arXiv:2305.01714v1 [cs.DS] 2 May 2023

流式　环境下使用渐近最优颜色的边着色

Soheil Behnezhad MohammadSaneian 东北大学 东北大学

摘要

给定一个图 G，边着色是指为 G 的边分配颜色，使得任意两条共享端点的边获得不同的颜色。根据 Vizing 著名的定理，最大度为 Δ 的任意图，若要进行正确的边着色，至少需要 Δ 种颜色，至多需要 $(\Delta+1)$ 种颜色。本文研究流式环境下的边着色。边以任意顺序逐条到达。算法对输入进行单遍处理，并且必须使用远小于输入规模的空间输出解。由于边着色的输出与其输入一样庞大，分配的颜色也应以流式方式报告。

过去几年里，流式边着色问题已在一系列研究中得到探讨。主要挑战在于，算法无法“记住”它返回的所有颜色分配。为了确保解的有效性，现有算法使用的颜色数远多于 Vizing 界$\tilde{}$。具体而言，对于 n 个顶点的图，采用 $O(ns)$ 空间[1] 的最先进算法需要 $O(\Delta^2/s+\Delta)$ 种颜色。特别要注意的是，要实现渐近最优的$O(\Delta)$ 着色，该算法需要 $\Omega(n\Delta)$ 空间，这与输入规模相当。能否用亚线性空间实现这种着色，仍是一个悬而未决的问题。

在本文中，我们对这个问题给出了肯定的回答。我们提出了一种随机算法，能够使用 $\widetilde{O}(n\sqrt{\Delta})$ 空间返回渐近最优的 $O(\Delta)$ 边着色。更一般地，我们的算法能够使用 $\widetilde{O}(ns)$ 空间返回一个规范的 $O(\Delta^{1.5}/s+\Delta)$ 边着色，在整个 s 范围内优于以往的算法。

[1]本文中及全文其他部分，$\widetilde{O}(f) = O(f \cdot \text{多对数 } n)$。

1 引言

受其在海量图处理中的应用所驱动，图流模型在过去二十年中受到了广泛关注。该模型假设输入图过于庞大，无法存储在内存中，因此以边序列的形式呈现给算法。在该模型下，许多图问题已得到研究，包括最大匹配 [14, 35, 5, 27, 6], 图连通性[1, 9], 顶点着色 [10, 13, 12, 19, 21, 11], 边着色 [16, 22, 4], 割稀疏化器和谱稀疏化器 [2, 36, 7, 23, 37], 以及图聚类 [17, 18, 25, 3] 等诸多问题（这绝非一份详尽的清单）。本文继续沿着图流模型下边着色问题的研究方向展开。边着色的目标是为图的各条边分配颜色，使得任意两条相邻边都具有不同的颜色。由于边着色的输出规模与输入规模一样大，流式算法无法将其完整存储在内存中。因此，目标还在于以数据流的方式返回解决方案。为此，主要挑战在于，算法无法“记住”所有已报告的边颜色，却必须确保任意两条关联边获得不同的颜色。

边着色是图论中的一个基本问题，在调度、通信网络和 VLSI 设计等领域有许多实际应用。根据 Vizing 的经典结论，最大度为 Δ 的任意图，正确进行边着色至少需要 Δ 种颜色，至多需要 $(\Delta+1)$ 种颜色。[2] 虽然现有的求解 $(\Delta+1)$ 边着色算法相当复杂，但只需使用一个简单的贪心算法，依次遍历各条边，并为每条边任意选择一种可用颜色，即可得到 $(2\Delta-1)$ 着色。遗憾的是，即便是这个简单的贪心算法，在流式环境中也难以实现。请注意，该算法无法在内存中记住所有已分配的边颜色，因此难以判断下一条到达的边有哪些可用颜色。

已有工作：Behnezhad、Derakhshan、Hajiaghayi、Knittel 和 Saleh [16] 首先研究了流式边着色问题，并提出一种随机化算法，使用 $\sim O(n)$ 空间，对图进行 $O(\Delta^2)$ 边着色，其中 n 为顶点数。在后续工作中，Charikar 和 Liu [22] 证明，更一般地，对于任意参数 $s \geq 1$ ，都存在一种随机化流式算法，能够O使用 $\widetilde{O}(ns)$ 空间对图进行 $(\Delta^2/s+\Delta)$ 边着色。之后，Ansari、Saneian 和 Zarrabi-Zadeh[4] 使用简单而清晰的确定性算法，获得了相同的界。值得注意的是，对于渐近最优的 $O(\Delta)$ 边着色，上述算法需要 $O(n\Delta)$ 空间。遗憾的是，这并非输入规模的次线性空间，因为最大度为 Δ 的任意图至多有$O(n\Delta)$ 条边。换言之，对于 $O(\Delta)$ 着色，目前还没有已知方法优于这样一种朴素算法：将整张图存入内存，然后再对其着色。这一现状留下了一个重要的开放问题：

是否存在一种使用亚线性空间的流式算法，用于对 $O(\Delta)$ 进行边着色?

我们的贡献：本文对上述问题给出了肯定的回答。我们的主要结果是以下算法：

定理 1.1. 对任意 $s \geq 1$，存在一种随机化流式算法，能够以高概率报告一个 $O(\Delta^{1.5}/s+\Delta)$ 边着色，适用于任意边到达顺序，并使用 $\widetilde{O}(ns)$ 空间。

Se代入 $s=\sqrt{\Delta}$，我们得到以下推论，从而回答了上述问题。 :

[2]To 参见下界，注意到仅为给一个度为 d 的顶点的边着色就需要 Δ 种颜色 度数 Δ。

推论 1.2. 存在一种随机化流式算法，在任意边到达顺序下，以高概率报告一个使用 $\widetilde{O}(n\sqrt{\Delta})$ 空间的 a$O(\Delta)$ 边着色。

~我们证明，在任意顶点到达的情形下，空间复杂度可以进一步改进为 $O(n)$。这里，图的顶点不是按任意顺序逐条到达边，而是逐个到达；当顶点 v 到达时，它与此前顶点之间的所有边都会被揭示。

定理 1.3. 存在一种随机化流式算法，在~任意顶点到达的情形下，使用 $O(n)$ 空间为图进行 $O(\Delta)$边着色。

1.1 进一步的相关工作

在边以随机顺序到达这一额外假设下，边着色的流式算法也得到了研究。Behnezhad、Derakhshan、Hajiaghayi、Knittel 和 Saleh [16] 证明，存在一种单遍 $O(n)$ 空间算~法，可得到 5.44Δ 着色。Charikar 和 Liu [22] 随后证明，颜色数量实际上可以改进为 $(1+o(1))\Delta$，同时满足~随机到达条件，并保持内存用量为 $O(n)$。这两种算法都高度依赖随机到达假设，对抗性边到达并不适用，而这正是本文的研究重点。

在线边着色是另一个相关问题。在这个问题中，算法没有空间限制，但边逐条到达，且每条边都必须在到达时立即着色，不能再更改。注意，贪心算法很容易在在线设置中实现；因此，许多研究都聚焦于能否改进贪心算法的界。对于低度图，Bar-Noy、Motwani 和 Naor [15] 证明，贪心算法对于在线边着色是最优的。然而，过去几年中，对于度至少为 $\omega(\log n)$ 的图，已有多项改进，这些工作始于 Cohen、Peng 和 Wajc [24]；参见 [24, 40, 20, 39]及其中的参考文献。一般边到达情形下目前最好的界，是 Kulkarni、Liu、Sah、Sawhney 和 Tarnawski [39]提出的一种巧妙的 $\frac{e}{e-1}\Delta$着色算法。在任意边到达情形下，能否将颜色数量改进至 $(1+o(1))\Delta$，适用于 $\Delta=\omega(\log n)$，仍是一个重要的开放问题。

独立工作：在一项同期的独立工作中，Ghosh 和 Stoeckl [31] 也考虑了流式环境中的边着色。特别是，他们也达到了我们的推论 1.2 和定理 1.3 中的相同界限。在 [31] 中还给出了一种平滑的颜色/空间权衡。特别地，对于任意 $1\le s\le\sqrt{\Delta}$，他们实现了一个使用 $\widetilde{O}(ns)$ 空间的算法，该算法使用 $\widetilde{O}(\Delta^2/s^2)$ 种颜色为图着色。[3] 这可以与我们的定理 1.1 进行比较，后者使用 $O(\Delta^{1.5}/s)$ 种颜色和 $\widetilde{O}(ns)$ 空间。我们注意到，$\Delta^2/s^2\ge\Delta^{1.5}/s$ 对于 s 的整个范围都是如此。因此，我们的空间/颜色权衡从不使用更多颜色。但对于较小的 s，我们的算法使用更少的颜色。例如，在 $\widetilde{O}(n)$ 空间算法的极端情况下（即 $s=1$），我们的算法使用 $O(\Delta^{1.5})$ 种颜色，而 [31] 需要 $O(\Delta^2)$ 种颜色。最后，我们注意到，虽然本文只考虑了简单图和随机化算法，但 [31] 还将其算法扩展到了多重图，并对其进行了去随机化。

[3]在他们的论文中，这等价于表述为使用 $\widetilde{O}(n\sqrt{\Delta/t})$ 空间的 $O(\Delta t)$ 着色。

1.2 预备知识

除非另有说明，我们用 $G = (V, E)$ 表示输入图。我们分别用 $n := |V|$ 和$m := |E|$ 表示 G 中的顶点数和边数。我们用 Δ 表示图 G 的最大度。对于任意整数 k，我们用 $[k]$ 表示集合 $\{1, \ldots, k\}$。

图流模型：在标准图流模型中，任意图的边 G以任意顺序逐条到达。算法所需的空间远小于边的总数，只能对输入进行少数几遍（最好仅一遍）扫描，并应返回输出。对于边着色，输出大小与输入一样大，因此不可能存储整个输出后再一次性返回。因此，我们也允许算法以流式方式输出解。在文献中，这种模型也称为"W-流"模型 [32, 26]。本文介绍的所有算法都只对输入进行一遍扫描。我们以字数衡量空间，每个字由$\Theta(\log n)$ 比特组成。

我们针对一般边到达模型的算法，建立在为两种限制更严格的到达模型所开发的算法之上：一般顶点到达，以及二分图中的单侧顶点到达。下面给出这些标准模型的定义。

定义 1.4（顶点到达模型）。在此模型中，输入图的顶点 G 按照某个任意排列 π 逐个到达。某个顶点到达时，它与置换 π 中先前顶点相连的所有边也随之到达。

定义 1.5（二分图中的单侧顶点到达）。在该模型中，假设输入图 G 是一个二分图，其顶点集为 U 和 V。V 中的"离线"顶点从一开始就存在，而 U 中的"在线"顶点则按照任意顺序逐个到达。每当一个在线顶点 u 到达时，它与离线顶点 V 之间的所有边都会被揭示。

在我们的证明中，我们使用以下标准形式的切尔诺夫界。

命题 1.6（切尔诺夫界）。设 $X_1, ..., X_n$ 是 $[0, 1]$ 中的独立随机变量。设 $X = \sum_{i=1}^{n} X_i$ 和 $\mu = \mathbf{E}[X]$。那么，对于所有 $\delta \geq 0$ 和 $\mu' \geq \mu$，$\Pr[X \geq (1+\delta)\mu'] \leq \exp\left(-\frac{\delta^2}{2+\delta}\mu'\right)$.

