arXiv:2303.01453v1 [cs.DS] 2 Mar 2023

专家学习的改进空间界

AndersAamand JustinY. Chen HuyLˆeNguyˆễn 麻省理工学院 麻省理工学院 东北大学
aamand@mit.edu justc@mit.edu hu.nguyen@northeastern.edu
Sandeep Silwal 麻省理工学院silwal@mit.edu

摘要

对于包含 n 位专家、持续 T 天的专家建议在线学习问题，我们给出了空间与遗憾之间更优的权衡。给定用于 $\delta \in$ （0，1）的空间预算 n^δ，我们提供了一种遗憾为 $\tilde{O}(n^2T^{1/(1+\delta)})$ 的算法，优于近期工作 [PZ23] 中的遗憾界 $\tilde{O}(n^2T^{2/(2+\delta)})$。这种改进在 $\delta \to 1$ 的情形下尤为显著，此时我们的算法的遗憾趋近于 $\tilde{O}_n(\sqrt{T})$，与无空间限制的标准在线设置中的 T 依赖关系相匹配。

1 引言

理解信息约束下学习算法的性能，是机器学习中的一个基础研究方向。尽管在线学习中的遗憾等性能概念已得到充分研究，近期一系列工作开始探索学习中的额外约束，尤其关注有限内存 [Sha14、WS19、MSSV22] （另见第3节）。

本文聚焦于带有内存约束的专家在线学习问题，这是一种用于序贯决策的通用框架。在专家在线学习问题中，算法必须根据 n 位专家的预测，对未来连续 T 天内某个事件的结果进行预测。在时刻 $t \le T$，算法的预测只能依赖于它在前几天收到的信息，以及专家对第 t天的预测。作出预测后，真实结果会被揭示，算法和所有专家都会承受一定的损失（这可能取决于预测的准确性）。除了在线专家问题已在众多算法应用中得到广泛应用 [AHK12], 之外，鉴于现有算法会显式跟踪每位专家的累积损失，并遵循领先专家的建议，而这需要 $\Omega(n)$ 内存，因此研究带有内存约束的问题尤其有意义。

受这一认识不足的启发，近期有人在 [SWXZ22], 中提出了具有内存约束的专家在线学习问题，并研究了专家的损失构成独立同分布序列，或最佳专家的损失有界的情形。随后， [PZ23] 的工作去除了这些假设，并获得了一种算法，在 $\tilde{O}(T^{\frac{2}{2+\delta}})$[1] 遗憾

[1] $\tilde{O}$ 在 T 中隐藏了多对数因子。为简单起见，在本引言部分的遗憾界中，我们忽略额外的 $\text{poly}(n)$ 开销因子。

使用 n^δ 内存的一般情形；更详细的比较请参见第3节。直观而言， [PZ23] 的结果表明，只需在任意固定日期跟踪一个（不断演化的） n^δ 专家集合，就足以实现次线性遗憾。然而，当空间预算接近线性空间时，他们的工作留下了一个自然问题。在这一范围内，自然可以猜测遗憾 $\tilde{O}(\sqrt{T})$是可以实现的，即标准乘法权重更新（MWU）算法所达到的遗憾界（以及许多其他算法 [LW89、KV03、AHK12、Haz16]）；该算法使用$O(n)$空间。然而， [PZ23] 的算法在这一范围内只能达到 $\tilde{O}(T^{2/3})$遗憾。

我们通过给出一种使用 n^δ 内存的算法，实现约为 $\tilde{O}(T^{\frac{1}{1+\delta}})$的遗憾，从而在线性内存情形下获得 $\tilde{O}(\sqrt{T})$的遗憾，弥合了这一认知差距。

1.1 我们的结果

我们简要概述问题设定，以陈述我们的结果；完整细节留待第4节介绍。在在线专家问题中，在持续 T 天的一系列日子里，我们每天都要从 n 位专家中选择一位。于 t日选择专家 $i_t \in [n]$ 后，我们会收到一个损失向量 $\ell_t \in [0,1]^n$ ，并承担损失 $\ell_t(i_t)$。本文假设每位专家的损失序列均由非自适应对手选定。我们的目标是最小化在线学习中的标准遗憾，即将算法所做预测的总损失与事后看来最佳专家的总损失进行比较：遗憾 $= \sum_{t\in[T]} \ell_t(i_t) - \min_{i\in[n]} \sum_{t\in[T]} \ell_t(i)$。我们的主要结果如下。

定理1。 对于任意 $\delta \in (0, 1)$，存在一种针对在线专家问题的算法，在$T = n^{O(1)}$ 天内使用空间 $\tilde{O}(n^\delta)$，并以概率 $1-1/\mathrm{poly}(T)$达到遗憾 $\tilde{O}(n^2 T^{\frac{1}{1+\delta}})$。

相比之下， [PZ23] 给出了一种算法，在相当的空间预算下实现了遗憾 $\tilde{O}(n^2 T^{\frac{2}{2+\delta}})$。具体而言，对于所有 δ 的取值，我们在遗憾界中获得了更小的指数；见图2。此外，在接近线性空间的情形下，即 $\delta \to 1$，我们实现了遗憾 $\tilde{O}_n(\sqrt{T})$，达到了传统在线算法（例如乘法权重更新）的保证；这些传统算法会显式跟踪全部 n 个专家的性能历史。相比之下，在这一情形下， [PZ23] 实现了遗憾 $\tilde{O}_n(T^{2/3})$。

2 技术概述

为简洁起见，在本概述中我们忽略对数因子。我们首先对 [PZ23] 中的算法作高层次概述，然后介绍本项工作中的新思路，以实现改进后的上界。 [PZ23] 中的上界由两部分组成：基线算法和自举过程。对于 $\delta \in (0,1)$，基线算法在空间参数 $m = n^\delta$ 下，以 T 和 n 为依据，实现有界遗憾。在根据 n 和 m 对 T 作某种特定 设定时，该基线算法可实现 $T^{\frac{2}{2+\delta}}$ 的遗憾。自举过程则将这一相同的上界扩展到更大的 T 值。我们的主要贡献是开发出一种改进的基线算法，它在 T、 n 和 m 方面实现了更优的遗憾权衡。此外，我们简化并缩短了自举过程的分析，并给出一个引理：该引理本质上将任意两个算法 $\mathcal{A}_1$ 和 $\mathcal{A}_2$ 合并；这两个算法分别在时间 T_1 和 T_2 上具有遗憾 R_1 和 R_2 ，合并后得到的算法在 T_1T_2 天内具有遗憾 R_1R_2 。

2.1 [PZ23] 的基线算法

[PZ23] 中基线算法的核心思路是将时间 T 划分为一系列连续的区块，每个区块大小为 $T_0 < T$。每个区块中都会维护一个包含 $m + \sqrt{m}$ 位专家的池。算法在任意一天的预测，都是在当前区块中的专家上运行乘法权重更新（MWU）所得的输出。每经过 T_0 个时间步进入下一个区块时，权重都会重置。区块最初设为从 m 位专家中随机抽取的样本。对于池中的每位专家 e，算法都会记录其到达时间，以及自池中其他专家 e' 到达以来的平均损失。这需要的内存与池中专家的不同到达时间数量的平方成正比。

该算法还会保留那些已在池中停留较长时间且表现相对较好的专家。设 $t(e)$ 为专家 e 进入池中的时间，并令$\epsilon = 1/\sqrt{m}$。在每个区块结束时，如果存在一对专家 $e,\ e'$，满足 $t(e') < t(e)$，且从 $t(e)$ 起，e' 的平均损失至多为 e 的 $(1+\epsilon)$，我们就称 e' “支配” e，并将 e 从池中移除。移除所有被支配的专家后，算法会随机抽取 m 位新专家并加入池中。

内存 算法并未明确限制池中任一时刻的专家数量，但由于支配规则，该数量受到 m 位新抽取专家和 $\sqrt{m}$ 位较早加入专家的限制。我们在 [PZ23] 的第 3.2 节中通过势函数论证对此进行了形式化证明。我们的第一项改进是一种支配策略：在内存用量相同的情况下，它能使存储的较早加入专家数量实现二次 级改进，而且分析起来简单得多。我们将在第 2.2 节中介绍它。

遗憾 [PZ23] 基线算法所达到的遗憾界由以下公式给出

$$\frac{T}{\sqrt{m}} + \frac{T}{\sqrt{T_0}} + \frac{nT_0}{m}. \tag{1}$$

为了界定基线算法的遗憾，[PZ23] 的先前工作引入了驻留块和驱逐区块的深刻概念（尽管使用了不同的术语）。在每个区块开始时进行采样之前，我们提出如下假设性问题：给定当前的专家池，如果此时采样最佳专家 e^*，它会一直保留到时间结束吗？请注意，e^* 不会一直保留到时间结束的唯一原因，是专家池中已经存在某个专家，并且该专家将在某个时刻胜过 e^*。如果答案是“是”，则该区块为驻留块。否则，我们称其为驱逐区块。

对于任何驻留块，如果我们确实采样到了最佳专家，它将永远保留，此时我们所付出的唯一遗憾将是由于在每个桶中运行乘法权重更新而产生的遗憾。共有 T/T_0个桶，每个桶产生$\sqrt{T_0}$ 的遗憾，共同构成公式（1）的第二项。如果我们没有采样到最佳专家，驻留块的代价最高可达 T_0 的遗憾，因为池中的所有专家表现都可能远差于最佳专家。然而，由于我们有 m 次机会从 n 个专家中采样最佳专家，因此我们可能只会遇到 n/m 个未采样到最佳专家的驻留块。这构成了公式（1）的第三项。

对于一个驱逐块，即使我们采样到了最佳专家，也会将其驱逐；但正因如此，我们知道，驱逐 e^* 的专家 e' 必须在某个区间内与 e^* 具有竞争力。具体而言，设 t' 为 e' 占据主导并驱逐 e^* 的时间。要使这种情况发生，e'

必须一直留在池中直到 t'，并且在截至 t' 的区间内，其平均损失至多比最佳专家差一个 $(1+\epsilon)$ $=(1+1/\sqrt{m})$ 倍。在这些从驱逐区块开始的区间内，我们承担的平均遗憾为 $1/\sqrt{m}+1/\sqrt{T_0}$（第二项来自使用乘法权重更新的开销），这部分计入公式（1）的第一项和第二项。

通过设置 $T_0=m$ 和 $T=n\sqrt{m}=n^{1+\delta/2}$，即可得到关于 T 的最终遗憾界。因此，总遗憾为 $n=T^{1+\delta/2}=T^{2+\delta}$。

2.2 改进的支配策略

我们的首要贡献是提出了一种改进且更简单的支配定义。对于池中的一对专家e,e'，如果 $t(e')<t(e)$，且自 $t(e')$ 起 e' 的损失至多比自 $t(e)$ 起 e 的损失大 $(1+\epsilon)$ 倍，我们就称 e' 支配 e。与先前工作的关键区别在于，我们用来比较两位专家的损失，仅计算他们各自进入池之后的损失，而不是从这一对中较晚到达的专家进入池时起计算的损失。请注意，因此我们只需存储关于池中专家数量的线性（而非二次）信息。因此，我们可以设定 $\epsilon=1/m$，而不是$1/\sqrt{m}$。这一改变也大大简化了内存分析，我们将在下文概述。

