际研讨会，DISC 2016》，法国巴黎，2016年9月27日至29日。论文集，第243–256页。施普林格，2016年。 →引用于第4页[AMPST23] Yossi Azar、Chay Machluf、Boaz Patt-Shamir 和 Noam Touitou。竞争性顶点重着色：（在线脱离）。Algorithmica，第1–27页，2023年。 → 引用于第4页[AOSS18] Sepehr Assadi、Krzysztof Onak、Baruch Schieber 和 Shay Solomon。具有次线性更新时间的全动态极大独立集。在第50届 ACM SIGACT计算理论研讨会论文集，第815–826页，2018年。 →引用于第2页[AS21] Sepehr Assadi 和 Shay Solomon。通过超图极大匹配实现全动态集合覆盖：通过局部方法获得最优近似。在 Petra Mutzel、Rasmus Pagh 和 Grzegorz Herman（编辑）编，《第29届欧洲算法研讨会，ESA 2021》，2021年9月6日至8日，葡萄牙里斯本（线上会议），第204卷 LIPIcs，第8:1–8:18页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2021年。 → 引用于第2页[BBCG22] Suman K Bera、Sayan Bhattacharya、Jayesh Choudhari 和 Prantar Ghosh。稠密子超图的新型动态算法。在 ACM Web Conference 2022 论文集，第1093–1103页，2022年。 →引用于第2、4页[BBE+20] Nikhil Bansal、Martin Böhm、Marek Eliás、Grigorios Koumoutsos 和 Seeun William Umboh。嵌套凸体是可追逐的。Algorithmica，82(6):1640–1653，2020年。 →引用于第1、6页[BBN10] Nikhil Bansal、Niv Buchbinder 和 Joseph Naor。HST 上的度量任务系统与 k-服务器问题。在 ICALP（1），第287–298页，2010年。 → 引用于第4页[BBN12a] Nikhil Bansal、Niv Buchbinder 和 Joseph Naor。加权分页的原始-对偶随机化算法。J. ACM，59(4):19，2012年。 → 引用于第4页[BBN12b] Nikhil Bansal、Niv Buchbinder 和 Joseph Naor。广义缓存的随机化竞争算法。SIAM J. Comput.，41(2):391–414，2012年。 → 引用于第4页[BCL+18] Sébastien Bubeck、Michael B. Cohen、Yin Tat Lee、James R. Lee 和 Aleksander Madry。通过多尺度熵正则化实现 k-服务器问题。在第50届 ACM SIGACT计算理论研讨会论文集，STOC 2018，第3–16页，2018年。 → 引用于第4页[BCLL21] Sébastien Bubeck、Michael B. Cohen、James R. Lee 和 Yin Tat Lee。通过镜像下降与非公平粘合处理树上的度量任务系统。SIAM J. Comput.，50(3):909–923，2021年。 → 引用于第4页[BDL21] Aaron Bernstein、Aditi Dudeja 和 Zachary Langley。加权图动态匹配的框架。在第53届 ACM SIGACT计算理论研讨会（STOC），第668–681页。ACM，2021年。 → 引用于第17页

[BF99] Gerth Stølting Brodal 和 Rolf Fagerberg。稀疏图的动态表示。载于《算法与数据结构》：*6 99 11 14,* 届国际研讨会，WADS' 加拿大温哥华，8月——论文集，页——。施普林格，。引文见第[BGMN19] Niv Buchbinder、Anupam Gupta、Marco Molinaro 和 Joseph (Seffi) Naor。 *k*服务器带着微笑：通过投影实现在线算法。载于 ACM-SIAM 离散算法研讨会，SODA 2019，已接收，2019 年。 → 引文见第 4、5 页[BHN19] Sayan Bhattacharya、Monika Henzinger 和 Danupon Nanongkai。动态集合覆盖的新确定性算法。载于 David Zuckerman（编），第 60 届 IEEE 计算机科学基础年会，FOCS 2019，美国马里兰州巴尔的摩，2019 年 11 月 9—12 日，页 406—423。