全点对有界边连通性的高效算法

Shyan Akmal ！美国马萨诸塞州剑桥市 MIT EECS 与 CSAIL

Ce Jin ！美国马萨诸塞州剑桥市 MIT EECS 与 CSAIL

arXiv:2305.02132v1 [cs.DS] 3 May 2023

摘要

我们的工作研究无权图中最大流的一种变体的算法。在全点对连通性（APC）问题中，给定一个包含 G 个 n 顶点和 m 条 m 边的图 G，任务是针对 G中的所有顶点对（s，t），计算从 s 到 t 的边不相交路径的最大数量（等价地，最小（s，t）割的大小）。近期重要的算法突破表明，在无向图上，APC 可以在 $n^{2+o(1)}$ 时间内求解，这在本质上已达到最优。相比之下，有向图上 APC 的真正时间复杂度仍是开放问题：该问题可以在 $\tilde{O}(m^\omega)$时间内求解，其中 $\omega \in [2, 2.373)$是矩阵乘法的指数，但目前尚不清楚是否存在相匹配的条件下界。

受 [Abboud 等人、ICALP 2019], 工作的启发，我们研究 APC 的一个有界版本，称为 k-有界全点对连通性（k-APC）问题。在这一 APC 变体中，除图 G外，还给定一个整数 k，任务是仅对最小割值小于 k 的顶点对（s,t），报告最小（s,t）-割的大小（如果最小（s,t）-割的大小至少为 k，我们只需报告它是“大”的，而不必计算其精确值）。

我们的主要结果是一个运行时间为 $\tilde{O}((kn)^\omega)$ 的算法，可求解有向图上的 k-APC。这是首个能够比直接精确求解更一般的 APC 问题更快地求解 k-APC 的算法，对所有 $k \geq 3$ 均成立。对于所有 $k \leq \text{poly}(\log n)$，其运行时间为 $\tilde{O}(n^\omega)$；在细粒度复杂性领域广为接受的猜想下，这基本达到了 $k = 1$ 情形下 k-APC 的最优运行时间。此前，只有在一般有向图中求解 $k \leq 2$ 时 [Georgiadis 等人、ICALP 2017], ，以及在有向无环图这一特殊情形中求解 $k \leq o(\sqrt{\log n})$ 时 [Abboud 等人、ICALP 2019]，才能达到这一运行时间。我们的结果采用了以往工作中使用的同一代数框架，该框架由 [Cheung、Lau 和 Leung，FOCS 2011]提出。直接实现这一框架需要求逆一个大型随机矩阵。我们新算法的关键洞见是，求解 k-APC 时，只需对一个低秩随机矩阵求逆，而不必对一般随机矩阵求逆。

我们还为 k-APC 的一个变体提出了一种新算法，即 k有界全点对顶点连通度（k-APVC）问题。在该问题中，对于每一对顶点（s,t），如果从 s 到 t 的内部顶点不相交路径的最大数量小于 k，则需要报告该最大数量；否则报告从 s 到 t 至少存在 k 条内部顶点不相交路径。

我们的第二项成果是一种运行时间为 $\tilde{O}(k^2 n^\omega)$ 的算法，用于解决有向图中的 k-APVC。此前的工作展示了如何在 $O((kn)^\omega)$ 时间内解决较简单版本的 k-APVC 问题（对于图中不相邻的顶点对（s,t），只需返回答案） [Abboud 等人，ICALP 2019]。相比之下，我们的算法解决了完整的 k-APVC 问题，并且在 $\omega > 2$ 时速度更快。

2012 年 ACM 学科分类 计算数学 → 图算法

关键词与短语 最大流、全点对、连通度、矩阵秩

资助 *Shyan Akmal*：部分获得美国国家科学基金会资助，项目编号为 CCF-2129139 和 CCF-2127597。*Ce Jin*：部分获得美国国家科学基金会项目 CCF-2129139 和 Siebel 奖学金的资助。