内存 在每个区块结束时，专家会被驱逐，并采样 m 个新专家。我们将论证，经过此过程后，算法跟踪的专家最多为 $2m$个。考虑驱逐步骤后仍留在池中的一对专家 $e,\ e'$。不失一般性，假设 e'比 e先到达。那么，自 e 到达以来的损失必须小于自 e'到达以来损失的 $1/(1+\epsilon)$ 倍。由于这对所有专家对都成立，且损失必须属于 $[0,T]$，因此最多有 $\log_{1/(1+\epsilon)}(T)$ 个 $=O(1/\epsilon)$。适当设置 ϵ后，这意味着驱逐后最多有 m 个专家，因此算法在任意时刻跟踪的专家最多为 $2m$ 个。

遗憾 采用这一新支配规则的基线算法，其遗憾为

$$\frac{T}{m}+\frac{T}{\sqrt{T_0}}+\frac{nT_0}{m}. \tag{2}$$

我们借助 [PZ23] 中保留和驱逐区块的核心思想来计算遗憾。公式（1）与我们在公式（2）中的遗憾界之间的关键区别在于，我们构造中用于支配的 ϵ 小了一个二次量级，因此得到的是 T/m，而不是 $T/\sqrt{m}$ 项。

通过设置 $T_0=m^2$ 和 $T=nm^2=n^{1+2\delta}$，可以得到关于 T 的最终遗憾界。此时，总遗憾为 $nm=n^{1+\delta}=T^{\frac{1+\delta}{1+2\delta}}$。这严格优于 [PZ23] 给出的界（见图 2 中的第一条灰色曲线）；然而，在使用线性内存时，仍会得到不理想的 $T^{2/3}$ 遗憾界。

2.3 改进的分层基线

上述算法的一个根本瓶颈在于：采样新专家与长时间保留任何已采样专家之间存在权衡。前者对应公式（2）中遗憾的 nT_0/m 项，后者则对应公式（2）中的 $T/\sqrt{T_0}$ 项；长时间保留已采样的专家，才能充分获益于乘法权重更新。nT_0/m 项促使我们将 T_0 设为较小的值

从而实质上提高采样频率，增加在上述某个驻留块中采样到最佳专家的可能性。另一方面，后一个项$T/\sqrt{T_0}$则促使我们将任何专家保留更长时间，即令 T_0 取更大的值。上述算法必须在这两个相互冲突的目标之间取得平衡。一个理想的目标是通过将 T_0 设为较小的值来实现“鱼和熊掌兼得”，从而“探索”许多专家；同时，将 T_0 设为较小的值，并“利用”我们当前正在跟踪的专家。我们的改进受到尝试实现这种假想行动方案的启发。

为了进一步改进算法，我们使用由不同大小的 k 区块构成的层级结构：我们将时间 T 划分为大小为 T_{k-1} 的区块，再将这些区块进一步划分为大小为 T_{k-2} 的区块，依此类推，直到最小的区块大小为 T_0。通过在不同层级之间传递表现良好的专家，并将每个层级视为一个元专家，我们可以借助更短的层级以更高的频率有效地抽样专家，同时保留跨越更长时间范围的层级所带来的乘法权重更新优势，从而实现“鱼和熊掌兼得”。

在本概述中，我们详细描述 $k = 2$ 情形，以展示使用多个区块为何有帮助（我们的算法将使用 $k \approx \log\log T$）。

在公式（2）中，区块大小 T_0 出现在第二项和第三项中。较大的区块大小会降低第二项中的遗憾，因为我们在重置之间运行乘法权重更新的时间更长，而乘法权重更新的平均遗憾与其运行时间跨度的平方根成反比。另一方面，较小的区块大小会降低我们在驻留块中付出的遗憾，因为在驻留块中我们不会抽样最佳专家。换句话说，小块意味着我们会更频繁地进行抽样。使用两种不同区块大小的目标，是通过将遗憾计入较大的驱逐块和较小的驻留块来实现鱼和熊掌兼得。

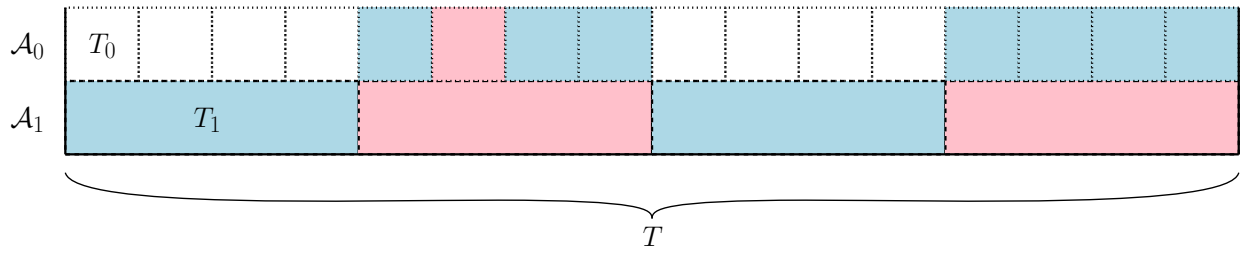

图1：该图展示了仅包含两个层级时的层级基线方案。蓝色区块表示驱逐区块，红色区块表示驻留块。

设 $T_0 < T_1 < T$ 为块大小，其中 T_0 整除 T_1，且 T_1 整除 T。请参见图1。设 $\mathcal{A}_0$ 和 $\mathcal{A}_1$ 分别为上一节中使用 T_0 和 T_1 的单层基线算法的输出。层级基线将是对 $\mathcal{A}_0$ 和 $\mathcal{A}_1$ 运行的乘法权重更新算法的输出，并且每经过 T_1 个区块重置一次。对于每个 T_1 区块，我们可以将遗憾界定为 $\mathcal{A}_0$ 和 $\mathcal{A}_1$ 中较优者，再加上乘法权重更新算法产生的额外 $\sqrt{T_1}$ 项。对于每个 T_1 驱逐块，我们可以按照公式（2）中的前两项来界定遗憾，但使用块大小 T_1。对于每个 T_1 驻留块，我们将考虑 $\mathcal{A}_0$ 的遗憾。

每个 T_0 驱逐块都会对遗憾贡献两项：一项是驱逐专家相对于最佳专家的损失，另一项是运行乘法权重更新所产生的损失。第一项对总遗憾的贡献至多是加性项 T/m。第二项则会为我们计数的每个驱逐块贡献 $\sqrt{T_0}$。然而，由于我们只在 T_1 驻留块中计数 T_0 驱逐块，并且在 n/m T_1 驻留块之后我们将

采样并保留最佳专家，因此这总共贡献的遗憾为 $\frac{n}{m}\left(\frac{T_1}{T_0}\right)\sqrt{T_0}$。

最后，我们考虑 T_1 驻留块中的 T_0 驻留块。总共可能有 $\frac{n}{m}\ T_0$ 个驻留块，之后我们才会抽样到最佳专家，而该专家会永远保留。然而，即使我们在 T_0 区块中抽样到了最佳专家，每个区块仍必须支付 $\sqrt{T_0}$ 乘法权重更新成本。因此，我们做出一项关键的算法性改动：在每个 T_1 区块结束时，我们在 $\mathcal{A}_1$ 中复制 $\mathcal{A}_0$驱逐后每个专家的副本，并确保这些复制出的专家在 $\mathcal{A}_0$ 下一个 T_1 区块之前不会被驱逐。这会使我们的内存总量增加到至多 k^3m ，因为每个区块在驱逐后最多保留 km 个专家，并且最多从每个更小的区块接收 km 个专家。此外，这保证了如果我们曾将最佳专家抽样到一个 T_0 驻留块中，那么它就会出现在之后的每个 T_1 区块中。因此， T_0 驻留块中的 T_1驻留块对遗憾的总贡献为 $\frac{n}{m}T_0+\frac{T_1}{T_0}$。总之，我们可以将这个双层算法的遗憾界定为

$$\frac{T}{m}+\frac{T}{\sqrt{T_1}}+\frac{n}{m}\left(\frac{T_1}{\sqrt{T_0}}+T_0\right). \tag{3}$$

设置 $T_0=m^{4/3}$、 $T_1=m^2$ 和 $T=nm^{4/3}=n^{1+4\delta/3}$ ，可得遗憾 $nm^{1/3}=n^{1+\delta/3}=T^{\frac{1+\delta/3}{1+4\delta/3}}=T^{\frac{3+\delta}{3+4\delta}}$。当 δ 从 0 变为 1 时，这个界在遗憾 T 和 $T^{4/7}$ 之间插值，整体上优于单层算法，尤其是在内存线性情形下（见图2中的第二条灰色曲线）。第5节中的完整分层基线算法将这一分层思想扩展到 $k=O(\log\log T)$，从而在 $T=nm$ 时实现遗憾 $\tilde{O}(T^{\frac{1}{1+\delta}})$。

我们强调，仅仅在每个 T_1 区块之上递归运行 [PZ23] 的算法（见图1），并不足以实现 $T^{\frac{1}{1+\delta}}$ 遗憾。关键在于将较高层级的专家向下传递到较低层级，以便用 $\frac{nT_0}{m}$ 这一较小的遗憾项，替代 $\mathcal{A}_1$ 中由驻留块导致的“较大”遗憾项。然而，我们不能只将较高层级 $\mathcal{A}_0$ 的专家传递到 $\mathcal{A}_1$，因为这会引发一个微妙的问题：在 $\mathcal{A}_0$ 中表现良好的专家与 e^* 具有竞争力的时间区间，可能与 $\mathcal{A}_1$ 中的专家和 e^* 具有竞争力的时间区间重叠。这就造成了两难：我们无法保证，在涵盖两个专家分别与 e^* 具有竞争力的时间区间的并集 的较大时间区间内，存在一个具有竞争力的专家。为说明这一点，考虑以下情形 $T=5$ 中的玩具示例：假设专家 e_1 的损失序列为 $[1,0,0,0,1]$， e_2 的损失序列为 $[0,0,0,1,1]$,，e^* 的损失序列为 $[0,0,1,0,0]$, ，而我们在这 5 天里只能选择一位专家跟随。那么， e_1 在第 2 至第 5 天与 e^* 具有竞争力（即这几天的总损失与 e^* 相同），而 e_2 在第 1 至第 4 天与 e^* 具有竞争力，但没有任何专家在整个时间区间内与 e^* 具有竞争力。

我们通过强制规定来避免这个问题：在一个小块中，传递到较大块的专家在这些较大块结束之前不能被驱逐。这意味着，如果这个小块是一个驱逐块，那么驱逐专家必须在一段时间内与 e^* 具有竞争力，而这段时间由若干个小块和一个或多个较大块组成。

2.4 面向更大 T 的自举

上述结果适用于 T 的某个特定取值。事实上， T 是输入的一部分，算法无法对其加以控制。[PZ23]的作者提出了一种递归宽度缩减技术，通过引导其基线算法，使其适用于更大的 T 值。宽度缩减是指该过程使用基线算法来缩小损失范围。我们给出了这一结果的一个更一般版本（如下所述），并在第6节中提供了更简洁的分析。