2 技术概述

在本小节中，我们对我们的算法进行非正式的高层概述。

如前所述，解决流式边着色问题的主要挑战在于，算法无法"记住"分配给各条边的所有颜色，因为这会占用过多空间。事实证明，当度数的变化不均衡时，这尤其构成挑战。为传达本小节中的关键直觉，我们先聚焦于二分图中的单侧顶点到达模型（定义 1.5）。我们注意到，即使在这种受限到达模型中，文献中已知的最佳算法仍是 [16, 22, 4] 提出的算法，该算法需要 $O(\Delta^2)$ 种颜色，并占用 $O(n)$ 空间。这里，~我们将介绍如何将其改进为渐近最优的 $O(\Delta)$ 着色，且只需 $O(n)$ 空间。我们在更一般的边到达模型下给出的定理 1.1 中的最终算法，建立在这个顶点到达算法之上。

由于在线顶点的所有边会同时到达，因此确保它们获得不同颜色并不困难。难点在于，同时确保离线顶点的所有边

也获得不同颜色。为此，我们首先介绍一种算法，它使用 $O(\Delta \log\ n)$ 种颜色、$O(n\Delta)$ 的“预处理空间”和 $O(n)$ 的工作空间。随后，我们将展示如何将其转化为一种 $\tilde{}$ $O(\Delta)$ 着色算法，使其总共使用 $O(n)$ 的空间。

一种使用 $O(\Delta \log\ n)$ 种颜色但空间开销较大的算法：令 $K = \Theta(\Delta \log\ n)$ 为我们使用的颜色数量。在预处理步骤中（即尚未看到图中的任何边之前），我们为每个离线顶点 $v \in V$ 在内存中存储一个颜色 $\{1, \ldots, K\}$ 的随机排列 π_v，总共需要 $\tilde{}$ $O(n\Delta)$ 的预处理空间。现在假设第一个在线顶点 u 到达。对于它的每个离线邻居 v_i，我们考虑 v_i 的第一个随机颜色 $\pi_{v_i}(1)$。由于离线顶点的随机排列彼此独立，当 $\Delta \geq 2$ 我们为 v_i 选出的颜色与 u 的所有其他邻居 v_j 的 $\pi_{v_j}(1)$ 都不同时，概率为 $(1 - \frac{1}{K})^{\deg(u)} \geq (1 - \frac{1}{\Delta})^{\Delta} \geq 0.25$。注意，$(1 - \frac{1}{\Delta})^{\Delta}$ 是单调递增的，且当 $\Delta = 2$ 时其值为 0.25。若此情况发生，我们就将颜色 $\pi_{v_i}(1)$ 分配给边 (u, v_i)。如果 v_i 的第一种颜色并非唯一，我们就丢弃它，并揭示 v_i 的下一种颜色 $\pi_{v_i}(2)$。这一过程的每一轮都能成功为 u 中恒定比例的剩余未着色边着色。因此，以高概率只需 $O(\log n)$ 轮即可为 u 的所有边着色。另一方面，由于我们为每个离线顶点预留了 $K = \Theta(\Delta \log\ n)$ 种颜色，因此可以为它们的每条边揭示最多 $\Theta(\log n)$ 种颜色，从而以高概率为整个图完成着色。实现此算法时，我们只需为每个离线顶点 v 维护一个计数器 d_v，记录其随机排列中已有多少种颜色被揭示；总共只需 $O(n)$ 的工作内存。

$\tilde{}$减少空间：为了摆脱先前算法中 $O(n\Delta)$ 所需的巨大预处理空间，我们限制所需的随机性。为此，我们不再为每个离线顶点 v 选择 $\{1, \ldots, K\}$ 的完全随机置换 $\tilde{}$，因为这需要 $\Theta(nK) = \Theta(n\Delta)$ 的空间，而是只选取一个随机数 $r_v \in [K]$，并使用循环移位排列（r_v，$r_v + 1$，...，K，1，...，$r_v - 1$）。这样做的优点是，只需将随机数 r_v 存储在内存中，就能以 $O(1)$ 的空间存储这个循环移位排列。缺点是缺乏独立性会破坏上述分析。例如，如果 $\pi_{v_i}(1) = \pi_{v_j}(1)$，那么我们也有 $\pi_{v_i}(2) = \pi_{v_j}(2)$，因此无法论证每一轮都能为在线顶点 u 的边着色一个常数比例。为了解决这一问题，我们为每个离线顶点 v 选取 $t = \Theta(\log\ n)$ 个随机数 r_v^1，...，r_v^t，并考虑这些循环移位排列

$$[r_v^1, \ldots, \Delta, 1, \ldots, r_v^1 - 1], \ldots, [r_v^t, \ldots, \Delta, 1, \ldots, r_v^t - 1].$$

现在，如果根据第一个循环移位排列，v_i 和 v_j 的第一个颜色选择相同，我们就考虑第二个循环移位排列，然后是第三个，以此类推。这消除了为在线顶点的边提出的颜色之间的依赖关系，但离线顶点的边之间的依赖关系仍然存在。幸运的是，分析过程并不需要消除后者，因此我们$\tilde{}$可以用 O（n）空间实现该算法。

通过 k-出采样减少颜色数：上述算法是贪心的：我们先为在线顶点 u 的每条边揭示一个随机颜色，贪心地为那些提议颜色唯一的边着色，然后再揭示下一批提议。与这种最多需要 $O(\log\ n)$ 轮、不可避免地将颜色数量增加到 $O(\Delta \log\ n)$ 的贪心算法不同，我们先为在线顶点 u 的每条边 $e_i = (u, v_i)$ 抽取 3 个随机颜色 x_i^1, x_i^2, x_i^3，并同时考虑这 3 个颜色。我们证明，以概率 $1 - 1/\text{poly}(\Delta)$，可以为 u 的每条边 e_i 从颜色 x_i^1, x_i^2, x_i^3 中选取一个，使得 u 的所有边都获得不同颜色。这个定理的证明建立在一个新的引理（引理 3.6）之上；我们证明了该引理关于单侧随机 k-出模型中完美匹配存在性的结论。这个引理可能具有独立的.....

有趣的是，如果我们有一个随机二分图，其顶点集为 V 和 U，并且 V 中的每个顶点都与 k 个 U 中的顶点均匀地相邻，那么 G 在满足 $k \geq 3$ 和 $|U| > e|V|$ 的条件下，具有完美匹配。尽管最近有人使用 k-出采样来获得更高效的分布式图连通性 [33] 和最小割算法，[30, 8]，但据我们所知，这是它首次应用于图着色。

从顶点到达到边到达：现在概述我们如何从顶点到达模型转向更一般的边到达模型。为说明核心直觉，我们只关注定理 1.1 中较简单的特殊情形，其中 $s = 1$。也就是说，算法使用 $O(n)$ 空间，并返回一个 $O(\Delta^{1.5})$ 边着色。我们的第一步是将上述算法推广到非二部顶点到达模型（定义 1.4）。事实上，这一步大致可以由文献中一种已知的随机二分化技术得出，我们在附录 A 中介绍了该技术。接下来更具挑战性的一步，是将算法推广到批次到达设置：在每一步中，我们看到的不是某个顶点的所有边，而是它的若干条 $\Theta(\sqrt{\Delta})$ 边（且无法保证该顶点的下一批边何时到达）。实现这个更一般的算法后，我们运行它的一个实例 $\mathcal{A}$。为了将边输入这个批次到达算法 $\mathcal{A}$，我们持续将边存储在集合 H 中。每当 H 中的边数达到 n，我们就查看 H 中度最大的顶点 v。如果 $\deg_H(v) \geq \sqrt{\Delta}$，我们就将该顶点 v 的这些边中的 $\Theta(\sqrt{\Delta})$ 条作为下一批输入算法 $\mathcal{A}$，并将它们从 H 中移除。否则，H 中的最大度小于 $\sqrt{\Delta}$；此时，我们使用 $O(\sqrt{\Delta})$ 种颜色，对 H 中的所有边进行贪心边着色，并将它们全部从 H 中移除。每次进行贪心着色时，我们都会给图中的 n 条边着色。因此，这种情况至多发生 $m/n = O(\Delta)$ 次，总共需要 $O(\Delta^{1.5})$ 种颜色。对于较大的 s，定理 1.1 的证明需要以更复杂的白盒方式应用顶点到达算法；算法的详情见第 5 节。

3 顶点到达模型下的流式边着色

本小节首先证明定理1.3，其重述如下，适用于顶点到达模型下的流式边着色。我们在更一般的边到达模型中使用的算法，建立在本小节介绍的顶点到达算法之上。

定理 **1.3** （重述）。 存在一种随机化流式算法，能够在任意顶点到达顺序下对 $O(\Delta)$ 图进行边着色，所需空间为 $\widetilde{O}(n)$。

3.1 基本归约

我们首先介绍两种基本归约，它们实质上将顶点到达模型下一般图中的边着色问题，归约为单侧顶点到达模型下二分图中的同一问题。

下面的引理表明，我们可以聚焦于二分图，而非一般图。其思路是随机地将顶点集 V 划分为两个集合 A 和 B，对 G 中连接 A 与 B 的二分子图进行边着色 $G[A]$，并对诱导子图 $G[B]$ 递归处理。由于这是一个标准思路，且可由已知归约直接推出，我们将其证明放在附录 A 中。