致谢 第一作者感谢 Virginia Vassilevska Williams 就计算矩阵秩的算法进行了富有见地的讨论。

1 引言

计算最大流是一个经典问题，在图论和计算机科学领域已得到广泛研究。在无权图中，这一任务具体化为计算连通度，这本身也是一个有趣的计算问题。给定一个含有 n 个顶点和 m 条边的图 G，对于 G 中任意顶点 s 和 t，从 s 到 t 的连通度 $\lambda(s,t)$ 定义为从 s 到 t 的边不相交路径[1] 的最大数量。由于最大流可以在近线性时间内计算，因此对于 $m^{1+o(1)}$ 中给定的任意顶点 s 和 t，我们可以在时间 [4]内计算 $\lambda(s,t)$。

如果我们的目标不是仅仅返回单个连通度的值，而是计算图中的所有连通度，会怎样呢？这就引出了全点对连通性（APC）问题：在该问题中，我们给定一个如上所述的图 G，任务是计算 $\lambda(s,\ t)$ 中所有顶点对 $(s,\ t)$ 的连通度。在无向图中，可以在 $n^{2+o(1)}$ 时间 [2]，求解 APC，因此，这个“全点对”问题实质上并不比在稠密图中输出单个连通度更难。

在有向图中，APC 似乎要困难得多，各种条件下界（在第 1.2 节中讨论）表明，该问题不太可能在二次时间内求解。对每一对顶点分别直接计算连通度，会得到一个 $n^2m^{1+o(1)}$ 时间的算法来解决该问题。利用流向量框架（在第 3 节中讨论），可以在有向图中以 $\tilde{O}(m^\omega)$ 时间[2] [6]，求解 APC，其中 ω 是 ˜ 矩阵乘法的指数。已知算法表明 $\omega < 2.37286$ [3]，因此，只要输入图不是过于稠密，该 $O(m^\omega)$ 时间算法就比朴素算法更快。

我们的工作聚焦于 APC 问题的一个有界版本，我们将其正式表述为 k-有界全点对连通性（k-APC）问题：在该问题中，给定一个如上所述的有向图 G，我们的任务是计算图中所有顶点对 $(s,\ t)$ 的 $\min(k,\ \lambda(s,\ t))$。直观而言，这是对 APC 问题的一种松弛形式，我们的目标仅是计算连通度较小的顶点对 $(s,\ t)$ 的 $\lambda(s,\ t)$ 精确值。对于所有其他顶点对，只需报告其连通度较大即可，其中 k 是用于区分连通度值大小的阈值。

当 $k = 1$ 时，k-APC 问题等价于计算输入图的传递闭包（在此问题中，对于每一对顶点 (s,t)，我们需要确定 G 是否包含一条从 s 到 t的路径），这可以在 $O(n^\omega)$ 时间内完成 [7]。类似地，对于 $k = 2$ 的特殊˜情形，已知可以通过一种采用精心设计的矩阵乘积的分治算法，在 $O(n^\omega)$ 时间内求解 k-APC [9]。正如我们在第 1.2 节中所讨论的，有证据表明，当 $k \leq 2$ 时，k-APC 的这些运行时间基本上是最优的。

即使对于 $k = 3$，目前也尚不清楚 k-全点对连通性是否能比计算所有顶点对 (s,t) 的 $\lambda(s,t)$ 的精确值更快地求解！粗略地说，这是因为，已知的用于全点对连通性的 $O(m^\omega)$ 时间算法涉及将连通性信息编码在一个 $m \times m$ 矩阵的逆矩阵中，而求一个 $m \times m$ 矩阵的逆通常需要 $O(m^\omega)$ 时间。这一步编码对于 k-全点对连通性 似乎同样是必要的。对于 $k = 2$，关于强连通图结构的巧妙组合观察使我们能够跳过这一计算；但对于 $k \geq 3$，从先前的工作中完全无法看出如何避免这一瓶颈。此外，现有的困难性结果与这样一种可能性相容：对于任意常数 k，k-全点对连通性 都可以在 $O(n^\omega)$ 时间内求解。

---

[1] 根据门格尔定理，$\lambda(s,\ t)$也等于从图 G 中删除的最少边数，使其成为一个不含 s 到 t 路径的图。[2] 给定一个函数 f，我们用 $\tilde{O}(f)$表示 $f \cdot \text{poly}(\log f)$。