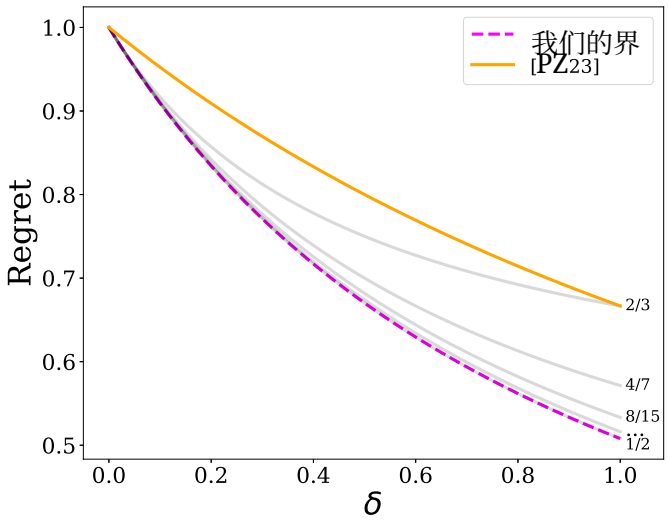

图2： x轴表示空间预算中 n 的指数， y 轴表示遗憾中 T 的指数。橙色曲线显示了 $f(\delta) = 2/(2+\delta)$的图像，代表 [PZ23]的主要结果。灰色曲线表示我们使用由不同大小的 k 个区块组成的层级结构所得到的遗憾界，分别为 $k = 1$、2、3、.....。如 k 增加所示， k 的每种设置都会改善内存—遗憾权衡；在近线性空间范围内，我们得到的指数依次为 $2/3$、 $4/7$、 $8/15$、.....。我们最终选择 $k = O(\log\log T)$，从而得到定理1中 T 的 $1/(1+\delta)$指数，该结果由洋红色虚线曲线表示。

引理 1。 假设有两个用于专家问题的算法 *ALG1* 和 *ALG2*，面对n 个专家，且每日损失范围为 1（即最大损失与最小损失之差），其参数如下：*ALG1* 运行 T_1 天，遗憾至多为 R_1；*ALG2* 运行 T_2天，遗憾至多为 R_2；两者的上述保证均以概率 $1-\delta$ 成立。此外，假设这两个算法的空间复杂度均为 m。那么，存在一个运行 $T_1 \cdot T_2$ 天的算法 *ALG'*，其遗憾至多为 $R_1 \cdot R_2 + O\left(T_1 \cdot \sqrt{T_2 \ln(n/\delta)}\right)$，且该保证以概率 $1-\delta\,(nT_1+1)$ 成立。此外，*ALG'* 的空间复杂度至多为 $O(m)$。

3 相关工作

与 [SWXZ22] 和 [PZ23] 这些研究与我们的工作最为相关。 [SWXZ22]开启了对次线性空间下专家在线学习问题的研究。他们证明，当每位专家的损失序列独立同分布时，可以用 n^{δ} 空间实现 $O(\sqrt{T \cdot n^{(1-\delta)/2}})$遗憾。他们的算法设计与独立同分布假设密切相关。他们的论文还给出了该问题的匹配下界（这一下界也适用于我们的对抗设置）。值得注意的是， [Sha14]也证明了有限内存下独立同分布设定的一个定性相似下界。他们还研究了非自适应对手的（标准）最坏情况模型，并证明，在最佳专家具有次线性 损失这一强假设下，次线性遗憾是可能实现的。

[PZ23] 证明了，实现次线性空间下的次线性遗憾，既不需要独立同分布假设，也不需要最佳专家在所有 T 天内均达到次线性损失。具体而言，他们证明，在第4节定义的标准非自适应对手模型下，可以用 n^{δ}空间以高概率实现$\tilde{O}(n^2T^{\frac{2}{2+\delta}})$遗憾。我们的工作也采用了 [PZ23] 研究的这一相同的一般模型，并以 n^{δ}空间实现了 $\tilde{O}(n^2T^{\frac{1}{1+\delta}})$遗憾，改进了 [PZ23] 针对 $\delta > 0$给出的遗憾界。特别是， $\delta \to 1$ （忽略 poly(n) 因子），我们的遗憾趋近于 $\tilde{O}_n(\sqrt{T})$，而 [PZ23] 的遗憾则趋近于 $\tilde{O}_n(T^{2/3})$。前者的缩放关系是恰当的，因为已知在不考虑空间限制的标准专家在线学习问题中， $\tilde{\Theta}(\sqrt{T})$遗憾既可实现，也是必要的 [AHK12]。 [PZ23] 还证明，在更强的自适应对手模型下（输入序列可以依赖于算法迄今使用的随机性），次线性空间下不可能实现次线性遗憾。这也进一步说明了研究非自适应对手设定的动机。

请注意， [SWXZ22] 给出的下界适用于独立同分布设定，因此也自动适用于非自适应对手的一般设定。然而，正如 $\delta \to 0$所示， [SWXZ22] 的下界意味着遗憾至少为 $\Omega(\sqrt{Tn})$。如果 $n \ll T$，这意味着在对数空间下，仍可能实现次线性（以 T 遗憾计）。在这个极小空间范围内，要么加强下界，证明遗憾至少为 $\Omega(T)$，要么加强我们在极小内存范围内的上界；这两者都是未来研究的有趣方向。

专家问题 专家问题和乘法权重更新算法都相当通用，并已应用于许多算法与优化问题，包括提升、图算法、投资组合优化、线性规划、统计估计和学习理论等 [Bro51, LW89, FS95, CO96, OC98, CBL06, CKM+11, GH16, KM17, HLZ20]。我们特别推荐综述 [AHK12]，以了解更多信息。在有界内存范围内探索这些应用，是一个值得关注的开放方向。

内存受限学习受限内存学习本身就是一个内容丰富的研究领域，已有大量关于凸优化、核方法、统计查询和通用机器学习算法的研究 [WS00, RR07, MCJ13, Sha14, SD15, SVW16, KRT17, Raz17, DS18, DKS19, SSV19, GRT19, BIK+22]。这些研究中的许多工作侧重于下界，或并非针对在线模型。相关领域的研究成果十分丰富，更多信息请参阅所引用的文献。

在线多臂赌博机设置中的内存约束也已有研究，但这些工作大多聚焦于随机设置，因此与我们关注的最坏情况视角不同。读者可参阅 [LSPY18, CK20, AW20, JHTX21, MPK21] 及其中所引用的文献。

4 预备知识

我们考虑包含 n 位专家、持续 T 天的专家学习问题。对于每一天 $t \in [T]$，每位专家 e 都会做出一个预测 $x_e^t \in [0,1]$。随后，我们会收到关于每位专家 e的损失 $\ell_t(e)$ 的反馈。我们用 $\ell_t \in [0,1]^n$ 表示编码所有专家损失的损失向量。我们采用与 [PZ23],完全相同的模型，下面对此进行概述。

查询模型 在我们所采用的空间受限模型中，无法在每个时间步显式存储每位专家的损失和预测。为对此进行形式化，我们首先说明如何获得专家的损失。在每个时间步 t之前，我们可以选择 E_t 位专家，并观察他们的预测。然后，我们选定一位固定专家 $e \in E_t$ ，并遵循其预测。接着， E_t 中的所有专家的损失都会被揭示，因此我们的算法在第 t 天的损失为 $\ell_t(e)$。

内存/空间模型 我们的内存模型可以用标准的流式计算模型来理解。一个内存字包含 $O(\log(nT)$ 位，并且在每个时间步 t，我们的内存状态由 M_t 个不同的内存字组成。其中一些内存字用于表示我们观察其预测和损失的专家 E_t 。观察损失后，我们更新内存状态，使其使用 M_{t+1} 个内存字。我们称一个算法的空间复杂度为 M ，如果$M_t \le M$ 对所有 $t \in [T]$均成立。

我们的算法具有随机性，并假设可以通过预言机访问随机比特，这与 [PZ23]类似。通过流式计算文献中的标准工具，可以利用仅需额外多对数空间的伪随机生成器轻松移除这一假设，其中 n 和 T [Nis92, Ind06]。因此，我们忽略生成随机比特的空间复杂度。

对手模型 我们的工作基于非自适应对手，这是流式算法中的一个标准假设。对于所有 $t \in [T]$ ，损失向量 ℓ_t 均由对手构造。对手可以使用随机性，但其随机比特独立于我们的算法所使用的任何随机比特。或者，对手首先预先构造损失向量，之后我们的算法通过上述查询模型与对手交互，并使用自身独立的随机性。

其他记号 我们使用 O 记号来隐藏 n 和 T中的对数项。与先前的研究[SWXZ22, PZ23], 一样，为简便起见，我们假设 n 和 T 呈多项式关系。我们还指出，通过应用标准的倍增技巧，我们的结果很容易扩展到 T 事先未知的情形；参见注释1。

我们还陈述标准的乘法权重更新（MWU）算法，这是我们分析中的关键工具，并给出其保证 [AHK12]。

引理2 （MWU保证， [AHK12, PZ23]）。 假设 $n, T, \eta > 0$ ，并令 $\ell_t \in [0,1]^n (t \in [T])$ 表示所有专家在第 t 天的损失。乘法权重更新算法满足

$$\sum_{t=1}^{T} \langle p_t, \ell_t \rangle - \min_{i^* \in [n]} \ell_t(i^*) \le \frac{\log n}{\eta} + \eta T,$$

且其成立的概率至少为 $1-\delta$,

$$\sum_{t=1}^{T} \ell_t(i_t) - \min_{i^* \in [n]} \ell_t(i^*) \le \frac{\log n}{\eta} + \eta T + O\left(\sqrt{T \log(n/\delta)}\right).$$

取 $\eta = \sqrt{\log n / T}$ 乘法权重更新算法的总遗憾为 $O\left(\sqrt{T \log(nT)}\right)$，其成立概率至少为 $1 - 1/\mathrm{poly}(T)$ ，标准实现需要 $O(n)$ 字的内存。

5 算法

算法 1 获取预测**输入：** 桶 $\mathbf{B} = B_0, \ldots, B_{k-1}$，跨桶权重 **w 输出：** 包含 k 个预测（专家）的数组 $\hat{\mathbf{y}}$将 $\hat{\mathbf{y}}$ 初始化为大小为 k 的数组**对于** $i \in \{0, \ldots, k-1\}$ **执行**从 B_i 中按权重成比例地选择一位专家 e_i **如果** $i = 0$ **则** $\hat{\mathbf{y}}_0 \leftarrow e_i$**否则**$(w, w')$ $\leftarrow \mathbf{w}_{i-1}$ 分别是当前桶和前一个桶的权重$\hat{\mathbf{y}}_i \leftarrow e_i$，概率为 $\frac{w}{w+w'}$否则，$\hat{\mathbf{y}}_i \leftarrow \hat{\mathbf{y}}_{i-1}$**结束如果 结束循环 返回** $\hat{\mathbf{y}}$