引理 3.1。假设存在一种流式算法，能够使用 $s(n, \Delta)$ 字空间，以 $f(\Delta) \geq \Delta$ 种颜色对任意具有 n 个顶点、最大度为 Δ 的二分图进行边着色。那么存在一种流式算法，能够对任意一般的（即不一定是二分图的）具有 n 个顶点的图进行边着色，其最大度

为 Δ，使用

$$f(\Delta) + f(\Delta/2) + f(\Delta/4) + ... + 2f(1)$$

~种颜色和 $O(n + s(n, \Delta))$ 空间。如果用于二分图的算法适用于顶点到达（分别为边到达），那么用于一般图的算法也适用于顶点到达（分别为边到达）。

我们强调，要在顶点到达模式下处理一般图，必须确保我们输入引理 3.1 的二分图算法适用于双侧顶点到达。接下来的归约表明，任何单侧顶点到达算法也能处理双侧顶点到达，只是颜色数量会增加一倍。

声明 3.2。假设存在一种流式算法，能够在单侧顶点到达模式下为任意 n 个顶点、最大度为 Δ 的二分图进行边着色，使用 $f(\Delta)$ 种颜色。那么，可以将其转化为一种算法，在双侧顶点到达模式下为该二分图进行 $2f(\Delta)$ 着色，且所需空间渐近相同。

证明。设 $G = (V, U, E)$ 为一个在双侧顶点到达模式下到达的二分图。我们将其边划分为两个子图 G_U 和 G_V。若 V 中的一个顶点到达，就将其边加入 G_V；若 U 中的一个顶点到达，就将其边加入 G_U。这样，G_V（分别为 G_U）就是一个在单侧顶点到达模式下到达的二分图，其在线部分为顶点集 V（分别为 U）。由于 G_U 和 G_V 的最大度均不超过 Δ（因为它们是 G 的子图），我们可以分别在 G_U 上运行一个引理 3.3 的实例，并在 G_V 上并行运行另一个实例，使用互不相交的颜色。这样只会使颜色数量和空间都增加一倍，证明完毕。 □

3.2 算法

上一小节中的归约表明，相较于一般图，我们可以聚焦于单侧顶点到达的二分图。以下引理是第 3 小节其余部分的主要贡献。

引理 3.3。存在一种流式算法，在单侧顶点到达的情况下，可使用 $O(\Delta)$ 种颜色对任意最大度为 Δ 的二分图进行边着色，并使用 $O(n)$ 空间，且此结论以高概率成立。（着色始终有效，空间界以概率 $1 - 1/\text{poly}(n)$ 成立。）

我们先来看一下 如何引理 3.3 证明定理1.3 使用前一小节的归约 .

定理的证明 *1.3*。 首先，我们将引理3.3 的算法用于断言 3.2，从而得到一种流式算法，对于某个常数 C，$C(\Delta)$ 能够在双侧顶点到达的情况下为二分图着色，并使用 $O(n)$ 空间。然后，我们将此算法代入引理 3.1，从而得到一种适用于一般~图的算法。该算法使用 $O(n)$ 空间，并且根据引理 3.1，其颜色数为

$$C\Delta + C(\Delta/2) + C(\Delta/4) + \ldots + C \leq 2C\Delta = O(\Delta). \quad \square$$

因此，只剩下证明引理 3.3。本小节给出下文形式化描述的算法，即算法 1。我们在小节 3.3 分析该算法的空间复杂度，并在小节 3.4分析其正确性以及它所使用的数量。最后，我们在小节 3.5 证明该算法也可以推广到批次到达模型。

算法 1：适用于二分图中单侧顶点到达的流式边着色。

参数： $c := 2.72$

1. 对于每个离线顶点 v，从 $[c\Delta]$ 中均匀地抽取 3 个互不相同的随机数 r_v^1、r_v^2 和 r_v^3，并将它们存储在内存中。[a]

2. 此外，对于每个离线顶点 v，我们在内存中存储一个计数器 $\deg_v$，以便随着边的到达跟踪其度。

3. 当一个在线顶点 u到达时：

   (a) 对于每条边 $e_i = (u, v_i)$，考虑以下三种颜色 x_i^1、x_i^2、x_i^3：

$$\begin{aligned} x_i^1 &:= \left((r_{v_i}^1 + \deg_{v_i}) \bmod c\Delta\right), \\ x_i^2 &:= \left((r_{v_i}^2 + \deg_{v_i}) \bmod c\Delta\right) + c\Delta, \\ x_i^3 &:= \left((r_{v_i}^3 + \deg_{v_i}) \bmod c\Delta\right) + 2c\Delta. \end{aligned}$$

   (b) 如果存在一种边的颜色分配方式 u，使得每条边 e_i 都获得一个来自 $\{x_i^1, x_i^2, x_i^3\}$ 的颜色，并且 u 的所有边都获得不同的颜色，那么我们就分配这些颜色，将它们输出到数据流中，并从内存中移除 u 的边。否则，我们将 u 的所有边添加到集合 S 中，并将其存储在内存中。（在命题3.4 中，我们将证明 u的边被成功着色的概率为 $1-O(1/\Delta^5)$。

4. 只剩下为 S 中的边着色。注意，由于 S 是 G 的子图，其最大度不大于 Δ。因此，我们可以通过现有的二分图离线边着色算法，使用 Δ 个新颜色为其边着色；例如参见 [29]。

[a] 为简洁起见，在此处及后续分析中，我们假设 $c\Delta$ 是一个整数。如果不是，则必须将 $c\Delta$ 替换为 $\lceil c\Delta \rceil$。

3.3 空间复杂度

在算法 1中，对于每个顶点，我们都会保存它的度和三个随机数，这些信息总共可以用 $O(n)$ 个字存储。空间复杂度中唯一需要进行非平凡界定的部分，是我们存储在内存中并在最后着色的边集合 S 的大小。本小节的主要结果是证明集合 S 的大小为 $O(n)$，且以高概率成立。我们首先在断言 S 中利用与 k-出子图的联系（在引理 3.6 中形式化），界定 S 的期望大小。随后，我们在断言 3.7 中证明 S 大小的高概率界。

断言 3.4。任取一个在线顶点 u。我们存储 u 的边的概率，在算法 1 的步骤 3b 中至多为 $O(1/\Delta^5)$。根据期望的线性性，这意味着

$$\mathbf{E}\,|S| \le O(n/\Delta^4).$$

证明。证明此断言的主要思路是，在下面适当定义的“颜色图”上使用最大匹配问题。当在线顶点 u 到达输入时，我们按如下方式构造二分图 H_u。

定义 3.5（颜色图）。对于任意在线顶点 u，我们将颜色图 $H_u = (X, Y, E)$ 定义如下。集合 X 对应于 u 的边，即对于每条边 (u, v_i)，我们都有一个顶点

对应于 v_i 中的 X。对于范围 $[c\Delta]$ 内的每种颜色，我们在集合Y中都有一个对应的顶点。 X 中的每个顶点都有恰好三条边连接到 Y，分别对应三种颜色：

$$y_i^1 := \left((r_{v_i}^1 + \deg_{v_i}) \bmod c\Delta\right),$$
$$y_i^2 := \left((r_{v_i}^2 + \deg_{v_i}) \bmod c\Delta\right),$$
$$y_i^3 := \left((r_{v_i}^3 + \deg_{v_i}) \bmod c\Delta\right),$$

其中 v_i是与 u相邻的任意离线顶点。注意，由于在算法1中， $r_{v_i}^1$、 $r_{v_i}^2$和$r_{v_i}^3$彼此不同，因此三种颜色 y_i^1、 y_i^2和y_i^3也将彼此不同。

我们首先说明， H_u 中的完美匹配对应于 u所有边的有效边着色。设 M 是 H_u中的一个完美匹配。假设与边 e_i 对应的节点在 M 中匹配到颜色y_i^j （对所有 e_i而言，该颜色各不相同）。在这种情况下，我们将颜色x_i^j 赋予 e_i ，其中x_j^i 的定义见算法1。要看出 u 中的所有边都获得不同颜色，只需注意：

$$y_i^j = x_i^j \bmod c\Delta.$$

此外，所有匹配的 y_i^j 都是唯一的，因为 M 是一个匹配。因此，如果我们在颜色图 H_u 中找到一个完美匹配，就可以利用该匹配轻松地为与 u 关联的边着色。

因此，接下来只需证明 H_u 确实具有完美匹配（且其概率足够大）。为此，我们使用以下关于所谓3.6 -出模型中完美匹配存在性的引理 k。引理3.6 证明，如果二部图一侧的每个顶点都与另一侧的 3 个随机顶点相邻，那么只要另一侧足够大，该图就具有完美匹配。根据 Frieze [28] 于 1986 年得到的经典结果，如果二部图两侧的顶点都选择 k 条随机边，那么当且仅当 $k \geq 2$时，该图以高概率具有完美匹配。然而，当只有图的一侧选择随机边时，结论并不成立；以下引理的重点正是这种情况3.6。

引理 3.6. 考虑一个随机二分图 G ，其顶点集为 V 和 U ，其中每个顶点 $v \in V$ 都恰好与 U 中的 3 个不同顶点相邻，这些顶点从 U 的所有大小为 3 的子集中均匀地选取，且各顶点的选择彼此独立。如果对于固定的 $c > e$， $|V| \leq n$ 和 $|U| = cn$，则图 G 至少以 $1 - O(1/n^5)$ 的概率存在完美匹配。

证明。首先，我们论证可以不失一般性地假设 $|V| = n$。因为如果我们有 $|V| < n$ ，就可以向 V 一侧添加一些虚拟顶点，并为它们随机选择邻居 U。这个新图中的任何完美匹配，去掉其中的虚拟边后，都能得到原始图中的一个完美匹配。因此，在该图中找到完美匹配的概率不低于 $|V| = n$ 的情况。

对于某个固定的 $\varepsilon > 0$，我们令 $c = (1 + \varepsilon)e$。我们假设 n 大于任何所需的常数（可能是 ε 的函数）。注意，如果 n 固定，那么通过调整常数，我们可以确保$1 - O(1/n^5) = 0$，从而使引理中的概率陈述变得显然。