▶ 开放问题1. </style}能否在快于 $O(m^\omega)$ 的时间内解决 k-全点对连通性问题（k= $k=3$）？

由于对 k-APC 的复杂性缺乏了解，研究人员也研究了这一问题的较简单版本。给定图 G 中的顶点 s 和 t，我们将从 s 到 t 的顶点连通度 $\nu(s,t)$ 定义为从 s 到 t 的内部顶点不相交路径的最大条数。我们可以考虑 APC 和 k-APC 问题对应的顶点连通度问题。在全点对顶点连通度（APVC）问题中，给定一个有 n 个顶点和 m 条边的图 G，任务是计算 G 中所有顶点对 (s,t) 的 $\nu(s,t)$ 值。在 k-有界全点对顶点连通度（k-APVC）问题中，给定与上述相同的输入 G，但任务仅是计算 G 中所有顶点对 (s,t) 的 $\min(k,\nu(s,t))$。

k-APVC 问题并不存在 $O(m^\omega)$ 的瓶颈，而现有的用于 k-全点对连通性问题的算法技术似乎会遇到这一瓶颈；直观而言，这是因为可以将图的全部点连通度信息编码在一个 $n\times n$ 矩阵的逆矩阵中，而非一个 $m\times m$ 矩阵。因此，[1] 能够提出一种运行时间为 $O((kn)^\omega)$ 的算法，为所有不相邻的顶点对 (s,t) 计算 $\min(k,\nu(s,t))$。鉴于这一结果，自然会提出这样的问题：更一般的 k-APVC 和 k-全点对连通性问题是否也能在相同的运行时间内解决?

▶ **开放问题2.** 能否在 $O((kn)^\omega)$ 的时间内解决 k-APVC 问题?

▶ **开放问题3**。 k-APC 能否在 $O((kn)^\omega)$ 时间内求解?

1.1 我们的贡献

我们解决了上一节提出的全部三个开放问题。

首先，我们提出了一种更快的 k-APC 算法，其时间复杂度与先前工作中求解更简单版本的 k-APVC 所给出的运行时间相当。

▶**定理4**。 对于任意正整数 k， *k-APC* 可以在 $O((kn)^\omega)$ 时间内求解。

这是第一个能够比直接使用以下方法精确求解 APC 更快地求解 k-APC 的算法
the $\tilde{O}(m^\omega)$ 时间算法，适用于所有常数 $k\geq 3$。

其次，我们给出一种求解 k-APVC 的算法。若 $\omega>2$，该算法比 [1] 给出的 $O((kn)^\omega)$ 时间算法更快；后者只能求解 k-APVC 的一个受限版本。

▶**定理 5.** 对于任意正整数 k， *k-APVC* 可在 $\tilde{O}(k^2n^\omega)$ 时间内求解。

1.2 与先前结果的比较

条件下界

细粒度复杂性领域中有许多广为人知的猜想（这些猜想假设某些计算任务的复杂度存在下界），它们被用作计算机科学问题条件性困难性结果的依据。本节回顾全点对连通性及其变体的已知困难性结果。本节所用问题和猜想的定义见附录 A。

假设布尔矩阵乘法（BMM）需要 $n^{\omega-o(1)}$ 时间，已知求解 k-全点对连通性和 k-APVC 需要 $n^{\omega-o(1)}$ 时间，即使对于 $k=1$ [7] 也是如此。特别是，这一假设意味着我们的 k-全点对连通性和 k-APVC 算法在 k 为常数时是最优的。

假设强指数时间假设（SETH）成立，先前的工作表明，全点对连通性需要 $(mn)^{1-o(1)}$ 时间 [11, 定理 1.8], APVC 需要 $m^{3/2-o(1)}$ 时间 [13, 定理 1.7],而 k-全点对连通性需要 $\left(kn^2\right)^{1-o(1)}$ 时间 [11, 定理4.3]。