——

算法 2 更新权重

输入：桶 B $= B_0$，…，B_{k-1}，跨桶权重 w，算法预测 $\hat{\mathbf{y}}$，损失 $\boldsymbol{\ell}$，块大小 $\{T_i\}_{i=0}^{k}$输出：$i \in \{0, \ldots, k-1\}$ 的更新后的桶 B 和权重 w执行$\eta_i^{(1)} \leftarrow \sqrt{\log(|B_i|)/T_i}\eta_i^{(2)} \leftarrow \sqrt{2/T_i}$对每位专家 $e \in B_i$ 执行▷ 更新桶内的内部权重 将 e 在 B_i 中的损失增加 $\ell_t(e)$ 对 B_i 中相应权重执行乘法权重更新，速率为 $\eta_i^{(1)}$，损失为 $\ell_t(e)$ 结束循环 如果 $i > 0$，则▷ 更新桶之间的外部权重 对 $\mathbf{w}_{i-1}$ 中的大权重执行乘法权重更新，速率为 $\eta_{i-1}^{(2)}$，损失为 $\ell_t(\hat{\mathbf{y}}_i)$ 结束如果 对 $\mathbf{w}_i$ 中的小权重执行乘法权重更新，速率为 $\eta_i^{(2)}$，损失为 $\ell_t(\hat{\mathbf{y}}_i)$ 结束循环 返回 B，w

5.1 算法描述

[PZ23] 的“基线”算法将总时间 T 划分为更短的周期（桶），并跟踪每个桶内少数专家的进展。算法通过对每个桶内的专家运行乘法权重更新（在桶之间重置）来生成预测。在

算法 3 更新桶

输入： 桶 $\mathbf{B} = B_0, \ldots, B_{k-1}$、跨桶权重 $\mathbf{w}$、块大小 $\{T_i\}_{i=0}^{k}$、空间参数 m、时间 t

输出： 更新后的桶 $\mathbf{B}$ 和权重$\mathbf{w}$

$\epsilon \leftarrow \log(T)/m$

$k' \leftarrow$ 最大值 $\{i \in \{0, \ldots, k-1\} : t \equiv 0 \text{ (模 } T_i)\}$ ▷ 要更新的桶数量

$\tau \leftarrow$ 最大值$\{t' \leq t : t \equiv 0 \text{ (模 } T_{k'+1})\}$ ▷ 最近一次更新时间为 $B_{k'+1}$

</style}对于 $i \in \{0, \ldots, k'\}$ **执行** ▷ 淘汰专家，重置权重

从 B_i 中移除任意到达时间至少为 $\tau + 0.5$ 的专家 e，如果 $\exists e' \in B_i :$ e' ϵ-支配 e（根据定义**1**）

将 B_i 中所有剩余专家的权重设为 **1**

将 $\mathbf{w}_i$ 中的权重设为 **1**

结束循环

对于 $i \in \{k', k'-1, \ldots, 1\}$ **执行** ▷ 将专家从较小的桶转发到较大的桶

在 B_i 中，为 $B_0, \ldots, B_{i-1}$ 桶中的所有专家创建副本，并为其设置唯一的到达时间在 $[t, t+0.5)$ 中，损失为 **0**，权重为 **1**

结束循环

对于 $i \in \{0, \ldots, k'\}$**，执行** ▷ 采样新专家

采样 m 个专家（可重复采样），并将其添加到 B_i 中，为其设置唯一的到达时间 $[t+0.5, t+1)$，损失为 **0**，权重为 **1**

结束循环

返回 B $, \mathbf{w}$

算法4 HierarchicalBaseline 输入：块大小 $T_0 < T_1 < \ldots < T_k = T$**，其中** $T_i \equiv 0$ **（模** T_{i-1}**），适用于** $i \in [k]$**；专家数量**n**；空间参数** m**1：创建** k **个桶 B** $= B_0$**，.....，** B_{k-1}**，每个桶将存储形式为（专家、到达时间、损失、权重）的元组 [2]2：创建数组 w，其中包含** $k-1$ **对权重（大权重、小权重），初始值均为 1 3：B** $\leftarrow$ **UpdateBuckets(B, w** $\{T_i\}_{i=0}^{k}$**,** m**, 0)** ▷ **用样本初始化桶 4：对于** $\{1, \ldots, T\}$ **中的时间** t**，执行 5：**$\hat{\mathbf{y}}$ $\leftarrow$ **GetPredictions(B, w) 6：选择专家** $\hat{\mathbf{y}}_{k-1}$ **进行预测 7：观察损失** $\ell \leftarrow \{\ell_t(e) : e \in B\}$ ▷ **查询损失向量 8：B, w** $\leftarrow$ **UpdateWeights(B, w,** $\hat{\mathbf{y}}$ ℓ**,** $\{T\}_{i=0}^{k}$**)** ▷ **更新权重 9：如果** $t \equiv 0$ **（模** T_0**）则 10：B** $\leftarrow$ **UpdateBuckets(B, w** $\{T_i\}_{i=0}^{k}$**,** m, t**)** ▷ **更新桶 11：结束如果 12：结束循环**

在每个桶结束时，根据某种特定定义表现不佳的专家会被驱逐，同时会抽取一些新的专家（有关其算法的更详细描述，请参见第 2 节）。我们的算法采用了这一底层框架，但使用了新的驱逐规则和

不同大小的桶层级。

主要算法是算法 4，其关键子程序见算法 1、2 和 3。该算法的输入是一组 $k+1$ 值 $T_0 < T_1 < \ldots < T_k = T$，满足 $T_i \equiv 0$（模 T_{i-1}），用于指定 $i \in [k]$ 层级中 k 桶的块大小。此外，算法还接收专家数量 n和空间参数 m （算法最终使用的空间为 $O(k^3 m)$）。

每个桶都会维护一组 $O(k^3 m)$ 专家（见引理 3），以及每位专家的到达时间、损失和权重。到达时间是专家到达该桶时的时间 $[T]$，损失是该专家从到达时起所产生的损失总和，而权重则对应于一个在桶内专家上运行的乘法权重更新子程序。从高层次来看，在每个时间步，层级中的每个桶都会输出一个待跟随的预测专家。我们在各层级之间递归地使用乘法权重更新（使用权重 $\mathbf{w}$），以确定要输出哪位预测专家作为最终预测。接着，我们观察所有桶中所有专家的真实损失，并更新每个桶内乘法权重更新中的乘法权重，以及桶之间的乘法权重。最后，如果到达某个桶的末尾，我们就更新该桶中的专家：淘汰部分专家、重置权重、将小桶中的专家转发到大桶，并采样一些新专家。下面将详细介绍预测、更新权重和更新桶的过程。

预测（算法 1）预测主要由两类权重构成。每个桶都会为其专家池中的专家维护一组“内部”权重。此外，算法还会为每对相邻的桶维护一对“外部”权重。每个桶根据内部权重，按比例从其专家池中选取一位专家 e_i。我们将最小桶中的预测记为 $y_0 = e_0$。然后，从较小的桶到较大的桶依次进行，在第 i 步，按相应的外部权重比例，从 e_i 的预测和 y_{i-1} 的预测中选取预测 y_i。

更新权重（算法 2）对于每个桶，我们观测该桶中每位专家的损失。我们根据乘法权重更新规则更新相应的内部权重和外部权重。

更新桶（算法 3）如果到达满足 $t \equiv 0 \pmod{T_i}$ 的时间，我们就更新桶 B_i 中的条目。令 k' 为我们更新的最大桶的索引，并令 τ 为桶 $B_{k'+1}$ 上次更新的时间。更新桶包含四个步骤：驱逐、重置权重、转发和采样。首先，对于每个待更新的桶，我们根据以下支配定义，驱逐池中所有自时间 $\tau + 0.5$ 以来加入的专家。

定义1 (ϵ-支配)。 算法跟踪的每位专家 e 都有一个到达时间，以及从该时间起的损失。给定参数 ϵ，若专家 e' ϵ支配 e，则 e' 在 e 之前到达，且 e' 的损失至多为 e损失的 $(1+\epsilon)$ 倍。

对于待驱逐的专家，如果池中的任何其他专家使用$\epsilon = \log(T)/m$支配它们，就将它们驱逐。然后，将池中所有剩余专家的内部权重，以及层级 $i-1$ 和 i 之间的外部权重重置为 1。

接下来，对于每个需要更新的桶，我们复制从较小桶中驱逐后剩余的所有专家。我们复制这些专家，但不复制他们的到达时间或

损失。相反，我们将他们的到达时间设为当前时间，并将他们的损失设为 0。最后，我们采样 m 个新专家。无论是转发还是采样专家，我们都会为每位专家分配一个唯一的到达时间，记作 $[t, t+1)$，以确保支配规则定义明确。

5.2 改进的基线算法的分析

本节将分析改进的基线算法算法 4，并证明其空间使用量和遗憾的上界。下面的引理给出了空间使用量的上界。

引理 3。 算法 *4* 使用 $O(k^3 m)$ 个词的空间。

证明。 算法所需的空间主要取决于桶**B**中的条目数量。对于桶中的每个条目，我们存储固定数量的词；总体而言，在每个时间步查询的专家数量至多等于**B**中的条目数量。因此，在任意时间步存储的词数 t 可以由**B**中的条目数量乘以某个固定常数作为上界。

考虑任意时间步 t 以及满足 $t \equiv 0$ (模 T_i)的任意桶 B_i，即， B_i 将在算法 3 中更新。回顾一下， τ 是未被更新的最小桶的起始时间。我们将论证，在第 8 行执行驱逐步骤后， B_i 中至多包含 $O(m)$ 个自时间 $\tau + 0.5$以来到达的专家。由于所有专家的到达时间各不相同，并且根据ϵ-支配的定义1，驱逐后仍然保留的每个专家 e 的损失必须小于在 e之前到达的任一仍保留专家损失的 $1/(1+\epsilon)$。由于损失位于 $[0,1]$, 中，因此从待驱逐的专家池中（即自 $\tau + 0.5$以来到达的专家中）至多有 $\log_{1+\epsilon}(T) = O(\log(T)/\epsilon) = O(m)$ 个专家仍然保留。

根据上述论证，对于每个 $j > i$， B_i 最多可包含 $O(m)$ 个在上一轮驱逐中存活下来的专家；在该轮中， $k' = j$ （B_j 是被更新的最大桶）。其他专家都将在当前步骤被考虑驱逐，因此驱逐后 B_i 中的专家总数至多为 $O((k-i)m)$。

第 11 行转发后， B_i 收到了来自所有 B_j 、针对 $j < i$ 而在驱逐中存活下来的专家。此步骤后， B_i 中的专家总数为

$$O\left(\sum_{j=0}^{i}(k-j)m\right) = O(kmi).$$

第 13 行的采样增加了 m 个额外条目。因此，所有桶中的条目总数为 re

$$O\left(\sum_{i=0}^{k-1} kmi\right) = O(k^3 m). \qquad \square$$

接下来，我们分析基线算法的遗憾。我们的目标是证明以下定理。

定理 2。 F 或者对于任意常数 $\gamma > 0$，以概率 $1 - O(T^{-\gamma})$，算法 *4* 的遗憾为 i s

$$\tilde{O}\left(\frac{T_k}{m} + \frac{T_k}{\sqrt{T_{k-1}}} + \frac{n}{m}\left(\sum_{i=0}^{k-1}\frac{T_i}{\sqrt{T_{i-1}}}\right)\right), \tag{4}$$