根据 Hall 定理， G 存在完美匹配，当且仅当对于每个子集 $S \subseteq V$，都有 $|N_G(S)| \geq |S|$。由于 V 中的每个顶点的度恰好为 3，因此 $|S| \leq 3$ 时这一条件必然成立。所以，只需证明对于所有大小至少为 4 的集合，这一条件以高概率成立。令 $4 \leq k \leq n$固定。顶点 $v \in V$的所有边都连向 U 中一个大小至多为 $k - 1$ 的固定子集 U 的概率恰好为 $\binom{k-1}{3} / \binom{|U|}{3}$。因此，我们有

$$\Pr[\text{exists } S \subseteq V : |S| = k \text{ and } |N(S)| < k] \leq \binom{n}{k} \cdot \binom{|U|}{k-1} \cdot \left(\frac{\binom{k-1}{3}}{\binom{|U|}{3}}\right)^k$$

$$\leq \left(\frac{ne}{k}\right)^k \cdot \left(\frac{cne}{k-1}\right)^{k-1} \cdot \left(\frac{(k-1)(k-2)(k-3)}{cn(cn-1)(cn-2)}\right)^k$$

(since $\binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$ for all n and k)

$$\leq \left(\frac{ne}{k}\right)^k \cdot \left(\frac{cne}{k-1}\right)^{k-1} \cdot \left(\frac{k-1}{cn/(1+0.1\varepsilon)}\right)^{3k}$$

（此处我们使用 $cn/(1+0.1\varepsilon) \leq cn-2$，该式在 n 大于某个常数时成立。）

$$= \frac{(1+0.1\varepsilon)^{3k} e^{2k-1} (k-1)^{2k+1}}{c^{2k+1} \cdot k^k \cdot n^{k+1}}$$

$$\leq \left(\frac{(1+0.5\varepsilon)e}{c}\right)^{2k-1} \cdot \left(\frac{k}{n}\right)^{k+1}$$

(Here we use $(1+0.1\varepsilon)^2 \leq 1+0.5\varepsilon$ for small enough ε)

$$\leq \left(\frac{1+0.5\varepsilon}{1+\varepsilon}\right)^{2k-1} \left(\frac{k}{n}\right)^{k+1}. \qquad \text{(Since } c = (1+\varepsilon)e.\text{)}$$

要完成证明，请注意，如前所述，对于 $k \leq 3$，Hall 保证是确定成立的。对于 $k=4$，上述不等式的上界为 $(k/n)^{k+1} = O(1/n^5)$，因此，V中所有大小为 $k=4$ 的子集，在 U 中至少有 4 个邻居的概率至少为 $1-O(1/n^5)$。对于 $5 \leq k \leq n^{0.1}$的每一种选择，上述不等式的上界为 $(k/n)^{k+1} \leq n^{-0.9\times 6} = n^{-5.4}$；对所有 $O(n^{0.1})$ 种这样的 k 选择应用并集界可得，以至少 $1-O(n^{-5.3})$ 的概率，V中任意大小为 $5 \leq k \leq n^{0.1}$ 的集合在 U 中也至少有 k 个邻居。最后，对于 $n^{0.1} < k \leq n$的情况，上述不等式的上界为

$$\left(\frac{1+0.5\varepsilon}{1+\varepsilon}\right)^{2k-1} \left(\frac{k}{n}\right)^{k+1} \leq \left(\frac{1+0.5\varepsilon}{1+\varepsilon}\right)^{2k-1} = 2^{-O_\varepsilon(k)} = 2^{-O_\varepsilon(n^{0.1})} \ll O(1/n^6).$$

因此，再次对所有此类 k 选择应用并集界可得，以至少 $1-O(1/n^5)$ 的概率，V 中任意大小为 $n^{0.1} < k \leq n$ 的集合在 U 中也至少有 k 个邻居。综合 k 的所有这些情况可得，以至少 $1-O(1/n^5)$ 的概率，Hall 条件成立，且 G 存在完美匹配。

□

现在注意到，颜色图 H_u 满足引理 3.6 的条件。与 u 相连的边至多有 Δ 条，因此 $|X| \leq \Delta$。另一方面，$|Y| \geq c\Delta$。此外，对于每条边 $e_i = (u, v_i)$，按照定义 3.5 的规定，其在 X 中对应的顶点会与分别对应于颜色 y_i^1、y_i^2、y_i^3 的顶点相连。由于各个 $r_{v_i}^j$ 彼此不同且均匀分布，X 中的每个顶点都会与图 H_u 中 Y 的 3 个彼此不同且均匀分布的顶点相连，因此我们可以应用引理 3.6。这意味着，H_u 以至少 $1-O(1/\Delta^5)$ 的概率存在完美匹配。这表示我们将每个在线顶点的边存入 S 的概率至多为 $O(1/\Delta^5)$，从而完成断言 3.4 的证明。

□

值得注意的是，为了在 H_u 中找到最大匹配，我们使用了 [34] 的算法，其空间复杂度为 $O(|V|+|E|)$，在这里即为 $O(\Delta)$。在看到下一个在线顶点之前，我们可以删除这部分内存。

虽然命题 3.4 界定了算法 1 的期望空间，但接下来的命题 3.7 以高概率将空间界定为 $O(n)$。

断言 3.7。 算法 *1* 使用 $O(n)$ 空间，概率为 $1-1/\text{poly}(n)$。

证明。对于在线顶点 u，定义 X_u 为指示随机变量，当且仅当我们将 u 的边存储在 S 中时，X_u 为真。此外，定义 $X = \sum_{u \in U} X_u$。我们知道，

$$\Pr[|S| > O(n)] \leq \Pr[\Delta \cdot X > O(n)] = \Pr[X > O(n/\Delta)].$$

我们知道 $\mathbf{E}[X] \leq n \cdot O(1/\Delta^5)$ 根据命题 3.4。然而，由于这些 X_u 并非 彼此独立，我们无法使用切尔诺夫界来为它们的和提供集中界。要理解其中的原因，考虑两个在线顶点 u_1，u_2 它们拥有相同的离线邻居集合，并且在输入中前后相继出现。如果 $X_{u_1} = 1$ 那么 $X_{u_2} = 1$ 也成立，因为用于 H_{u_2} 的颜色与用于 H_{u_1} 的颜色完全相同，只是偏移了一位。

为了界定 X，我们的计划是将所有 X_u划分为若干组，使得每组中的所有随机变量彼此独立，然后分别对每组应用切尔诺夫界。我们知道，如果两个在线顶点 u 和 w 没有任何共同的离线邻居，那么 X_u 和 X_w彼此独立。这是因为算法 1 的步骤 3a 中指定的颜色，都是在线顶点邻居集合上的随机性的函数。因此，如果这些集合不相交，那么X_u 和 X_w 彼此独立。

我们将这些变量之间的依赖图定义如下：该依赖图的顶点集为 $\{X_u : u \in U\}$，当且仅当 u 和 w 共享一个离线邻居时，X_u、X_w 与 u、$w \in U$ 之间存在一条边。由于每个顶点 $u \in U$ 至多有 Δ 个离线邻居，而每个离线邻居至多有 Δ 个在线邻居，因此该图的最大度至多为 Δ^2。

由于最大度为 Δ' 的任意图都可以通过简单的顺序贪心算法，用至多 种颜色为顶点 着色，依赖图可以用至多 $\Delta' + 1$ 种颜色为顶点着色。注意，这样做之后，根据依赖图的定义，每个颜色类中的顶点对应的随机变量相互独立。因此，我们可以对每个颜色类中各项之和应用切尔诺夫界。稍微滥用一下记号，设 $\Delta^2 + 1$，..., X_1 是某个颜色类中的一组变量。定义 X_t E μ =。由于 E($[X_i] = O$)（根据命题 $1/\Delta^5$3.4），由期望的线性性可得，对某个常数 ，

$$\mu = O(t/\Delta^5). \tag{1}$$

应用切尔诺夫界可得，对于任意 $\delta > 0$ 和 $\mu' \geq \mu$,

$$\Pr[X_1 + X_2 + ... + X_t \geq (1 + \delta)\mu'] \leq \exp\left(\frac{-\delta^2}{2 + \delta}\mu'\right). \tag{2}$$

若 $t \leq n/\Delta^3$，则称该颜色类为小颜色类；否则称为大颜色类。对于小组，我们无需证明任何集中界，因为属于小组的顶点总数为 $(\Delta^2 + 1)(n/\Delta^3) = O(n/\Delta)$，且每个顶点在 S 中存储 Δ 条边，因此 S 中此类边的数量有一个确定性上界 $O(n)$。因此，只需分析大组。

为了处理大组，我们针对 Δ 考虑两种情况。

情况 1: $\Delta = O((n/\log n)^{1/4})$。将 $\delta = \Delta^4$ 代入 (2)，得到:

$$\Pr\left[\sum_{i=1}^{t} X_i \geq (1 + \Delta^4)\mu\right] \leq \exp\left(\frac{-\Delta^8}{2 + \Delta^4}\mu\right).$$

令 $\mu' = C \cdot t/\Delta^5$。注意 $\mu \leq C \cdot t/\Delta^5 = \mu'$

且 $t \geq \frac{n}{\Delta^3}$ 由于我们只考虑大型组，因此得到，

$$\exp\left(\frac{-\Delta^8}{2+\Delta^4}\mu'\right)_{1/4} \leq \exp\left(\frac{-\Delta^8}{2+\Delta^4}\cdot\frac{Cn}{\Delta^8}\right) = \exp\left(\frac{-Cn}{2+\Delta^4}\right).$$

由于我们有 $\Delta \leq O\left(\frac{n}{\log n}\right)$，因此最后一项可以上界为 n^{-10}。

令 t_1、t_2、...、t_k 表示各组的大小。我们知道 $k \leq n$；因此，对k应用并集界，某一组的值偏离其均值超过 Δ^4 倍的概率至多为 n^{-9}。因此，以至少 $1-n^{-9}$ 的概率，所有组的值偏离其均值都小于 Δ^4 倍。在这种情况下，注意到 $\sum_i t_i = n$，我们得到

$$X \leq \Delta^4 \cdot \sum_i \frac{C \cdot t_i}{\Delta^5} = O(n/\Delta).$$

回顾一下， X 是将边存储在 S 中的在线顶点总数，由此可得， S 中的边总数以高概率至多为 $\Delta \cdot O(n/\Delta) = O(n)$。