令 $\omega(1,2,1)$ 为最小实数[3]，使得我们能够在 $n^{\omega(1,2,1)+o(1)}$ 时间内计算一个$n \times n^2$ 矩阵与一个 $n^2 \times n$ 矩阵的乘积。假设 4-团猜想成立，则有向图上的 k-APVC 问题（因此 k-APC 问题也是如此）需要$\left(k^2 n^{\omega(1,2,1)-2}\right)^{1-o(1)}$ 时间 [1]。4-团猜想还意味着，求解无向图上的 APVC 需要 $n^{\omega(1,2,1)-o(1)}$ 时间 [10]。

受限图类的算法

如前所述，在我们的工作之前，对于一般有向图上的 k-全点对连通性，已知的算法都不具实质意义。然而，对于有向无环图（DAG）上的 k-全点对连通性，已有更快的算法。具体而言， [1] 提出了两种求解 DAG 上 k-全点对连通性的算法，其运行时间分别为 $2^{O(k^2)}mn$ 和 $(k \log n)^{4^k+o_{(k)}}n^\omega$。

相比之下，我们在定理4中的算法可以求解一般有向图上的 k-全点对连通性；当 $m \geq n^{\omega-1}$ 或 $k \geq \omega(\sqrt{\log n})$（例如）时，该算法比前一种算法更快，且始终比后一种算法更快；从技术角度来看，该算法也比先前的论证简单得多。不过，这些用于求解 DAG 上 k- 全点对连通性的算法还能返回用于证明连通性值的割，而我们的算法不能。

对于无向图这一特殊情形，APVC 可以在 $m^{2+o(1)}$ time [13,˜定理 1.8], 中所述的时间内求解；当 $\omega > 2$ 时，这比前面提到的 $O(m^\omega)$ 时间算法更快。对于无向图， k-APVC 可以在 $k^3 m^{1+o(1)} + n^2$ poly(log n) 时间内求解。相比之下，我们在定理 5 中提出的算法能够处理无向和 有向图上的 k-APVC，并且当 k 的值足够大时，在稠密图上具有更快的运行速度。

在最大度为 d, [6, 的有向平面图中，5] 定理 1. 证明 APC 可在 $O\left(d^{\omega-2} n^{\omega/2+1}\right)$ 时间内求解。

其他相关工作

在 [14], 中，作者研究了 k-全点对连通性的一种对称变体。此处，输入是一个包含 n 个顶点和 m 条边的有向图 G，目标是计算所有顶点对（s, t）的 min($k, \lambda(s, t)$ $\lambda(t, s)$) 值。这个较简单的问题可以在 $O(kmn)$ 时间内解决 [14]。

1.3 组织结构

本文其余部分致力于证明定理 4 和定理 5。在第 2 节中，我们介绍符号、一些有用的定义，以及关于矩阵计算的结果，这些内容将有助于证明我们的算法的正确性。在第 3 节中，我们直观地概述用于 k-全点对连通性和 k-APVC 的算法。在第 4 节中，我们描述一个用于捕获连通度值的“流向量”框架，并在第 5 节中利用该框架证明定理4。在第 6 节中，我们给出关于点连通度的有用结果，并在第 7 节中利用这些结果证明定理 5。我们在第 8 节中总结全文，并重点介绍一些由我们的工作引出的有趣的开放问题。

在附录 A 中，我们给出第 1.2 节中提到的问题和猜想的定义。在附录 B 中，我们讨论 [1] 中对 k-APVC 的处理方式与我们自身处理方式的不同，并给出第 6 节中所述某项结果的证明细节。

[3] 已知的快速矩阵乘法算法表明， $\omega(1, 2, 1) < 3.25669$ [8, 表 2]。

2 预备知识

图的假设

在全文中，我们用 G 表示一个具有 n 个顶点和 m 条边的有向图。不失一般性地，我们假设 G 的底层无向图是连通的，即 G 是弱连通的（否则，我们只需分别对 G 的每个弱连通分量运行我们的算法），因此有 $m \geq n-1$。我们假设 G 没有自环，因为自环不会影响不同顶点之间的连通度或点连通度值。