˜其中， O 在 T 中隐藏了对数因子。这里，为方便起见，我们定义了 $T_{-1} = 1$ 。

由于我们假设对手是无视未来的，只需固定任意一个损失序列 ℓ_1、……、$\ell_T \in [0,1]^n$，并证明算法以高概率在该损失序列上达到上述遗憾界。我们先引入一些记号，然后介绍停留块和驱逐块的术语，并陈述一系列关于它们结构的技术引理。最后，我们将展示如何由这些引理推出该定理。

记号 我们用 e^* 表示在 T 天内损失序列为 $\ell_1, \ldots, \ell_T \in [0,1]^n$ 的最优专家。对于专家 e 和区间 $I \subseteq [T]$，我们记为

$$L_e(I) = \sum_{t \in I} \ell_t(e)$$

专家 e 在时间 I 内的损失。对于两个专家 e 和 e'，我们记为

$$R_{e,e'}(I) = L_e(I) - L_{e'}(I)$$

e 相对于专家 e' 在时间 I 内的遗憾。我们仅在 $e' = e^*$ 的情况下使用此定义。更一般地，对于两个算法 $\mathcal{A}$ 和 $\mathcal{A}'$，它们分别执行序列 $(i_1, \ldots, i_T)$ 和 $(i'_1, \ldots, i'_T)$，以及区间 $I \subseteq [T]$，我们记为

$$R_{\mathcal{A},\mathcal{A}'}(I) = \sum_{t \in I} \ell_t(i_t) - \sum_{t \in I} \ell_t(i'_t),$$

其中 w 我们称之为算法 $\mathcal{A}$ 相对于算法 $\mathcal{A}'$ 在时间 I 内的遗憾 .

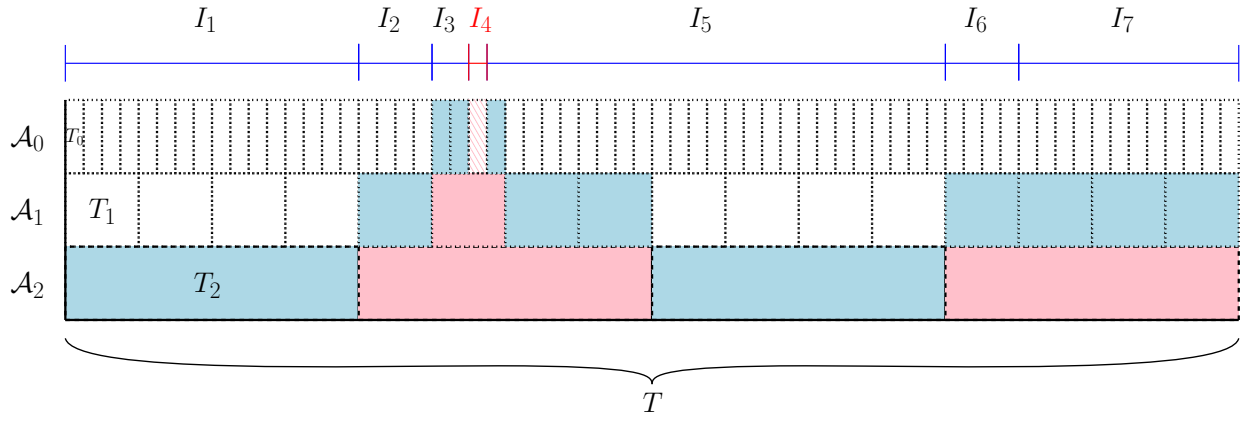

图3：我们的算法在 $k = 3$ 下的一个实例。在层级 i，时间被划分为长度为 T_i 天的区间。图中的红色区块是在 $\mathcal{S}$ 中出现的区块。上方的区间是在构造 $\mathcal{S}$ 时定义的区间。每个区间 I_j 都对应一个遗憾较低的专家 e_j，与最优专家 e^* 相比皆是如此（但 标记的 区间 I_4 除外，它对应于层级 0 的相应区块中出现的任意专家）。对于对应专家为 e_j 的区间 I_j，其下方某个层级 i 的蓝色区块 I_j 表示，在该区块所涵盖的天内，专家 e_j 位于层级 i 的池中。

停留块和驱逐块的定义 回想一下，算法 4 使用 k 个递归层级，分别记为层级 0、...、$k-1$。在层级 i，$T = T_k$ 天被划分为长度为 T_i 天的区块，其中 $T_0 < \cdots < T_k$。具体来说，给定层级 i 和 $j \in [T/T_i]$，我们定义 $B_{i,j} = \{(j-1)T_i + k \mid 1 \le k \le T_i\}$，它是层级 i 的第 j 个日块。我们将 $P'_{i,j}$ 定义为在驱逐过程和专家在层级之间转发之后、但在采样 m 个新专家之前，紧接区块 $B_{i,j}$ 所涵盖的天之前，位于层级 i 池中的专家。进一步地，我们将 $P_{i,j}$ 定义为采样 m 个新专家并将其纳入 $P'_{i,j}$ 后得到的层级 i 池。我们为这些新专家指定不同的到达时间，以确保驱逐规则在它们之间有明确定义。在分析中，我们只需将此采样视为每次采样一个专家[3]。设 $P'_{i,j} = P_i^{(0)}$，$P_i^{(1)}{}_{,j} \subset \cdots P_i^{(m)}{}_{,j} = P_{i,j}$ 为层级 i 池在依照给定到达顺序纳入每个新采样专家之后的专家多重集。

我们首先介绍如何将区块 $B_{i,j}$ 划分为两种类型，即停留块 和 驱逐块。这种划分只取决于算法各层级中专家的随机抽样，而不取决于我们作为子程序运行的各种乘法权重更新算法的结果。考虑任意一个固定的层级 i、任意一个 $j \in [T/T_i]$ 以及专家池 $P'_{i,j}$。在抽样选出要加入层级 i 的专家池 $P'_{i,j}$ 的 m 名新专家之前，我们可以提出以下假设性问题：如果我们抽样到全局最优专家 e^*，它会一直保留在层级 i 的池中直至时间结束，还是会被某个专家 e 驱逐？由于损失序列是预先固定的，而且一个专家只能被到达时间更早的另一个专家驱逐，因此，这个问题的答案只取决于新专家抽样之前池中的专家集合。

我们称区块 $B_{i,j}$ 为驻留块，若对每个 $0 \le t \le m-1$，只要 e^* 被采样为下一个专家，以便从 $P_i^{(\ell)}{}_{,j}$ 形成 $P_i^{(\ell+1)}{}_{,j}$，那么 e^* 就会一直保留在层级 i 的池中，直至时间终结。如果某个区块是驻留块，且最优专家 e^* 确实被采样（因此会一直保留到最后），我们就称该驻留块已实际化。如果 $B_{i,j}$ 不是驻留块，我们就称其为驱逐块。显然，给定区块是驻留块还是驱逐块，只取决于专家的随机抽样。重要的是，我们注意到，层级 i 上的驱逐块 $B_{i,j}$ 会产生一个专家 e 和一个天数区间 $I = \{(j-1)T_i + 1, (j-1)T_i + 2, \dots\}$，使得 $L_e(I) \le (1+\epsilon)L_{e^*}(I)$，并且专家 e 在区间 I 对应的天数内存在于层级 i 的池中。事实上，我们可以令 e 为池中任意一个会驱逐 e^* 的专家，并令 I 表示这次驱逐发生前的天数序列。于是，根据驱逐规则，$L_e(I) \le (1+\epsilon)L_{e^*}(I)$。

接下来，我们将指定驻留块的一个子集 $\mathcal{S}$。在定义 $\mathcal{S}$ 中的区块时，我们还将进一步定义某些时间区间 I_1、.....、I_r 以及专家 e_1、.....、e_r。我们还将对其中一些区间进行标记。所有这些量都是随机变量，仅取决于算法在其 k 层中执行的专家随机抽样。为定义这些量，我们使用下面这一归纳过程，该过程如图3所示：最初，我们设置 $t = 1$，并令 $\ell = k-1$。递归地，假设对于某个 $r_0 \ge 1$，我们已经定义了 I_1、.....、I_{r_0-1} 和 e_1、.....、e_{r_0-1}。进一步假设，我们以某种方式给定了 t 和 ℓ，使得在时间 t，层级 ℓ 开始一个新的日块。将该区块记为 $B(t, \ell) = \{t, t+1, \dots, t+T_\ell - 1\}$。我们分两种情况讨论。

[3]请注意，我们是从所有专家组成的集合中有放回地抽样。如果抽样到的专家已在池中，我们就会在池中加入该专家的一个新副本。因此，池中可能同时存在同一专家的多个副本，但它们会被赋予不同的到达时间，并且自各自到达以来遭受的损失也不同。在驱逐时，我们将这些专家视为不同的个体。特别是，某个专家甚至可能被自己较早抽样得到的副本驱逐。

- $B(t,\ \ell)$ **是一个驱逐块**。根据上述观察，存在一个天数区间 I，其中包含 $B(t,\ \ell)$，以及一个专家 e，在 I 中的这些天里，该专家始终处于层级 ℓ 的池中，并且 $L_e(I) \leq (1+\epsilon) L_{e^*}(I)$。我们定义 $I_{r_0} := I$ 和 $e_{r_0} := e$。我们进一步更新 $t \leftarrow t + T_\ell$。最后，我们将 ℓ 更新为 $\{0, 1, \ldots, k-1\}$ 中的最大值，使得在时间 t，层级 ℓ 开始一个新的区块。

- $B(t,\ \ell)$ **是一个驻留块**。我们将 $B(t,\ \ell)$ 纳入 $\mathcal{S}$，并进一步分情况讨论：

  1. $B(t,\ \ell)$ 已实际化：这意味着 e^* 被采样到层级 ℓ 池中，时间为 t，因此将保留至时间结束（因为 $B(t,\ \ell)$ 是一个驻留块）。我们设定 $e = e^*$ 和 $I = [t, t+1, \ldots, T]$，并终止该过程。 2. $B(t,\ \ell)$ 尚未实际化：若 $\ell = 0$，我们定义 $I_{r_0} = B(t,\ 0)$，并令 $e_{r_0} = e$ 为天数 $B(t,\ 0)$ 期间 0 层池中的任意一位专家。我们进一步标记 区间 I_{r_0}。我们更新 $t \leftarrow t + T_0$。最后，我们更新 ℓ，使其在 $\{0, 1, \ldots, k-1\}$ 中取最大值，并满足一个新区块在时间 t 开始于层级 ℓ。另一方面，若 $\ell \neq 0$，则保持 t 不变，更新 $\ell \leftarrow \ell - 1$，并继续下一步。

接下来，我们将证明一系列关于 $\mathcal{S}$ 中停留块结构的技术引理，这些停留块位于区间 I_1、.....、I_r 上，并涉及专家 e_1、.....、e_r。我们首先给出 $\mathcal{S}$ 大小的高概率上界。