情况 2： $\Delta = \omega((n/\log n)^{1/4})$。注意，将任何 u 条边存储在 S 中的概率为 $n^{-5/4+o(1)}$。对所有 n 个在线顶点应用并集界后，以 $O(n^{-0.25+o(1)})$ 的概率，没有边会被存储在 S 中。

□

3.4 正确性与颜色数量

在本小节中，我们将讨论为什么在我们的算法中，任意两条相邻的边都会获得不同的颜色。我们可以忽略在算法 1 的步骤 4 中着色的边，因为它们使用的颜色集合与其余边完全不同。因此，在本小节余下部分，我们将重点讨论其余的边。

对于在线顶点 u，通过步骤 3b，与其相连的边都会获得不同的颜色（如果无法做到这一点，我们就将 u 的边存入 S；如前所述，这些边可以忽略）。因此，剩下要证明的是离线顶点之间不存在冲突。

对于离线顶点 v，首先注意，对于任意边 e_i:

$$x_i^1 \in [0, c\Delta), \qquad x_i^2 \in [c\Delta, 2c\Delta), \qquad x_i^3 \in [2c\Delta, 3c\Delta).$$

由于这些范围互不相交，若两条边获得相同的颜色，它们必定属于同一个范围。

注意，在算法运行过程中，每当我们看到一条与 v 相连的边，$\deg_v$ 就会增加一。设 $x_i^1 = x_j^1$ 为两条属于 v 的边，其中 e_i 先于 e_j 到达。定义 α 为边 e_i 到达时 $\deg_v$ 的值，定义 β 为边 e_j 到达时 $\deg_v$ 的值。于是我们得到：

$$\big((r_v^1 + \alpha) \bmod c\Delta\big) = \big((r_v^1 + \beta) \bmod c\Delta\big),$$

这等价于 $\alpha - \beta \bmod c\Delta = 0$。由于 $|\alpha - \beta| \leq \Delta$ 和 $c > 1$，这与假设矛盾，因此我们不可能有 $x_i^1 \neq x_j^1$。同样的结论也适用于第二和第三个颜色范围。因此，离线顶点的所有边也都被赋予了不同的颜色，我们的算法找到的着色是有效的边着色。

最后，可以从算法 1 验证所用颜色的数量为 $3c\Delta + \Delta = O(\Delta)$。本小节与上一小节共同完成了引理 3.3 的证明；如前所述，这也完成了定理 1.3的证明。

3.5 推广到批次到达

我们的算法 1 考虑顶点到达模型，即在线顶点的所有边同时到达。在这里，我们考虑一种更一般的批次到达模型，该模型在边到达和顶点到达之间进行插值。我们证明，我们的算法1可以推广到此模型。这种推广在我们第5小节

定义 3.8（单侧批次到达模型）。在该模型中，我们有一个二分图 $G=(U,V,E)$，其中包含在线顶点 U 和离线顶点 V。对于问题的参数 k，边以批次的形式到达，每个批次包含 k 条边，且同一批次中的所有边都与同一个在线顶点 u 关联。单个在线顶点 v 的边最多可以分为 Δ/k 个不同的批次到达。重要的是，我们无法保证同一个在线顶点的不同批次会连续到达（否则，该模型将等价于顶点到达模型）。我们使用 $B_i(u)$ 表示在线顶点 u 的第 i 个批次。

我们证明，算法 1 在单侧批次到达模型中可以得到以下界。注意，令 $k=\Delta$ 即可恢复定理 1.3，因此这只是一个更一般的结果。

引理 3.9。在单侧批次到达模型中，对于大小为 k 的批次，存在一种流式算法，使用 $O(n)$ 空间，并以高概率输出一个正确的 $O(\Delta^2/k)$ 边着色。

证明。注意，算法 1 中唯一用到“在线顶点的所有边同时到达”这一假设的部分是步骤 3b。在该步骤中，每条边 e_i of u的离线顶点都会为 e_i提出三个颜色 $\{x_i^1,x_i^2,x_i^3\}$。承诺为每条边 e_i 从 $\{x_i^1,x_i^2,x_i^3\}$ 中选择一种颜色，就已确保离线顶点的边之间不会发生冲突。另一方面，引理 3.6 保证，对于每个在线顶点 u，以概率 $1-O(1/\Delta^5)$，其所有边都存在一种使用各自提议颜色的合法着色。然而，要找到这种合法着色，关键是同时获得该在线顶点的所有边。如果像本引理所述的情况那样，在线顶点的边分批到达，那么我们就无法确保同一在线顶点 u 的不同批次中的边获得不同颜色。为解决这个问题，我们先仅使用各自提议的颜色，对在线顶点每个批次中的边分别进行合法着色。如果这使边 e被分配到颜色 c，那么我们为 e 报告的最终颜色为 (c,i)，其中 i 表示边 e 所属的批次 u。由于每个顶点至多有 Δ/k 个批次，采用这一方法所需的颜色总数为 $O(\Delta\cdot\Delta/k)=O(\Delta^2/k)$。最后请注意，空间仍为 $O(n)$，因为我们只需额外记录每个顶点已到达的批次数，这可以用 $O(n)$个计数器存储，每个在线顶点对应一个计数器。

□

4 边到达： $O(\Delta)$ 边着色，使用 $\widetilde{O}(n\sqrt{\Delta})$ 空间

在本小节中，我们证明以下特殊情形的定理 1.1：当 $s=\sqrt{\Delta}$ 时。也就是说，我们使用 $O(\Delta)$ 空间实现渐近最优的 $\widetilde{O}(n\sqrt{\Delta})$ 边着色。稍后，我们将在小节 1.1 中证明适用于 s 完整范围的定理 5。

为解决边到达模型，我们将其归约为定义 3.8中的单侧批次到达模型。在本小节中，我们只考虑批次大小为 $k:=\lceil\sqrt{\Delta}\rceil$ 的批次到达模型。

命题 4.1. 假设存在一个流式算法 $\mathcal{A}$，在单侧批次到达且批次大小为 $k=$ 的情况下，使用 $O(\Delta)$ 种颜色对最大度为 Δ 的二分图进行边着色

$\Theta(\sqrt{\Delta})$ 使用空间 $\widetilde{O}(n\sqrt{\Delta})$。那么存在一个流式边着色 $\mathcal{A}'$ 算法，在边到达的情况下，使用 $\widetilde{O}(n\sqrt{\Delta})$ 的空间对任意最大度为 Δ 的一般图进行 $O(\Delta)$ 边着色。

证明。我们首先假设图 G 是二部图，其顶点集为 U 和 V，但其边以任意顺序到达。我们从一个缓冲区 $T = \emptyset$ 开始，并将数据流中到达的任意边添加到 T 中。每当存在某个顶点 u，使得 T 中至少有 k 条边与其相连时，我们就从 T 中移除 u 的这些边，并将它们输入批到达算法 $\mathcal{A}$。这样，每个批次都恰好包含 k 条边，且这些边都与同一个顶点相邻。不过，该顶点可能属于 U 或 V。也就是说，这不是单侧到达，而是双侧到达。通过声明 3.2，可直接将其归约为单侧到达，但最终的颜色数量会增加一倍。最后，如果数据流结束时 T 中仍有剩余边，我们就使用离线算法通过 Δ 种颜色为它们全部着色。注意，由于在任意时刻，T 中的每个顶点的度最多为 k，因此该归约所需的空间仅为 $O(nk) = O(n\sqrt{\Delta})$，正如所要求的那样。

根据上述归约可知，在边到达模型下，若图是二分图，则存在一种算法 $\mathcal{A}''$，能够为图进行边着色，使用 $C(\Delta)$ 种颜色，其中某个常数为 C，并且使用 $\widetilde{O}(n\sqrt{\Delta})$ 空间。现在应用引理 3.1。由此可得一种针对边到达模型下一般图的算法，使用 $\widetilde{O}(n\sqrt{\Delta})$ 空间，且使用的颜色数量至多为

$$C\Delta + C\Delta/2 + C\Delta/4 + \ldots + C \leq 2C\Delta = O(\Delta). \quad \square$$

本小节接下来的目标是证明以下算法，将其与命题 4.1结合即可证明定理 1.1 对于 $s = \sqrt{\Delta}$:

引理 4.2。 存在一种流式算法，在单侧批到达（定义 *3.8*）模型下，对最大度为 Δ 、批次大小为 $k = \Theta(\sqrt{\Delta})$ 的任意二分图进行边着色，使用 $O(\Delta)$ 种颜色和 $\widetilde{O}(n\sqrt{\Delta})$ 空间。

证明。 或许有人会想，我们现在是否可以直接应用引理 3.9 中的朴素算法来证明引理 4.2。这行不通，因为引理 3.9 需要使用 $O(\Delta^2/k) = O(\Delta^{1.5})$ 种颜色，这远多于我们期望的 $O(\Delta)$ 种颜色。因此我们需要更多思路。

首先，我们假设 $\Delta \geq 300\log^2 n$。否则，整个图至多有 $O(n\Delta) = \widetilde{O}(n)$ 条边，因此我们可以将它们全部存储起来，并运行离线边着色算法。

考虑子图 $H_1, H_2, \ldots, H_k$，其中

$$H_i := \bigcup_{u \in U} B_i(u),$$

回想一下，$B_i(u)$ 是在线顶点 u 的第 i 个批次。换言之，H_i 是所有在线顶点的第 i 个批次所构成的集合。注意，每个子图 H_i 中边的到达顺序都遵循顶点到达模型。也就是说，对于每个在线顶点 v，它的所有边都在同一时间到达。还需注意，每个子图 H_i 在线侧的最大度至多为 k。现在，如果每个子图 H_i 的离线侧顶点的度也具有相同的上界 k，那么我们就可以针对 k 个子图 H_i 中的每一个，并行运行 $\Delta/k = O(\sqrt{\Delta})$ 个算法 1 的实例。该算法适用于顶点到达模型，并且每个实例使用互不相交的颜色调色板。这样一来，使用的颜色总数将为 $\frac{\Delta}{k} \cdot O(k) = O(\Delta)$，而空间复杂度将为 $\frac{\Delta}{k} \cdot O(n) = O(n\sqrt{\Delta})$。然而，不幸的是，每个 H_i 中的离线顶点的度可能高达 Δ。这是因为某个离线顶点的所有边都可能出现在其对应在线顶点的第一批次中，从而使得这些边.....