IEEE计算机学会，2019 年。 → 引文见第 2 页[BHNW21] Sayan Bhattacharya、Monika Henzinger、Danupon Nanongkai 和 Xiaowei Wu。动态集合覆盖：改进的摊还和最坏情况更新时间。载于 Dániel Marx（编），2021 年 ACM-SIAM 离散算法研讨会论文集，SODA *2021, 10 13,* 线上会议，1 月——页——。SIAM，。引文见第[BJN07] Niv Buchbinder、Kamal Jain 和 Joseph (Seffi) Naor。用于最大化广告拍卖收入的在线原始对偶算法。载于第 15 届欧洲研讨会论文集，页 253—264，2007 年。 → 引文见第 4 页[BKL+20] Sébastien Bubeck、Bo'az Klartag、Yin Tat Lee、Yuanzhi Li 和 Mark Sellke。近乎最优地追逐嵌套凸体。载于 2020 年 ACM-SIAM 离散算法研讨会论文集，SODA 2020，页 1496—1508。SIAM，2020 年。 → 引文见第 1、2 页[BKS12] Surender Baswana、Sumeet Khurana 和 Soumojit Sarkar。用于图伸展器的全动态随机化算法。《ACM 算法汇刊》（TALG），8(4)：1—51，2012 年。 →引文见第 2 页[BKS23] Sayan Bhattacharya、Peter Kiss 和 Thatchaphol Saranurak。通过乘法权重更新实现打包与覆盖线性规划的动态算法。载于 2023 年 ACM-SIAM 离散算法研讨会（SODA）论文集，页 1—47。SIAM，2023 年。 → 引文见第 4 页[BLP22] Sayan Bhattacharya、Silvio Lattanzi 和 Nikos Parotsidis。高效且稳定的全动态设施选址。载于第三十六届神经信息处理系统会议（NeurIPS），2022 年。→ 引文见第 2 页[BN09a] Niv Buchbinder 和 Joseph Naor。通过原始对偶方法设计竞争性在线算法。《理论计算机科学基础与趋势》，3(2-3)：93—263，2009 年。 →引文见第 5、10 页[BN09b] Niv Buchbinder 和 Joseph Naor。用于覆盖与打包的在线原始对偶算法。《数学运筹学研究》，34(2)：270—286，2009 年。 → 引文见第 2、3、4、26 页[BSS22] Sayan Bhattacharya、Thatchaphol Saranurak 和 Pattara Sukprasert。针对自适应对手的具有近乎最优重配置的简单动态伸展器。载于第 30 届欧洲算法研讨会（ESA），LIPIcs 第 244 卷，页 17:1—17:19，2022 年。 → 引文见第 2 页[CLNT22] Christian Coester、Roie Levin、Joseph (Seffi) Naor 和 Ohad Talmon。块感知缓存的竞争性算法。载于 SPAA '22：第 34 届 ACM算法与架构并行性研讨会，美国宾夕法尼亚州费城，2022 年 7 月 11—14 日，页 161—172。ACM，2022 年。 →引文见第 4 页[CQ18] Chandra Chekuri 和 Kent Quanrud。通过线性规划快速近似度量旅行商问题。CoRR，abs/1802.01242，2018 年。 → 引文见第 25 页[Csi84] Imre Csiszár。Sanov性质、广义i投影与条件极限定理。《概率年刊》，页 768—793，1984 年。 → 引文见第 5 页[FL93] Joel Friedman 和 Nathan Linial。关于凸体追逐。《离散与计算几何》，9：293—321，1993 年。 → 引文见第 1、2 页[GGK16] Albert Gu、Anupam Gupta 和 Amit Kumar。延迟处理的力量：在线维护常数竞争比斯坦纳树。《SIAM计算杂志》，45(1)：1—28，2016 年。 → 引文见第 2 页[GK14] Anupam Gupta 和 Amit Kumar。带删除操作的在线斯坦纳树。载于 Chandra Chekuri（编），第二十五届 ACM-SIAM 离散算法研讨会论文集，SODA 2014，美国俄勒冈州波特兰，2014 年 1 月 5—7 日，页 455—467。SIAM，2014 年。 → 引文见第 2 页[GKKP17] Anupam Gupta、Ravishankar Krishnaswamy、Amit Kumar 和 Debmalya Panigrahi。集合覆盖的在线和动态算法。载于第 49 届 ACM SIGACT计算理论研讨会，STOC 2017，加拿大魁北克省蒙特利尔，2017 年 6 月 19—23 日，页 537—550。ACM，2017 年。 → 引文见第 2 页