引理 4。对于任意 $\delta \in (0,\ 1)$，以至少 $1-\delta$ 的概率，有 $|\mathcal{S}| \leq \frac{n}{m} \log(1/\delta)$。—

证明。每当我们在上述过程中遇到一个驻留块时，就有 m 次机会实际采样最优专家 e^*。如果确实采样到了 e^*，过程就会结束，并且我们不会再向 $\mathcal{S}$ 添加区块。由于每次采样到 e^* 的概率是 $1/n$，因此我们得到

$$\Pr[|\mathcal{S}| \geq N+1] \leq (1-1/n)^{Nm} \leq \exp\left(-\frac{Nm}{n}\right).$$

取 $N = \frac{n}{m} \log(1/\delta)$，即可得到所需结果。 □

对于下一个引理，我们将考虑合成 算法 $\mathcal{A}^{synth}$，该算法在 $i = 1$、.....、r 期间的天数 I_i 中执行专家 e_i。由于该算法需要了解专家未来的表现，因此无法实际实现，但我们将仅在分析中使用它。

引理 5. 对于任意 $\delta \in (0,1)$，以至少 $1-\delta$ 的概率（对算法 *4* 执行的专家随机抽样而言），算法 $\mathcal{A}^{synth}$ 的遗憾至多为 $\epsilon T + \frac{n}{m} T_0 \log(1/\delta)$。 —

证明。注意，区间 $I_1, \ldots, I_r$ 构成 $[T]$ 的一个划分。在任意未标记区间 I_j 内，算法 $\mathcal{A}^{synth}$ 选择专家 e_j，使得 $L_{e_j}(I_j) \leq (1+\epsilon) L_{e^*}(I_j)$。对于标记区间$I_j$，我们采用平凡的上界 $L_{e_j}(I_j) \leq |T_0|$。这是因为只有层级为 0 的区间会被标记。根据引理 4，以至少 $1-\delta$ 的概率，标记区间的数量至多为 $\frac{n}{m} \log(1/\delta)$。令 $J_0 = \{j \in [r] \mid j \text{ 为未标记的}\}$，且 $J_1 = [r] \setminus J_0$。注意，对于每个 $j \in J_0$，

$$R_{e_j, e^*}(I_j) = L_{e_j}(I_j) - L_{e^*}(I_j) \leq \epsilon L_{e^*}(I_j) \leq \epsilon |I_j|.$$

因此，以至少 $1-\delta$ 的概率，算法 $\mathcal{A}^{synth}$ 的遗憾至多为

$$|J_1||T_0| + \sum_{j \in J_0} R_{e_j, e^*}(I_j) \leq \frac{n}{m} T_0 \log(1/\delta) + \sum_{j \in J_0} \epsilon |I_j| \leq \frac{n}{m} T_0 \log(1/\delta) + \epsilon T,$$

如所需。 □

现在我们可以证明定理 2 了。

定理 *2* 的证明。我们首先指出，专家的采样、转发和驱逐与用于执行各种乘法权重更新的随机比特完全无关。事实上，某个专家是否被驱逐或转发，仅取决于自其被采样以来相较于其他专家的损失。因此，我们可以固定 一次采样，然后仅针对乘法权重更新的随机性界定遗憾。更准确地说，我们固定一种专家采样，使得 $|\mathcal{S}'| = O(\frac{n}{m}\log\ T)$；根据引理 4，这以高概率在 T 中成立。

我们先简要回顾算法 4 中的乘法权重更新如何运作，并为证明引入一些记号。首先，对于每个层级 i，我们在每个长度为 T_i 天的区块上，对专家池运行乘法权重更新。我们将这些算法称为 $\mathcal{A}'_0$、.....、$\mathcal{A}'_{k-1}$，并按层级标记。我们定义 $\mathcal{A}_0 = \mathcal{A}'_0$，并递归地令 $\mathcal{A}_i$ 为在 $\mathcal{A}'_i$ 和 $\mathcal{A}_{i-1}$ 之间运行乘法权重更新、每经过一个长度为 T_i 的区块便重新启动所得到的算法。于是，算法 4 就是算法 $\mathcal{A}_{k-1}$。我们定义$\mathcal{S}_i = \{B \in \mathcal{S} \mid |B| = |T_i|\}$，即层级 i 上的 $\mathcal{S}$ 区块。进一步定义时间子集 $J_0 \subseteq J_1 \subseteq \cdots \subseteq J_k = [T]$ 如下： $J_k = T$；对于 $0 \le i \le k-1$，我们令 $J_i = \bigcup_{B\in\mathcal{S}_i} B$。我们将通过归纳法证明以下结论:

断言1. 对于任意常数 $\gamma > 0$，存在一个常数 $C > 0$，使得以概率 $1 - O(T^{-\gamma})$ 对每个 $0 \le i \le k-1$,

$$R_{\mathcal{A}_i,\mathcal{A}^{synth}}(J_i) \le C\sqrt{\log nT} \cdot \sum_{j=0}^{i} \frac{|\mathcal{S}_j|T_j}{\sqrt{T_{j-1}}}. \tag{5}$$

这里，为方便起见，我们定义了 $T_{-1} = 1$。此外，

$$R_{\mathcal{A}_{k-1},\mathcal{A}^{synth}}(T) \le C\sqrt{\log nT} \cdot \left(\frac{T}{\sqrt{T_{k-1}}} + \sum_{j=0}^{k-1} \frac{|\mathcal{S}_j|T_j}{\sqrt{T_{j-1}}}\right). \tag{6}$$

断言的证明。我们首先固定一个足够大的 C，使得以概率 $1 - O(T^{-\gamma})$，对于任意天数 T'，我们运行的所有不同乘法权重更新算法相较于最佳固定选项的遗憾至多为 $(C/2)\sqrt{\log(nT)}\sqrt{T'}$。根据引理2，这是可行的。

我们通过对 i 使用归纳法来证明该断言。对于 $i = 0$，显然有

$$R_{\mathcal{A}_0,\mathcal{A}^{synth}}(J_0) \le |J_0| = \frac{|\mathcal{S}_0|T_0}{\sqrt{T_{-1}}}.$$

现在令 $i \in \{1,\ldots,k-1\}$ 给定，并通过归纳假设，假定该界对小于 i 的值成立。我们声称，由于算法各层级之间的专家转发以及驱逐规则，对于层级 $i-1$ 上每个包含 T_{i-1} 天的区块 $B \subseteq J_i \setminus J_{i-1}$，相应层级 $i-1$ 的池在 B 的这些天内都包含由 $\mathcal{A}^{synth}$ 选出的专家。原因如下：注意，任何包含 B 中最左侧一天的区间 I_j，实际上都会根据我们的驱逐规则包含 B 的所有天；也就是说，如果专家 e_j 能存活到区块 B 的开始，那么它最早也要到区块 B 结束时才会被驱逐。此外，由于转发，在区块 B 的这些天里，e_j 确实会出现在层级 $i-1$ 的池中。根据乘法权重更新的保证，可得

$$R_{\mathcal{A}_{i-1},\mathcal{A}^{synth}}(J_i \setminus J_{i-1}) \le (C/2)\sqrt{T_{i-1}\log(nT)} \cdot \frac{|J_i \setminus J_{i-1}|}{T_{i-1}} \le (C/2)\sqrt{\log nT}\frac{|\mathcal{S}_i|T_i}{\sqrt{T_{i-1}}}.$$

将此与归纳假设结合起来，

$$R_{\mathcal{A}_{i-1},\mathcal{A}^{synth}}(J_i) \leq (C/2)\sqrt{\log nT}\frac{|\mathcal{S}_i|T_i}{\sqrt{T_{i-1}}} + C\sqrt{\log nT}\cdot\sum_{j=0}^{i-1}\frac{|\mathcal{S}_j|T_j}{\sqrt{T_{j-1}}}.$$

最后，根据 $\mathcal{A}_i'$ 和 $\mathcal{A}_{i-1}$ 之间的乘法权重更新保证（每经过长度为 T_i的区块后都会重新启动），

$$\begin{aligned}
R_{\mathcal{A}_i,\mathcal{A}^{synth}}(J_i) \leq & R_{\mathcal{A}_{i-1},\mathcal{A}^{synth}}(J_i) + (C/2)\sqrt{\log nT}|\mathcal{S}_i|\sqrt{T_i} \\
\leq & R_{\mathcal{A}_{i-1},\mathcal{A}^{synth}}(J_i) + (C/2)\sqrt{\log nT}\frac{|\mathcal{S}_i|T_i}{\sqrt{T_{i-1}}} \\
\leq & C\sqrt{\log nT}\cdot\sum_{j=0}^{i}\frac{|\mathcal{S}_j|T_j}{\sqrt{T_{j-1}}}
\end{aligned}$$

其中，倒数第二个不等式用到了 $|T_i| > |T_{i-1}|$。这就是所需的上界。最后，(6)基本上可由相同的论证得出。由于专家转发，对于层级 $k-1$上的每个日块 $B \subseteq T \setminus J_{k-1}$，层级 $k-1$的池都包含 $\mathcal{A}^{synth}$在 B这些天里所采用的专家。因此，

$$R_{\mathcal{A}_{k-1},\mathcal{A}^{synth}}(T) \leq C\sqrt{\log nT}\cdot\frac{T}{\sqrt{T_{k-1}}} + R_{\mathcal{A}_{k-1},\mathcal{A}^{synth}}(J_{k-1}),$$

其中，第一项来自乘法权重更新在 T_{k-1} 个区块之间重新启动，第二项是 $\mathcal{A}_{k-1}$在 J_{k-1}中的遗憾。将 (5) 中的上界代入 $i = k-1$，即可得到所需结果。 □

为了根据该命题完成证明，我们只需对每个 $|\mathcal{S}_i| \leq |\mathcal{S}'| = O(\frac{n}{m}\log T)$取上界。因此，我们得到

$$R_{\mathcal{A}_{k-1},\mathcal{A}^{synth}}(T) = O\left(\log(Tn)^2\cdot\left(\frac{T}{\sqrt{T_{k-1}}} + \frac{n}{m}\sum_{j=0}^{k-1}\frac{T_j}{\sqrt{T_{j-1}}}\right)\right).$$

将此结果与引理 5（其中 $1/\delta$ 是关于 T 的一个次数足够高的多项式）结合起来，即可得到所需的遗憾界。 □

5.2.1 参数设置

本节将设置定理 2 中的参数，以得到以下定理。

定理 3. 设 $T \geq C\frac{n}{m}$，其中常数 C 足够大。对于算法 *4*，存在一组参数，使得其遗憾以关于 T 的高概率满足

$$\tilde{O}\left(\frac{T}{m} + \sqrt{\frac{Tn}{m}}\right). \tag{7}$$

证明。我们先固定 T_i 的取值。令 $T_{-1} = 1$和 $T_0 = (T\frac{m}{n})^{2^{k+1}-1}$。注意，根据关于 T 的假设，$T_0 \geqq \sqrt{C} > T_{-1}$。现在通过归纳，对于 $1 \leq i \leq k-1$，我们设定 $T_i = \frac{T_{i-1}^{3/2}}{\sqrt{T_{i-2}}}$。