H_1 中的最大度等于 Δ，而不是 k。这是我们在证明其余部分中克服的主要障碍。

让我们在算法开始时，为每个在线顶点 u 均匀且相互独立地选择一个随机数 $b_u \in [k]$。现在，对于 u 的第 i 个批次，将其置换后的批次编号 $\rho_u(i)$ 定义为

$$\rho_u(i) := (i + b_u) \bmod k.$$

这样，我们现在将 $i \in [k]$ 的 H_i 重新定义为

$$H_i := \bigcup_{u \in U} B_{\rho_u^{-1}(i)}(u).$$

换言之，H_i 现在是这样一个图：对于每个在线顶点 u，包含其第 j 个批次中的边，且满足 $\rho_u(j) = i$。由于每个在线顶点 u 的批次数量至多为 $\Delta/k \le k$，因此 u的所有批次都会获得不同的批次编号。接下来需要界定每个 H_i中离线顶点的度。下面的引理断言，以高概率，该度实际上为 $O(k)$。

引理 4.3. 所有 H_i的最大度至多为 $2k$，其概率为 $1 - 1/\text{poly}(n$).

证明。由于对称性，我们只需关注界定 H_1中给定离线顶点 v 的度。设 v 的邻居为 u_1、u_2、...、u_l，其中每个 u_i 都是在线顶点。定义 X_i 为指示随机变量：当且仅当边 (v, u_i) 获得的批次编号为 1 时，$X_i = 1$。v 在H_1 中的度可以写成

$$X = \sum_{i=1}^{l} X_i.$$

由于 $l \le \Delta$ 且 $\mathbf{E}[X_i] = \frac{1}{k}$，根据期望的线性性，我们得到

$$\mathbf{E}[X] = \sum_{i=1}^{l} \mathbf{E}[X_i] \le \frac{\Delta}{k} \le k.$$

注意，所有 b_{u_i} 都在算法开始时均匀随机地选取。因此，任意给定的边的批次编号为 1 的概率是 $\frac{1}{k}$。由于所有 b_{u_i} 彼此独立，因此所有 X_i 也相互独立。于是，我们可以对 $\delta = 1$ 应用切尔诺夫界，并且有 $\Delta \ge 300 \log^2 n$（如引理 4.2 的证明开头所假设），我们得到，

$$\Pr[\text{degree } v \text{ in } H_1 \ge 2k] = \Pr\left[\sum_{i=1}^{l} X_i \ge 2k\right] \le \exp\left(\frac{-k}{3}\right) \le n^{-10}$$

对所有 n 个 v 选择以及所有 k 个 H_i 选择应用并集界，我们得到

$$\Pr[\text{exists } v \in V \text{ and } i \in [k] \text{ such that degree } v \text{ in } H_i \ge 2k] \le n^{-8}. \quad \square$$

现在，每个 H_i 的离线顶点和在线顶点的度都至多为 $2k$，我们可以遵循前面概述的方法，实现一个使用 $O(\Delta)$ 种颜色的着色算法，且空间复杂度为 $O(n\sqrt{\Delta}$。最终算法形式化如下，即算法 2。

算法 2：一种适用于单侧批次到达、批次大小为 $k = \Theta(\sqrt{\Delta})$ 的二分图流式边着色算法。该算法使用 $O(\Delta)$ 种颜色，占用 $O(n\sqrt{\Delta})$ 空间。这就是引理 4.2 中使用的算法；将其代入断言 4.1 即可证明定理 1.1 适用于 $s = \sqrt{\Delta}$。

参数： $k := \sqrt{\Delta}$， $c := 300$ （引理 4.3）

1. 如果 $\Delta \leq c \cdot \log^2 n$，我们就读取数据流中的所有边，并用 Δ 种颜色为它们着色。因此，在算法的其余部分中，假设 $\Delta > c \cdot \log^2 n$。

2. 随着边 G 到达，我们将它们分解为 k 个子图 H_1、……、H_k，并在这些子图上并行运行 k 个在线顶点算法 1 实例，在各实例中使用互不相交的颜色，总空间为 $O(nk)$。每个图 H_i 的最大度为 k，因此总共使用 $O(k \cdot k) = O(\Delta)$ 种颜色。

3. 对于每个在线顶点 u，均匀且独立地选择一个随机数 $b_u \in [k]$。

4. 对于每个在线顶点 u，存储一个计数器 I_u。它将记录任意时刻已看到的 u 批次数量。初始时，我们有 $I_u \leftarrow 0$。

5. 对于在线顶点 u，当大小为 k 的批次到达时：

   (a) $I_u \leftarrow I_u + 1$。(b) 将 u 的边添加到图 H_x 中，其中

   $$x := (I_u + b_u) \bmod k.$$

   (c) 这些将是 u 在图 H_x中的唯一边，因此这确实是一个在单侧顶点到达模式下的二分图，所以我们可以在图1 上运行算法的实例H_x，为其中的 u 边着色。

空间复杂度：由于我们在算法 2 中运行 $k = O(\sqrt{\Delta})$ 个算法 1 实例，因此保存所有这些实例所需的空间也是 $O(n\sqrt{\Delta})$，因为根据断言 3.7，每个实例占用 $O(n)$ 个字。此外，请注意，T 的大小不会超过 nk，否则其中必定存在一个度至少为 k 的顶点。我们还至多使用 $O(n)$ 的空间来存储计数器和随机数（在步骤 3 和 4 中指定）。因此，算法 2 使用的总空间为 $O(nk) = O(n\sqrt{\Delta})$。

颜色数量：对于算法 2所用的颜色数量，注意我们运行 k 个算法 1的实例，并且根据引理 $2k$，每个实例的最大度以高概率为 4.3。由于算法 1 使用与最大度成线性关系的颜色数量，因此颜色总数为 $O(k \cdot k) = O(\Delta)$，且该结论以高概率成立。

这就完成了引理 4.2的证明。 □

如前所述，将引理 4.2 代入断言 4.1 即可完成定理 1.1 关于$s = \sqrt{\Delta}$的证明，从而结束本节。

~5 边到达：使用 $O(ns)$ 空间进行 $O(\Delta^{1.5}/s+\Delta)$ 边着色

在第 4 节中，我们证明了定理 1.1 在 $s=\sqrt{\Delta}$这一特殊情形下成立。本节中，我们将更一般地证明定理 1.1 对 s的全部取值范围都成立。根据定理 1.1 的颜色/空间权衡，不失一般性，我们可以假设 $1\le s\le\sqrt{\Delta}$。为此，只需注意到，增大 s会增加算法的空间使用量 $\widetilde{O}(ns)$，但相应地会减少其使用的颜色数量 $O(\Delta^{1.5}/s+\Delta)$。然而，当 $s>\sqrt{\Delta}$时，$(\Delta^{1.5}/s+\Delta)$中的第二项会大于第一项，因此继续增大 s （从而增加空间）就没有意义了。在假设$1\le s\le\sqrt{\Delta}$成立的情况下，我们证明该图可以用 $O(\Delta^{1.5}/s)$种颜色进行边着色，并使用 $\widetilde{O}(ns)$的空间。

与小节 4类似，我们从归约到批次到达开始。不过，这里的归约略有不同，也更为一般。

断言 5.1. 假设存在一个流式算法 $\mathcal{A}$，它在单侧批次到达模式下，对最大度为 Δ 的二部图进行边着色，使用 $O(\Delta^{1.5}/s)$ 种颜色，每个批次大小为 $k=\Theta(\sqrt{\Delta})$，所需空间为 $\widetilde{O}(ns)$。那么存在一个流式边着色 $\mathcal{A}'$ 算法 $O(\Delta^{1.5}/s)$，能够在边到达模式下对任意最大度为 Δ 的一般图进行~边着色，所需空间为 $O(ns)$。

证明。 我们先假设图 G 是一个二部图，其顶点集为 U 和 V，但其边以任意顺序到达。与断言 4.1 小节的证明类似，我们将边缓冲在集合 T中。不过，我们不允许 T 的大小达到 nk （针对 k）：$=\lceil\sqrt{\Delta}\rceil$ 因为当 $s\ll k$时，这个大小过大，所以我们确保 $|T|\le ns$ 始终成立。为此，每当 $|T|$ 达到 ns时，我们就考虑以下两种情况：

情况 1：每个顶点在 T 中的边数少于 k 。在这种情况下，我们使用离线算法，给 T 中的所有边着色，使用 k 个新颜色，并将它们从内存中删除。我们声称，总共只需要 $O(\Delta^{1.5}/s)$ 个颜色。为此，请注意，每当发生这种情况时，我们都会从 T 中移除图的 ns 条边。因此，总共最多重复此过程 m/ns 次，其中 m 是原始图中的边数。由于 $m\le n\Delta$，这至多为 $n\Delta/ns=\Delta/s$。每次我们使用 k 个新颜色，因此总共使用的颜色数量为 $O(k\cdot\Delta/s)=O(\Delta^{1.5}/s)$。

情况 2：存在一个顶点 u，它在 k 中有 T。 在这种情况下，我们将 k 条边作为一个批次输入到 u 中的批到达算法 $\mathcal{A}$。请注意， u 可能同时属于 U 和 V。不过，使用命题 3.2 的归约，我们可以将其转换为单侧批次到达，代价是将颜色数量加倍。

由上述归约可知，在边到达模式下，存在一种算法 $\mathcal{A}''$，它为图进行边着色，使用 $f(\Delta)=C(\Delta^{1.5}/s)$种颜色，其中某个常数 C，且 $\widetilde{O}(ns)$空间，前提是该图为二部图。现在应用引理3.1。由此可得一种适用于一般图的算法~，在边到达模式下使用 $O(ns)$空间，且所用颜色数量至多为

$$f(\Delta)+f(\Delta/2)+f(\Delta/4)+\ldots+f(1).$$

由于 $s\le\sqrt{\Delta}$, $f(\Delta)$ 至少与 Δ 呈线性关系，因此该和的上界为 $O(f(\Delta))$ $O\ \Delta^{1.5}/s$ $=$ □( )。