[GKL21] Anupam Gupta、Gregory Kehne 和 Roie Levin。随机顺序在线集合覆盖与离线集合覆盖同样容易。载于第 62 届 IEEE 计算机科学基础年会，FOCS 2021，页 1253—1264。IEEE，2021 年。 → 引文见第 2 页[GKLX20] Xiangyu Guo、Janardhan Kulkarni、Shi Li 和 Jiayi Xian。在线和动态模型中的设施选址问题。载于《近似、随机化与组合优化。算法与技术》（APPROX/RANDOM 2020）。Schloss Dagstuhl—Leibniz信息学中心，2020 年。 → 引文见第 2 页[GKS14] Anupam Gupta、Amit Kumar 和 Cliff Stein。在线维护分配：匹配、调度和流。载于第二十五届 ACM-SIAM 离散算法研讨会论文集，SODA 2014，页 468—479。SIAM，2014 年。 → 引文见第 2 页[GL20a] Anupam Gupta 和 Roie Levin。具有有界重配置的全动态次模覆盖。载于 Sandy Irani（编），第 61 届 IEEE 计算机科学基础年会，FOCS 2020，美国北卡罗来纳州达勒姆，2020 年 11 月 16—19 日，页 1147—1157。IEEE，2020 年。 → 引文见第 2 页[GL20b] Anupam Gupta 和 Roie Levin。在线次模覆盖问题。载于 Shuchi Chawla（编），2020 年 ACM-SIAM 离散算法研讨会论文集，SODA 2020，美国犹他州盐湖城，2020 年 1 月 5—8 日，页 1525—1537。SIAM，2020 年。 →引文见第 3 页[GN14] Anupam Gupta 和 Viswanath Nagarajan。在线近似稀疏覆盖整数规划。《数学运筹学研究》，39(4)：998—1011，2014 年。 → 引文见第 3、4 页[GTW14] Anupam Gupta、Kunal Talwar 和 Udi Wieder。改变基：拟阵和匹配的多阶段优化。载于《自动机、语言与编程》—第 41 届国际研讨会，ICALP 2014，丹麦哥本哈根，2014 年 7 月 8—11 日，论文集，第 I 部分，页 563—575，2014 年。 →引文见第 4 页[IW91] Makoto Imase 和 Bernard M. Waxman。动态斯坦纳树问题。《SIAM离散数学杂志》，4(3)：369—384，1991 年。 →引文见第 2 页[KKR01] A. R. Karlin、C. Kenyon 和 D. Randall。动态TCP确认及其他关于 e/(e-1) 的故事。载于 ACM计算理论研讨会论文集，页 502—509，2001 年。 →引文见第 4 页[KLS23] Ravishankar Krishnaswamy、Shi Li 和 Varun Suriyanarayana。在线无关机器负载均衡与带重配置的广义流。载于 STOC，2023 年。 → 引文见第 2、18 页[ŁOP+15] Jakub Łącki、Jakub Oćwieja、Marcin Pilipczuk、Piotr Sankowski 和 Anna Zych。动态距离预言机的力量：用于斯坦纳树的高效动态算法。载于第四十七届 ACM计算理论研讨会论文集，页 11—20，2015 年。 →引文见第 2 页[NPS11] Joseph Naor、Debmalya Panigrahi 和 Mohit Singh。在线点加权斯坦纳树及相关问题。载于 IEEE

52 210 219, 届计算机科2011 →

学基础年会，FOCS，页4——。引文见第[Sel20] Mark Sellke。最优地追逐凸体。载于 2020 年 ACM-SIAM 离散算法研讨会论文集，SODA 2020，页 1509—1518。SIAM，2020 年。 → 引文见第 1、2、5 页[SW20a] Saurabh Sawlani 和 Junxing Wang。近乎最优的全动态最稠密子图。载于第 52 届 ACM SIGACT计算理论研讨会论文集，页 181—193，2020 年。 →引文见第 2 页[SW20b] Shay Solomon 和 Nicole Wein。改进的动态图着色。《ACM 算法汇刊》（TALG），16(3)：1—24，2020 年。 →引文见第 2 页