我们指出，这组参数的取值满足方程组

$$\frac{T}{\sqrt{T_{k-1}}} = \frac{n}{m}\frac{T_{k-1}}{\sqrt{T_{k-2}}} = \frac{n}{m}\frac{T_{k-2}}{\sqrt{T_{k-3}}} = \cdots = \frac{n}{m}\frac{T_1}{\sqrt{T_0}} = \frac{n}{m}T_0.$$

通过简单的归纳论证可知，对每个 i, $T_i > T_{i-1}$都成立。事实上，另一个直接的归纳论证表明，对于 $0 \le i \le k-1$，我们有等式 $T_i = T_0^{\,2^i}$。注意，$2^{k+1}-2T_{k-1} = \binom{Tm}{n}^{2^{k+1}-1} < T$，因此这是一个合法的参数设定。现在显然，采用这组参数后，对于 $i = 1$、.....、$k-1$，$\sqrt[T_i]{T_{i-1}} = \sqrt[T_{i-1}]{T_{i-2}}$。此外，很容易验证 $\sqrt[T]{T_{k-1}} = \frac{n}{m}\sqrt[T_{k-1}]{T_{k-2}}$。因此，由定理 2 可知，遗憾有界于

$$\tilde{O}\left(\frac{T}{m} + \frac{kn}{m}T_0\right).$$

注意，对每个 $1 \le i \le k-1$,

$$\frac{T_i}{T_{i-1}} = T_0^{1/2^i} \ge T_0^{1/2^k} \ge \left(T\frac{m}{n}\right)^{\frac{1}{2^{k+1}}},$$

因此，选择 $k = \lg\lg(Tm/n) - O(1)$，其中 $O(1)$表示一个足够大的常数，块大小之间的间隔便可大于任意常数。采用这组 k后，

$$T_0 = \left(\frac{Tm}{n}\right)^{1/2+\frac{1}{2^{k+1}-1}} = O\left(\sqrt{\frac{Tm}{n}}\right),$$

因此，我们得到最终的遗憾界：

$$\tilde{O}\left(\frac{T}{m} + k\sqrt{\frac{Tn}{m}}\right) = \tilde{O}\left(\frac{T}{m} + \sqrt{\frac{Tn}{m}}\right),$$

如期望的那样。 □

6 自举

本节介绍先前工作中的自举技术。为方便起见，我们用以下引理重述其结果。

引理 1. 假设对于具有n 位专家、每日损失范围为 1 （即最大损失与最小损失之差）的专家问题，存在两个算法 *ALG1* 和 *ALG2*，其参数如下：*ALG1* 运行 T_1 天，遗憾至多为 R_1；*ALG2* 运行 T_2 天，遗憾至多为 R_2；二者的成功概率均为 $1-\delta$。此外，假设两个算法的空间复杂度均为 m。那么，存在一个算法 *ALG'*，运行 $T_1 \cdot T_2$ 天，遗憾至多为$R_1 \cdot R_2 + O\left(T_1 \cdot \sqrt{T_2 \ln(n/\delta)}\right)$，成功概率为 $1 - \delta\,(nT_1 + 1)$。此外，*ALG'* 的空间复杂度至多为 $O(m)$。

证明。令 j^* 为最佳专家的索引。该算法运行 T_1 轮，每轮包含 T_2天。假设我们运行 ALG2 的一个副本，并在每轮开始时重置。对于每一轮 i 和每位原始专家e_j，令合成专家 s_j 为在两个选项（原始专家 e_j 和 ALG2 的副本）之间使用乘法权重更新（MWU）得到的答案。根据 ALG2 的保证，ALG2 的副本在第 i轮中的遗憾至多为R_2。根据并集界，以概率 $1-\delta(nT_1+1)$，我们假定所有 MWU 应用和算法均成功。根据 MWU 的保证，$e_{i,\ j}$ 的遗憾至多比 ALG2 高$R_3 := O\left(\sqrt{T_2 \ln(n/\delta)}\right)$。接下来，考虑一个定义在各轮上的新专家问题，定义如下。令 $loss_i(e)$ 为专家 e 在第 i 轮的损失，以及截断损失

$$tloss_i(e) := \max\left(loss_i(e) - loss_i(ALG2), -R_2\right)$$

我们注意到截断损失的两个重要性质。首先，请注意 $|tloss_i(s_j)| \leq \max(R_2, R_3)\ \ \forall j$。其次，对于最佳专家 e_{j^*}，我们有

$$tloss_i(s_{j^*}) + loss_i(ALG2) = \max\left(loss_i(s_{j^*}), loss_i(ALG2) - R_2\right) \leq loss_i(e_{j^*}) + R_3.$$

我们使用截断损失，在这些轮次中对 n 个合成专家运行 ALG1。根据 ALG1 的保证，与最佳合成专家相比，该算法的遗憾至多为$R_1 \cdot \max(R_2, R_3)$。我们有

$$\begin{aligned}\sum_i loss_i(ALG1) &\leq \sum_i tloss_i(ALG1) + loss_i(ALG2)\\ &\leq R_1 \cdot \max(R_2, R_3) + \sum_i tloss_i(s_{j^*}) + loss_i(ALG2)\\ &\leq R_1 \cdot \max(R_2, R_3) + \sum_i \left(loss_i(e_{j^*}) + R_3\right).\end{aligned}$$

因此，总遗憾至多为 $O(T_1 R_3) + R_1 R_2$。

请注意，空间复杂度也会增加一个常数乘法因子。这是因为我们只需为 ALG1 实际使用的专家创建合成专家。由于它们的空间复杂度都至多为 m，ALG1 每天至多查询 m 个专家（参见我们在第4节中的空间模型）。因此，在创建合成专家时，为了在乘法权重更新中跟踪额外参数所需的空间，可以计入 ALG1 所跟踪的专家。同样，ALG2 的任何副本也需要 $O(m)$ 的额外空间，但我们每天只有一个 ALG2 副本。

□

推论1. 假设存在一种算法，在 T 天内以至少 $1-\delta$ 的概率实现遗憾值 R，并使用空间 m。那么，对于 $i \geq 1$，存在一种算法，在 T^i 天内实现遗憾值

$$O\left(\left(R^i + \sum_{j=0}^{i-2} R^j T^{(i+1-j)/2}\right)\sqrt{\ln(n/\delta)}\right),$$

以至少 $1-\delta(nT+1)^{i-1}$ 的概率。此外，存在一个常数 C，使得该方法使用的空间至多为 $C^i m$。

证明。 该推论可由归纳法立即得出。情形 $i = 1$ 由假设得出。对于$i > 1$，我们设定 $T_1 = T, R_1 = R$和 $T_2 = T^{i-1}$ 并

$$R_2 = \left(R^{i-1} + \sum_{j=0}^{i-3} R^j T^{(i-j)/2} \right) \sqrt{\ln(n/\delta)}$$

并代入引理。 □

7 将所有内容整合起来

现在，我们可以将上一节的结果整合起来，证明本文的主要定理。

定理1. 对于任意 $\delta \in (0,1)$，存在一种用于在线专家问题的算法，可在 $T = n^{O(1)}$ 天内使用空间 $\tilde{O}(n^{\delta})$，并以概率 $1-1/\mathrm{poly}(T)$达到遗憾 $\tilde{O}(n^2 T^{\frac{1}{1+\delta}})$。

证明。 首先假设 T 具有形式 $(nm)^i$，其中 i是某个整数。注意 $i = O(1)$。根据定理3，存在一种算法，可在长度为 $T_0 = nm$ 天的序列上以高概率在 T内达到遗憾 $O(n)$。将该算法作为基础算法代入推论1，可得一种算法，可在 T 天内达到遗憾

$$R = \tilde{O}\left(n^i + \sum_{j=0}^{i-2} n^j (nm)^{\frac{i+1-j}{2}} \right) = \tilde{O}(n^{i+1})$$

也以高概率在 T 内。（注意，多对数因子也会增大一个常数 ˜ 倍的因子 i，该因子被隐藏在 O 记号中。）这里我们只对所有出现的 m 将 $m \le n$ 作了上界估计。利用 $T = (nm)^i$，并令 $m = n^{\delta}$，可得

$$R = \tilde{O}\left(nT^{\frac{1}{1+\delta}} \right),$$

正如所需。

如果 T 不是 $(nm)^i$ 的形式，我们就选择满足 $T' = (nm)^i \ge T$ 的最小 $i \in \mathbb{N}$，并人为扩展损失向量序列：在剩余的 $T' - T$ 天里，给予所有专家相同的损失。然后，我们使用与 T' 相对应的参数运行算法，但在 T 天后终止。算法在扩展序列上的遗憾与原始序列上的遗憾相同，并且有如下界

$$\tilde{O}\left(nT'^{\frac{1}{1+\delta}} \right) = \tilde{O}\left(n(nmT)^{\frac{1}{1+\delta}} \right) = \tilde{O}\left(n^2 T^{\frac{1}{1+\delta}} \right),$$

正如所需。根据引理 3，可以界定算法所使用的空间。注意，由于 i 是常数，自举过程根据推论 1 只会使空间增加一个常数因子。回顾一下，基线算法使用了 $k = O(\log \log n)$ 个递归层，因此根据该引理，空间使用量可界定为 $O(k^3 m) = \tilde{O}(n^{\delta})$。

□

注释1。 与先前的研究 *[PZ23]*, 一样，我们的结果可以很容易地扩展到处理 T未知的情况：我们只需像定理*1*的证明中那样猜测 T 的值。更具体地说，我们用几何级数递增的 T 猜测值运行算法的多个副本，并在所有不同的算法副本上运行乘法权重更新。这只会使空间和遗憾界额外增加一个 $O(\log n)$ 因子。