在第 4 节和第 5 节中，我们研究 k-APC 问题，因此允许 G 的顶点之间存在平行边（即 G 可以是多重图）。不过，不失一般性，我们假设对于任意不同的顶点 s 和 t，从 s 到 t 最多有 k 条边（因为如果从 s 到 t 有超过 k 条平行边，我们可以删除其中一些，使平行边的数量降至 k，而不改变 $\min(k, \lambda(s, t))$ 的值）。在第 6 节和第 7 节中，我们研究 k-APVC 问题，因此假设 G 是没有平行边的简单图，因为从 u 到 v 的平行边不会影响顶点连通度 $\nu(s, t)$ 的值；但如果 $u = s$ 和 $v = t$，则 $\nu(s, t)$ 的值只会增加从 s 到 t 的额外平行边数量。

图术语与记号

给定图 G 中一条从 u 到 v 的边 e，我们记作 $e = (u, v)$。我们称 u 为 e 的尾，称 v 为 e 的头。进入顶点 v 的边的尾所对应的顶点称为 v 的入邻点。类似地，从 v 出发的边的头所对应的顶点称为 v 的出邻点。给定图 G 中的顶点 u，我们用 $E_{\text{in}}(u)$ 表示进入 u 的边集，用 $E_{\text{out}}(u)$ 表示从 u 出发的边集。类似地，$V_{\text{in}}(u)$ 表示 u 的入邻点集，$V_{\text{out}}(u)$ 表示 u 的出邻点集。此外，我们定义 $V_{\text{in}}[u] = V_{\text{in}}(u) \cup \{u\}$，以及 $V_{\text{out}}[u] = V_{\text{out}}(u) \cup \{u\}$。最后，令 $\deg_{\text{in}}(u) = |E_{\text{in}}(u)|$ 和 $\deg_{\text{out}}(u) = |E_{\text{out}}(u)|$ 分别表示 u 的入度和出度。

给定顶点 s 和 t，(s,t)-割是一个边集 C，删除 C 中的边后，会得到一个不存在从 s 到 t 的路径的图。根据门格尔定理，最小 (s,t)-割的大小等于从 s 到 t的连通度 $\lambda(s,t)$。类似地，(s,t)-顶点割是一个顶点集 C'，其中 $s, t \notin C'$，删除 C' 后，会得到一个不存在从 s 到 t 的路径的图。显然，当 (s,t) 不是一条边时，顶点割存在。当 (s,t) 不是一条边时，门格尔定理表明，最小 (s,t)-顶点割的大小等于从 s 到 t的顶点连通度 $\nu(s,t)$。

矩阵表示法

令 A 为一个矩阵。对于索引 i 和 j，我们令 $A[i,j]$ 表示 A 的 (i,j) 个元素。更一般地，如果 S 是一组行索引，T 是一组列索引，我们令 $A[S,T]$ 表示将 A 限制为 S 中的行和 T 中的列所得到的子矩阵。类似地，$A[S,*]$ 表示将 A 限制为 S 中的行所得的矩阵，而 $A[*,T]$ 表示将 A 限制为 T 中的列所得的矩阵。我们令 $A^\top$ 表示 A 的转置。如果 A 是一个方阵，则我们令 $\text{adj}(A)$ 表示 A 的伴随矩阵。如果 A 可逆，则我们令 A^{-1} 表示其逆矩阵。如果一个定理、引理或命题的陈述涉及 A^{-1}，通常表示该陈述的一部分是断言 A^{-1} 存在（或者，如果 A 是一个随机矩阵，则断言 A^{-1} 以某个概率存在）。我们令 I 表示单位矩阵（该矩阵的维度将始终可由上下文明确）。给定一个向量 $\vec{v}$，对于任意索引 i，我们令 $\vec{v}[i]$ 表示 $\vec{v}$ 中的第 i^{th} 个元素。我们令 $\vec{0}$ 表示零

向量（该向量的维度始终可以从上下文中明确得知）。给定一个正整数 k，我们用 $[k] = \{1, \ldots, k\}$ 表示前 k 个正整数构成的集合。

矩阵与多项式计算

给定一个素数 p，令 $\mathbb{F}_p$ 表示含有 p 个元素的有限域。在 $\mathbb{F}_p$ 的˜个元素上进行算术运算，可以在 $O(\log p)$时间内完成。