本小节接下来的目标是证明以下算法，将其与断言 5.1结合即可证明定理1.1。

引理 5.2。 对于 $1 \le s \le \sqrt{\Delta}$，存在一种流式算法，可在单侧批到达（定义 3.8）的条件下，对最大度为 Δ 的任意二分图进行边着色，其中每个批次的大小为 $k = \Theta(\sqrt{\Delta})$，使用 $O(\Delta^{1.5}/s)$ 种颜色和 $\widetilde{O}(ns)$ 空间。

证明。我们假设 $\Delta \ge 900\log^2 n$。否则，我们可以存储整个图，并使用 $O(n)$ 空间对其进行边着色。

由于边以大小为 k 的批次到达，每个顶点最多有 Δ/k 个批次。对小节 4 中引理 4.2 的算法进行简单的推广，需要运行 $\Delta/k = O(\sqrt{\Delta})$ 个顶点到达算法的并行实例。然而，由于每个实例需要 $O(n)$ 的空间，因此总计需要 $O(n\sqrt{\Delta})$ 的空间，这大于当 $O(ns)$ 时所需的 $s \ll \sqrt{\Delta}$。

为了解决上述问题，我们将每个在线顶点 u 的批次划分为组。具体而言，我们将 u 的前 $\frac{k}{s}$ 个批次称为组 1，将接下来的 $\frac{k}{s}$ 个批次称为组 2，依此类推。因此，在线顶点拥有的组总数 u 至多为 $\frac{k}{k/s} = s$。注意，在线顶点 u 在每个组 i 中的度至多为

$$(\#\text{ batches in group } i) \cdot (\text{degree in each batch}) = \frac{k}{s} \cdot k = O\left(\frac{\Delta}{s}\right). \tag{3}$$

与我们在上一小节遇到的挑战类似，每个组中离线顶点的度数都没有小于 Δ 的上界。为了解决这一问题，我们在数据流开始时，为每个顶点 u 均匀且独立地抽取一个随机数 $g_u \in [s]$。现在，对于第 i 组，将其置换后的组编号定义为

$$\sigma_u(i) = (i + g_u) \bmod s, \tag{4}$$

并让 $H_{i,u}$ 在其 j 组中包含 u 的边，其中 $\sigma_u(j) = i$。现在，对于任意 $i \in [s]$，我们定义

$$H_i := \bigcup_{u \in U} H_{i,u}$$

包含所有具有置换后的组编号 i的边的图。

以下断言与引理 4.3 类似，并且在满足以下条件时界定了每个 H_i 的最大度： Δ 足够大。

断言 5.3. 以 $1 - 1/\text{poly}(n)$ 的概率，对所有 $i \in [s]$，H_i 中的最大度至多为 $O(\Delta/s)$。

证明。证明与第 4 节的引理 4.3 类似。取一个在线顶点 $u \in H_i$。令 j 为满足 $\sigma_u(j) = i$ 的组。注意，满足这一条件的组只有一个 j （因为 σ_u 是从 $[s]$ 到 $[s]$ 的置换）。由于根据 (3)，顶点 u 在组 j 中有 $O(\Delta/s)$ 条边，因此它在图 H_i 中也有 $O(\Delta/s)$ 条边。注意，对在线顶点而言，这一保证是确定性的。对于离线顶点，我们使用切尔诺夫界。具体来说，取离线顶点 v 的一条边 $e = (u, v)$。当且仅当 e 所属的组 j 获得组号 $\sigma_u(j) = i$ 时，这条边才会被加入图 H_i。由于 σ_u 是 $[s]$ 的随机排列，这种情况发生的概率为 $1/s$。因此，期望而言， u 中至多有 Δ/s 条边属于 H_i。此外，注意，对于 v 的不同边，这一选择彼此独立，因为在线顶点的随机排列相互独立。因此，我们可以应用切尔诺夫界，得到

$$\Pr[v \text{ has degree } \ge 2\Delta/s \text{ in } H_i] \le \exp(-\frac{\Delta}{3s}).$$

注意到 $s \le \sqrt{\Delta}$ 和 $\Delta \ge 900\log^2 n$，我们得到 $\Delta/3s \ge \sqrt{\Delta}/3 \ge 10\log n$。因此，上述概率至多为 n^{-10}。对所有 n 种离线顶点选择以及所有 $s \le n$ 个组应用并集界，我们得到，以概率 $1 - 1/n^8$，每个 H_i 中的最大度确实为 $O(\Delta/s)$。

□

我们运行 s 引理的算法的并行实例3.9，每组运行一个实例，每个实例~~使用各自的颜色。这些实例各自的空间需求为 $O(n)$，因此总共使用 $O(ns)$空间，达到了所需的界。关于颜色数量，回顾一下，每组中的边分为 k/s 个批次，因此引理3.9 需要 $O(\Delta' \cdot k/s)$种颜色，其中 Δ' 是该组的最大度，根据断言5.3 ，其上界为 $O(\Delta/s)$。因此，算法使用的颜色总数至多为~~

$$\underbrace{s}_{\#\text{ of groups}} \times \underbrace{k/s}_{\text{batches in each group}} \times \underbrace{\Delta/s}_{\text{max degree in each group}} = O(\Delta k/s) = O(\Delta^{1.5}/s),$$

如所需。至此，证明完成；算法的伪代码见算法3 。 □

A 附录：向二分图的归约

本小节中，我们利用一种随机二分化思路给出引理3.1 的证明。这种思路已在边着色文献中得到广泛应用。据我们所知，Karloff 和 Shmoys 首先将这一思路用于 [38] 并行算法领域，而 Behnezhad 等人则将其用于 [16] 流式边着色领域。我们强调，本小节中的算法并无创新之处；我们仅为使内容自成一体而在此加以介绍。

~~引理的证明 3.1。 如果 $\Delta \le 10\log n$，那么我们将整个图存储在内存中，使用 $O(n)$ 空间，并使用离线算法通过 $(\Delta + 1)$ 种颜色对图进行边着色。因此，假设 $\Delta > 10\log n$。

设 V 为图的顶点集。我们将 V 划分为两个子集 A 和 B：将每个顶点随机放入 A 或 B，每个顶点的选择均匀且独立于其他顶点。现在，考虑 G 的二部子图 G'，其中包含 G 中任意一条边 (u, v)，其一个端点位于 A，另一个端点位于 B。注意，在给定 $v \in A$ 的条件下，原始图中 v 的每个邻居都以 1/2 的概率独立地属于 B。这意味着 G' 中 v 的期望度为 $\deg_G(v)/2$。通过简单应用切尔诺夫界，并回想 $\Delta > 10\log n$，我们得到 G' 的最大度为 $\Delta/2 + O(\sqrt{\Delta}\log n) = (1/2 + o(1))\Delta$，概率为 $1 - 1/n^{10}$。因此，我们可以在 G' 上运行二分图边着色算法。但请注意，这样会使从 A 到 A 或从 B 到 B 的边未着色。我们递归地应用相同的算法。也就是说，我们将 A 和 B 分别递归划分为两个子集。由此得到一个最大度为 $\Delta/4 + O(\sqrt{\Delta}\log n)$ 的子图，且该结果以高概率成立。我们将~~此过程~~继续进行 $O(\log \Delta)$ 步，直到所得子图的最大度小于 $O(\log n)$，此时我们将整个子图存储在内存中，并使用离线算法对其着色。算法使用的颜色总数为

$$\sum_{i=1}^{\log \Delta} f\left(\left(\frac{1}{2^i} + o(1)\right)\Delta\right) \le \sum_{i=1}^{\log \Delta} f\left(\frac{\Delta}{2^{i-1}}\right).$$

注意，这种顶点划分可以在数据流开始时、任何边到达之前完成。对于每个顶点，只需存储它属于两个随机子集中的哪一个即可

算法 3： 适用于单侧批次到达且批次大小为 $k = \Theta(\sqrt{\Delta})$ 的二分图流式边着色。该算法使用 $O(\Delta^{1.5}/s)$ 种颜色和 $O(ns)$ 空间。这是用于引理 5.2 的算法，将其代入断言 5.1 即可证明定理1.1。

参数： $k := \sqrt{\Delta}$, s (我们使用的空间为 $= O(ns)$)

1. 如果 $\Delta \leq 900\log^2 n$，则存储整个图，并使用离线算法用 Δ 种颜色为其着色。因此，在算法余下的部分中，假设 $\Delta > 900\log^2 n$。 2. 我们在输入图 G 的子图 H_1、.....、H_s 上运行 s 个引理 3.9 的算法并行实例。每个子图 H_i 都是一个二分图，其中一部分顶点包含其在线顶点，另一部分包含离线顶点。每个 H_i 的最大度为 $O(\Delta/s)$（由断言 5.3 得出），其边将按大小为 k 的批次到达，其中同一批次中的所有边 $\tilde{}$ 都与同一个在线顶点相邻。根据引理 3.9，每个实例需要 $O(n)\tilde{}$ 空间，因此所需的总空间为 $O(ns)$。 3. 对于每个在线顶点 u，均匀且独立地选取并存储一个随机数 $g_s \in [s]$ 。 4. 对于每个在线顶点 u，存储一个计数器 I_u。该计数器用于跟踪截至任意时刻已看到的 u 批次数量。初始时，我们有 $I_u \leftarrow 0$。 5. 当一个在线顶点 u 的大小为 k 的批次到达时：(a) $I_u \leftarrow I_u + 1$。(b) 令 $I'_u := \lceil I_u/(k/s) \rceil$ 为 u 的组号。具体而言， u 的前 k/s 个批次属于第 1 组，接下来的 k/s 个批次属于第 2 组，依此类推。(c) 将 u 的 k 条边作为一个批次，从缓冲区 T 中移除，并将它们作为一个批次输入图 H_i，其中 (d) 使用在图 H_i 上运行的引理 3.9 的实例，为 u 的边着色。

$$i := (I'_u + g_u) \bmod s.$$

在每个 $O(\log\ \Delta)$ 层。因此，这只需要 $O(n\log\ \Delta)$ 的额外空间。此外，如果原始图的边按顶点到达的方式到达，那么随机二分图的边也如此。