参考文献

[AHK12] Sanjeev Arora、Elad Hazan 和 Satyen Kale。《乘法权重更新方法：一种元算法及其应用》。计算理论，8(1):121–164，2012年。[AW20] Sepehr Assadi 和 Chen Wang。《有限内存下的探索：用于抛硬币、带噪比较和多臂老虎机的流式算法》。载于 Konstantin Makarychev、Yury Makarychev、Madhur Tulsiani、Gautam Kamath 和 Julia Chuzhoy 编，*第52届ACM SIGACT年度计算理论研讨会论文集，STOC 2020，美国伊利诺伊州芝加哥，2020年6月22日至26日*，第1237–1250页。ACM，2020年。[BIK+22] Ainesh Bakshi、Piotr Indyk、Praneeth Kacham、Sandeep Silwal 和 Samson Zhou。《通过核密度估计求解核矩阵的次二次算法》。*CoRR*，abs/2212.00642，2022年。[Bro51] George W Brown。《通过虚拟博弈迭代求解博弈》。*Act. Anal. ProdAllocation*，13(1):374，1951年。[CBL06] Nicolo Cesa-Bianchi 和 G´abor Lugosi。《预测、学习与博弈》。剑桥大学出版社，2006年。[CK20] Arghya Roy Chaudhuri 和 Shivaram Kalyanakrishnan。《使用有界臂记忆实现多臂老虎机的遗憾最小化》。载于第三十四届*AAAI*人工智能大会（*AAAI 2020*）、第三十二届人工智能创新应用大会（*IAAI 2020*）、第十届*AAAI*人工智能教育进展研讨会（*EAAI 2020*），美国纽约市，*2020*年*2*月*7*日至*12*日，第10085–10092页。AAAI出版社，2020年。[CKM+11] Paul F. Christiano、Jonathan A. Kelner、Aleksander Madry、Daniel A. Spielman 和 Shang-Hua Teng。《无向图中的电流、拉普拉斯系统与更快的最大流近似》。载于 Lance Fortnow 和 Salil P. Vadhan 编，第*43*届*ACM*计算理论研讨会论文集（*STOC 2011*），美国加利福尼亚州圣何塞，*2011*年*6*月*6*日至*8*日，第273–282页。ACM，2011年。[CO96] Thomas M. Cover 和 Erik Ordentlich。《带辅助信息的通用投资组合》。*IEEE*信息论汇刊，42(2):348–363，1996年。[DKS19] Yuval Dagan、Gil Kur 和 Ohad Shamir。《流式模型中线性预测的空间下界》。载于 Alina Beygelzimer 和 Daniel Hsu 编，学习理论会议（*COLT 2019*），美国亚利桑那州菲尼克斯，*2019*年*6*月*25*日至*28*日，机器学习研究论文集第99卷，第929–954页。PMLR，2019年。[DS18] Yuval Dagan 和 Ohad Shamir。《用少量内存和通信检测相关性》。载于 S´ebastien Bubeck、Vianney Perchet 和 Philippe Rigollet 编，学习理论会议（*COLT 2018*），瑞典斯德哥尔摩，*2018*年*7*月*6*日至*9*日，机器学习研究论文集第75卷，第1145–1198页。PMLR，2018年。

[FS95] Yoav Freund 和 Robert E. Schapire。决策理论对在线学习的推广及其在提升中的应用。载于 Paul M. B. Vit´anyi 编，计算学习理论：第二届欧洲会议，*EuroCOLT '95*，西班牙巴塞罗那，*1995*年*3*月*13*日至*15*日，论文集，计算机科学讲义第904卷，第23–37页。施普林格，1995年。[GH16] Dan Garber 和 Elad Hazan。用于近似半定规划的次线性时间算法。《数学规划》，158(1-2):329–361，2016年。[GRT19] Sumegha Garg、Ran Raz 和 Avishay Tal。双遍学习的时间-空间下界。载于 Amir Shpilka 编，第*34*届计算复杂性会议，*CCC 2019*，*2019*年*7*月*18*日至*20*日，美国新泽西州新不伦瑞克，LIPIcs第137卷，第22:1–22:39页。Schloss Dagstuhl - Leibniz-Zentrum f¨ur Informatik，2019年。[Haz16] Elad Hazan。在线凸优化导论。《优化基础与趋势》，2(3-4):157–325，2016年。[HLZ20] Samuel B. Hopkins、Jerry Li 和 Fred Zhang。通过遗憾最小化，轻松实现稳健的重尾均值估计。载于 Hugo Larochelle、Marc' Aurelio Ranzato、Raia Hadsell、Maria-Florina Balcan 和 Hsuan-Tien Lin 编，神经信息处理系统进展*33*：*2020*年神经信息处理系统年会，*NeurIPS 2020*，*2020*年*12*月*6*日至*12*日，线上，2020年。[Ind06] Piotr Indyk。稳定分布、伪随机生成器、嵌入与数据流计算。《*ACM*期刊》，53(3):307–323，2006年。[JHTX21] Tianyuan Jin、Keke Huang、Jing Tang 和 Xiaokui Xiao。多臂老虎机的最优流式算法。载于 Marina Meila 和 Tong Zhang 编，第*38*届国际机器学习会议论文集，*ICML 2021*，*2021*年*7*月*18*日至*24*日，线上活动，机器学习研究论文集第139卷，第5045–5054页。PMLR，2021年。[KM17] Adam R. Klivans 和 Raghu Meka。使用乘法权重学习图模型。载于 Chris Umans 编，第*58*届*IEEE*计算机科学基础年会，*FOCS 2017*，美国加利福尼亚州伯克利，*2017*年*10*月*15*日至*17*日，第343–354页。IEEE计算机学会，2017年。[KRT17] Gillat Kol、Ran Raz 和 Avishay Tal。学习稀疏奇偶校验的时间-空间困难性。载于 Hamed Hatami、Pierre McKenzie 和 Valerie King 编，第*49*届*ACM SIGACT*计算理论年会论文集，*STOC 2017*，加拿大魁北克省蒙特利尔，*2017*年*6*月*19*日至*23*日，第1067–1080页。ACM，2017年。[KV03] Adam Kalai 和 Santosh S. Vempala。在线决策问题的高效算法。载于 Bernhard Sch¨olkopf 和 Manfred K. Warmuth 编，计算学习理论与核机器：第*16*届计算学习理论年会暨第*7*届核方法研讨会，*COLT/Kernel 2003*，美国哥伦比亚特区华盛顿，*2003*年*8*月*24*日至*27*日，论文集，计算机科学讲义第2777卷，第26–40页。施普林格，2003年。

[LSPY18] David Liau、Zhao Song、Eric Price 和 Ger Yang。常数空间中的随机多臂老虎机。载于 Amos J. Storkey 和 Fernando P´erez-Cruz 编，人工智能与统计国际会议，*AISTATS 2018*，*2018*年*4*月*9*日至*11*日，西班牙加那利群岛兰萨罗特岛 *Playa Blanca*，机器学习研究论文集第84卷，第386–394页。PMLR，2018年。[LW89] Nick Littlestone 和 Manfred K. Warmuth。加权多数算法。载于第*30*届计算机科学基础年度研讨会，美国北卡罗来纳州三角研究园，*1989*年*10*月*30*日至*11*月*1*日，第256–261页。IEEE计算机学会，1989年。[MCJ13] Ioannis Mitliagkas、Constantine Caramanis 和 Prateek Jain。内存受限的流式PCA。载于 Christopher J. C. Burges、L´eon Bottou、Zoubin Ghahramani 和 Kil- ian Q. Weinberger 编，神经信息处理系统进展*26*：第*27*届神经信息处理系统年会，*2013*年神经信息处理系统会议。会议论文集，会议于*2013*年*12*月*5*日至*8*日在美国内华达州太浩湖举行，第2886–2894页，2013年。[MPK21] Arnab Maiti、Vishakha Patil 和 Arindam Khan。具有有界臂记忆的多臂老虎机：最优臂识别和遗憾最小化的近乎最优保证。载于 Marc'Aurelio Ranzato、Alina Beygelzimer、Yann N. Dauphin、Percy Liang 和 Jennifer Wortman Vaughan 编，神经信息处理系统进展*34*：*2021*年神经信息处理系统年会，*NeurIPS 2021*，*2021*年*12*月*6*日至*14*日，线上，第19553–19565页，2021年。[MSSV22] Annie Marsden、Vatsal Sharan、Aaron Sidford 和 Gregory Valiant。高效凸优化需要超线性内存。载于 Po-Ling Loh 和 Maxim Raginsky 编，学习理论会议，*2022*年*7*月*2*日至*5*日，英国伦敦，机器学习研究论文集第178卷，第2390–2430页。PMLR，2022年。[Nis92] Noam Nisan。空间受限计算的伪随机生成器。《组合学》，12(4):449–461，1992年。[OC98] Erik Ordentlich 和 Thomas M. Cover。实现事后最优投资组合的代价。《运筹研究数学》，23(4):960–982，1998年。[PZ23] Binghui Peng 和 Fred Zhang。次线性空间中的在线预测。载于*2023*年*ACM-SIAM*离散算法年度研讨会（*SODA*）论文集，2023年。[Raz17] Ran Raz。一大类学习问题的时间-空间下界。载于 Chris Umans 编，第*58*届*IEEE*计算机科学基础年会，*FOCS 2017*，美国加利福尼亚州伯克利，*2017*年*10*月*15*日至*17*日，第732–742页。IEEE计算机学会，2017年。[RR07] Ali Rahimi 和 Benjamin Recht。面向大规模核机器的随机特征。载于 John C. Platt、Daphne Koller、Yoram Singer 和 Sam T. Roweis 编，神经信息处理系统进展*20*：第二十一届神经信息处理系统年会论文集，加拿大不列颠哥伦比亚省温哥华，*2007*年*12*月*3*日至*6*日，第1177–1184页。Curran Associates, Inc.，2007年。

[SD15] Jacob Steinhardt 和 John C. Duchi。内存受限稀疏线性回归的极小极大速率。载于 Peter Gr¨unwald、Elad Hazan 和 Satyen Kale 编，第*28*届学习理论会议论文集，*COLT 2015*，巴黎，法国，*2015*年*7*月*3*日至*6*日，第40卷，*JMLR*研讨会与会议论文集，第1564–1587页。JMLR.org，2015年。[Sha14] Ohad Shamir。统计学习与估计的在线和分布式算法的基本极限。载于 Zoubin Ghahramani、Max Welling、Corinna Cortes、Neil D. Lawrence 和 Kilian Q. Weinberger 编，神经信息处理系统进展*27*：*2014*年神经信息处理系统年度会议，*2014*年*12*月*8*日至*13*日，蒙特利尔，魁北克，加拿大，第163–171页，2014年。[SSV19] Vatsal Sharan、Aaron Sidford 和 Gregory Valiant。小误差线性回归的内存-样本权衡。载于 Moses Charikar 和 Edith Cohen 编，第*51*届*ACM SIGACT*计算理论研讨会论文集，*STOC 2019*，菲尼克斯，美国亚利桑那州，*2019*年*6*月*23*日至*26*日，第890–901页。ACM，2019年。[SVW16] Jacob Steinhardt、Gregory Valiant 和 Stefan Wager。内存、通信与统计查询。载于 Vitaly Feldman、Alexander Rakhlin 和 Ohad Shamir 编，第*29*届学习理论会议论文集，*COLT 2016*，美国纽约，*2016*年*6*月*23*日至*26*日，第49卷，*JMLR*研讨会与会议论文集，第1490–1516页。JMLR.org，2016年。[SWXZ22] Vaidehi Srinivas、David P. Woodruff、Ziyu Xu 和 Samson Zhou。专家问题的内存界。载于第*54*届*ACM SIGACT*计算理论研讨会论文集，STOC 2022，第1158–1171页，美国纽约市，2022年。计算机协会。[WS00] Christopher K. I. Williams 和 Matthias W. Seeger。使用 Nystr¨om 方法加速核机器。载于 Todd K. Leen、Thomas G. Dietterich 和 Volker Tresp 编，神经信息处理系统进展*13*，神经信息处理系统（*NIPS*）*2000*论文集，美国丹佛，加利福尼亚州，美国，第682–688页。MIT出版社，2000年。[WS19] Blake E. Woodworth 和 Nathan Srebro。开放问题：有限内存下凸优化的预言机复杂度。载于 Alina Beygelzimer 和 Daniel Hsu 编，学习理论会议，*COLT 2019*，美国菲尼克斯，*2019*年*6*月*25*日至*28*日，第99卷，机器学习研究论文集，第3202–3210页。PMLR，2019年。