现在回顾一些关于矩阵和多项式计算的众所周知的结果，这些结果将对我们的算法有所帮助。

▶ **命题 6**。 令 A 为一个 $a \times b$ 矩阵， B 为一个 $b \times a$ 矩阵。若 $(I - BA)$ 可逆，则矩阵 $(I - AB)$ 也可逆，其逆为

$$(I - AB)^{-1} = I + A(I - BA)^{-1}B.$$

证明。 只需验证将 $(I - AB)$ 与上述方程右侧相乘所得的结果为单位矩阵。事实上，我们有

$$\begin{aligned}
&(I - AB)\left(I + A(I - BA)^{-1}B\right) \\
&= I + A(I - BA)^{-1}B - AB - ABA(I - BA)^{-1}B \\
&= I + A(I - BA)^{-1}B - AB - A\big(I - (I - BA)\big)(I - BA)^{-1}B \\
&= I + A(I - BA)^{-1}B - AB - A(I - BA)^{-1}B + AB,
\end{aligned}$$

这便化简为 I，正如所需。 ◀

▶ **命题 7.** 设 A 是域 $\mathbb{F}_p$ 上的一个 $a \times a$ 矩阵。我们可以用 $O(a^\omega)$ 次域运算计算其逆矩阵 A^{-1} （若存在）。

▶ **命题 8** ([5, 定理 11])。 设 A 是域 $\mathbb{F}_p$ 上的一个 $a \times b$ 矩阵。那么，对于任意正整数 k，我们可以用 $O(ab + k^\omega)$ 次域运算计算 $\min(k, \text{秩 } A)$。

▶ **命题 9** (施瓦茨-齐佩尔引理 [12, 定理 72])。 设 $f \in \mathbb{F}_p[x_1, \ldots, x_r]$ 是一个次数为 d 的非零多项式。设 $\vec{a}$ 是 $\mathbb{F}_p^r$ 中的一个均匀随机点。那么， $f(\vec{a})$ 非零的概率至少为 $1 - d/p$。

3 证明概述

3.1 流向量编码

以往用于全点对连通性 [6]及其变体的算法分为两个步骤:

步骤 1：编码
在此步骤中，我们构造一个矩阵 M，它隐式编码了输入图的连通性信息。

步骤 2：解码
在此步骤中，我们遍历图中所有顶点对（s， t），并针对每一对在 M 的一个子矩阵上执行少量计算，以求得所需的连通性值。

编码步骤中的构造基于**流向量**框架，该框架由 [6] 提出，是对网络编码经典技术的一种推广。我们概述这种方法此前如何应用于全点对连通性问题。[4]

[4] 对于 APVC 问题，我们采用一种不同但类似的框架，详见第3.3节。

给定输入图 G，我们固定一个源点 s。令 $d = \deg_{\text{out}}(s)$，并令 $\mathbb{F}$ 为某个底域。[5] 我们的最终目标是为图中的每条边 e 分配一个特殊向量 $\vec{e} \in \mathbb{F}^d$，我们称之为流向量。

首先，对于每条边 $e \in E_{\text{out}}(s)$，我们引入一个 d 维向量 $\vec{v_e}$。直观而言，这些向量对应于从 s 泵出的某种初始流。重要的是，这些向量线性无关（以往的应用总是选取彼此不同的 d 维单位向量作为这些向量）。然后，我们将此流推送至图的其余部分：为每条边分配一个向量，该向量是分配给进入其尾的各条边的流向量的随机线性组合。也就是说，给定一条边 $e = (u, v)$，其中 $u \neq s$，最终的流向量 $\vec{e}$ 将是进入 u 的各条边对应流向量的随机线性组合。另一方面，如果边 $e = (s, v)$ 属于 $E_{\text{out}}(s)$，那么最终的流向量 $\vec{e}$ 将是进入 s 的各条边对应流向量的随机线性组合，再加上初始流 $\vec{v}_e$。