□

参考资料

[1] Kook Jin Ahn、Sudipto Guha 和 Andrew McGregor。通过线性测量分析图结构。载于第二十三届ACM-SIAM离散算法研讨会论文集，SODA 2012，日本京都，2012年1月17日至19日，第459–467页，2012年。[2] Kook Jin Ahn、Sudipto Guha 和 Andrew McGregor。图摘要：稀疏化、生成树和子图。载于第31届ACM SIGMOD-SIGACT-SIGART数据库系统原理研讨会论文集，PODS 2012，美国亚利桑那州斯科茨代尔，2012年5月20日至24日，第5–14页，2012年。[3] Kook Jin Ahn、Graham Cormode、Sudipto Guha、Andrew McGregor 和 Anthony Wirth。数据流中的相关聚类。载于第32届国际机器学习会议论文集，ICML 2015，法国里尔，2015年7月6日至11日，第2237–2246页，2015年。[4] Mohammad Ansari、Mohammad Saneian 和 Hamid Zarrabi-Zadeh。用于边着色的简单流式算法。载于第30届欧洲算法研讨会，ESA 2022，2022年9月5日至9日，德国柏林/波茨坦，第8:1–8:4页，2022年。[5] Sepehr Assadi。半流式最大匹配的两遍（条件）下界。载于2022年ACM-SIAM离散算法研讨会论文集，SODA 2022，虚拟会议/美国弗吉尼亚州亚历山德里亚，2022年1月9日至12日，第708–742页，2022年。[6] Sepehr Assadi 和 Soheil Behnezhad。随机顺序流式匹配中突破三分之二。载于第48届自动机、语言与编程国际研讨会，ICALP 2021，2021年7月12日至16日，苏格兰格拉斯哥（虚拟会议），第19:1–19:13页，2021年。[7] Sepehr Assadi 和 Aditi Dudeja。用于全局最小割的简单半流式算法。载于第4届算法简洁性研讨会，SOSA 2021，虚拟会议，2021年1月11日至12日，第172–180页，2021年。[8] Sepehr Assadi 和 Aditi Dudeja。用于全局最小割的简单半流式算法。载于第4届算法简洁性研讨会，SOSA 2021，虚拟会议，2021年1月11日至12日，第172–180页。SIAM，2021年。[9] Sepehr Assadi 和 Janani Sundaresan。（含噪）间隙环计数卷土重来：连通分量及其他问题的随机顺序流式下界。载于第55届ACM SIGACT计算理论研讨会论文集，STOC 2023，即将出版，2023年。[10] Sepehr Assadi、Yu Chen 和 Sanjeev Khanna。用于（$\delta+1$）顶点着色的次线性算法。载于第三十届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767–786页，2019年。[11] Sepehr Assadi、Amit Chakrabarti、Prantar Ghosh 和 Manuel Stoeckl。通过确定性和对抗鲁棒算法在图流中进行着色。CoRR，abs/2212.10641，2022年。

[12] Sepehr Assadi、Andrew Chen 和 Glenn Sun。流式模型中的确定性图着色。载于STOC ’22：第54届ACM SIGACT计算理论研讨会，意大利罗马，2022年6月20日至24日，第261–274页，2022年。[13] Sepehr Assadi、Pankaj Kumar 和 Parth Mittal。图流中的Brooks定理：用于 δ-着色的单遍半流式算法。载于STOC ’22：第54届ACM SIGACT计算理论研讨会，意大利罗马，2022年6月20日至24日，第234–247页，2022年。[14] Sepehr Assadi、Soheil Behnezhad、Sanjeev Khanna 和 Huan Li。关于流式和动态匹配中的正则性引理及障碍。载于第55届ACM SIGACT计算理论研讨会论文集，STOC 2023，即将出版，2023年。[15] Amotz Bar-Noy、Rajeev Motwani 和 Joseph Naor。贪心算法对于在线边着色是最优的。《信息处理快报》，44(5)：251–253，1992年。[16] Soheil Behnezhad、Mahsa Derakhshan、MohammadTaghi Hajiaghayi、Marina Knittel 和 Hamed Saleh。用于边着色的流式和大规模并行算法。载于第27届欧洲算法研讨会，ESA 2019，2019年9月9日至11日，德国慕尼黑/加兴，第15:1–15:14页，2019年。[17] Soheil Behnezhad、Moses Charikar、Weiyun Ma 和 Li-Yang Tan。常数轮中几乎达到3-近似的相关聚类。载于第63届IEEE计算机科学基础年会，FOCS 2022，美国科罗拉多州丹佛，2022年10月31日至11月3日，第720–731页，2022年。[18] Soheil Behnezhad、Moses Charikar、Weiyun Ma 和 Li-Yang Tan。相关聚类的单遍流式算法。载于2023年ACM-SIAM离散算法研讨会论文集，SODA 2023，即将出版，2023年。[19] Suman K. Bera、Amit Chakrabarti 和 Prantar Ghosh。通过流式及其他节省空间的模型中的退化度进行图着色。载于第47届自动机、语言与编程国际研讨会，ICALP 2020，2020年7月8日至11日，德国萨尔布吕肯（虚拟会议），第11:1–11:21页，2020年。¨Sayan Bhattacharya、Fabrizio Grandoni 和 David Wajc。通过nibble方法实现的在线边着色算法。载于2021年ACM-SIAM离散算法研讨会论文集，SODA 2021，虚拟会议，2021年1月10日至13日，第2830–2842页。SIAM，2021年。[20] Amit Chakrabarti、Prantar Ghosh 和 Manuel Stoeckl。图流的对抗鲁棒着色。载于第13届理论计算机科学创新会议，ITCS 2022，2022年1月31日至2月3日，美国加利福尼亚州伯克利，第37:1–37:23页，2022年。[21] Moses Charikar 和 Paul Liu。w-流模型中的改进边着色算法。载于第4届算法简洁性研讨会，SOSA 2021，虚拟会议，2021年1月11日至12日，第181–183页，2021年。[22] Yu Chen、Sanjeev Khanna 和 Huan Li。通过线性稀疏表示实现加权图稀疏化。载于第63届IEEE计算机科学基础年会，FOCS 2022，美国科罗拉多州丹佛，2022年10月31日至11月3日，第474–485页，2022年。[23] Ilan Reuven Cohen、Binghui Peng 和 David Wajc。在线边着色的紧界。载于第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日至12日，第1–25页。IEEE计算机学会，2019年。

[25] Vincent Cohen-Addad、Silvio Lattanzi、Slobodan Mitrovic、Ashkan Norouzi-Fard、Nikos Parot-sidis 和 Jakub Tarnawski。在常数次并行轮次中进行相关聚类。载于《国际机器学习会议》第38届会议论文集，ICML 2021，2021年7月18日至24日，线上会议，第2069–2078页，2021年。[26] Camil Demetrescu、Irene Finocchi 和 Andrea Ribichini。在图流问题中以空间换取遍历次数。《算法杂志》，6(1)，2010年12月。ISSN 1549-6325。doi：10.1145/1644015.1644021。[27] Joan Feigenbaum、Sampath Kannan、Andrew McGregor、Siddharth Suri 和 Jian Zhang。半流模型中的图问题。载于《自动机、语言与程序设计：第31届国际研讨会》，ICALP 2004，芬兰图尔库，2004年7月12日至16日。论文集，第531–543页，2004年。[28] Alan M. Frieze。一类随机图中的最大匹配。《组合理论杂志B辑》，40(2)：196–212，1986年。[29] Harold N Gabow。图的边着色算法。技术报告，1985年。[30] Mohsen Ghaffari、Krzysztof Nowicki 和 Mikkel Thorup。通过随机二出收缩实现更快的边连通性算法。载于《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日至8日，第1260–1279页。SIAM，2020年。[31] Prantar Ghosh 和 Manuel Stoeckl。在线边着色和 w-流式边着色的低内存算法。CoRR，abs/2304.12285，2023年。[32] Christian Glazik、Jan Schiemann 和 Anand Srivastav。在 w-流模型中单遍寻找欧拉游走，使用 o(n log(n))随机存取存储器。CoRR，abs/1710.04091，2017年。[33] Jacob Holm、Valerie King、Mikkel Thorup、Or Zamir 和 Uri Zwick。随机 k出子图仅留下 o(n/k) 条分量间边。载于第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日至12日，第896–909页。IEEE计算机学会，2019年。[34] John E. Hopcroft 和 Richard M. Karp。二部图最大匹配的 n5/2 算法。载于第12届切换与自动机理论年会（SWAT 1971），第122–125页，1971年。[35] Michael Kapralov。边到达模型中近似最大匹配的空间下界。载于《2021年ACM-SIAM离散算法研讨会论文集》，SODA 2021，线上会议，2021年1月10日至13日，第1874–1893页，2021年。[36] Michael Kapralov、Yin Tat Lee、Cameron Musco、Christopher Musco 和 Aaron Sidford。动态流中的单遍谱稀疏化。载于第55届IEEE计算机科学基础年会，FOCS 2014，美国宾夕法尼亚州费城，2014年10月18日至21日，第561–570页，2014年。[37] Michael Kapralov、Aida Mousavifar、Cameron Musco、Christopher Musco、Navid Nouri、Aaron Sidford 和 Jakab Tardos。动态流中的快速且节省空间的谱稀疏化。载于《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日至8日，第1814–1833页，2020年。

[38] Howard J. Karloff 和 David B. Shmoys。边着色问题的高效并行算法。《算法杂志》，8(1)：39–52，1987年。[39] Janardhan Kulkarni、Yang P. Liu、Ashwin Sah、Mehtaab Sawhney 和 Jakub Tarnawski。通过树递推和相关性衰减实现在线边着色。载于 STOC ’22：第54届ACM SIGACT计算理论研讨会，意大利罗马，2022年6月20日至24日，第104–116页。ACM，2022年。[40] Amin Saberi 和 David Wajc。贪心算法并非在线边着色的最优算法。载于第48届自动机、语言与编程国际研讨会，ICALP 2021，2021年7月12日至16日，苏格兰格拉斯哥（线上会议），LIPIcs第198卷，第109:1–109:18页。Schloss Dagstuhl - Leibniz-Zentrum f¨ur Informatik，2021年。