采用随机线性组合的目的是（以高概率）保持线性无关性。在这种设定下，对于任意顶点 v 和整数 ℓ，如果分配给 $E_{\text{in}}(v)$ 中各条边的 ℓ 个流向量中的某个子集线性无关，那么我们预期，分配给 $E_{\text{out}}(v)$ 中各条边的至多 ℓ 个流向量的每个子集也都线性无关。事实证明，这种性质可以推广为保持割两侧流向量的线性无关性，这意味着（以高概率）对于任意顶点 t，分配给 $E_{\text{in}}(t)$ 中各条边的流向量的秩等于 $\lambda(s, t)$。

直观地说，这是因为分配给 $E_{\text{in}}(t)$ 中各边的流向量将是分配给最小 (s,t)-割中各边的 $\lambda(s,t)$ 个流向量的线性组合，而分配给该割中各边的流向量应当是线性无关的。

将所有流向量作为列向量收集到一个矩阵中，可以生成单个矩阵 M_s，使得计算 $M_s[*, E_{\text{in}}(t)]$ 的秩即可得到所需的连通性值 $\lambda(s, t)$（计算这些秩即为前文提到的 **解码** 步骤）。先前的工作 [6, 1] 将初始泵入的 $\vec{v}_e$ 设为互不相同的单位向量。事实证明，对于这种起始向量的选择，可以构造一个单个矩阵 M（与固定选择的 s无关），使得对 M 的子矩阵进行秩查询，便能得到我们希望在全点对连通性问题及其变体中输出的答案。

在第 3.2 节中，我们将介绍如何运用流向量框架证明定理4。随后，在第3.3节中，我们将介绍如何修改这些方法来证明定理 5。

3.2 全点对连通性

~我们的出发点是 [6], 中介绍的用于全点对连通性（m^ω）的 O时间算法，该算法使用了第 3.1 节概述的流向量编码方案。

设 K 为一个矩阵，其行和列均由输入图中的边进行索引。对于每一对边（e, f），如果 e的头与 f的尾重合，则令 $K[e, f]$为 $\mathbb{F}$中的均匀随机域元素。否则， $K[e, f] = 0$。这些域元素恰好对应于流向量框架中随机线性组合所使用的系数。定义矩阵

$$M = (I - K)^{-1}. \tag{1}$$

然后， [6] 证明以高概率，对于任意一对顶点（s, t），都有

$$\operatorname{rank} M[E_{\text{out}}(s), E_{\text{in}}(t)] = \lambda(s, t). \tag{2}$$

[5] 在我们的应用中，我们将选取 $\mathbb{F}$ 为大小为 poly(m) 的有限域。

在此设置下，全点对连通度的算法很简单：首先计算 M （**编码**步骤），然后对每一对顶点（s, t），返回秩 $M[E_{\text{out}}(s)\ E_{\text{in}}(t)]$，作为从 s 到 t 的连通度（**解码**步骤）。

˜根据方程 (1)，只需对一个元素来自 $\mathbb{F}$ 的 $m \times m$ 矩阵求逆，我们就能在 $O(m^\omega)$ 时间内完成 **编码**步骤。事实证明，我们也能在相同的时间复杂度内完成**解码**步骤。因此，这给出了一个用于全点对连通性的 $O(m^\omega)$ 时间算法。

假设现在我们想要解决 k-全点对连通性问题。在这种情况下，一个简单的技巧（例如，在 [1, 定理 5.2] 的证明中所采用的技巧）可以加快 **解码** 步骤的运行速度。然而，如何加快 **编码** 步骤的速度却远非显而易见。若要按照第 3.1 节所述实现流向量方案，似乎几乎不可避免地需要对一个 $m \times m$ 矩阵求逆。事实上，无法克服这一瓶颈这一点，在 [1] 中被明确列为转而关注 k-APVC 问题的动机之一。

我们的改进

我们用于 k-全点对连通性的新算法的核心思想，是在 **编码** 步骤中使用方程 (1) 里的矩阵 K 的低秩版本。

具体来说，我们构造若干随机稀疏矩阵 L 和 R，其维度分别为 $m \times kn$和 $kn \times m$。接着令 $K = LR$，并论证对于这一 K 的选择，方程 (1) 中定义的矩阵 M 以高概率满足

$$\operatorname{rank} M[E_{\text{out}}(s), E_{\text{in}}(t)] = \min(k, \lambda(s,t)). \tag{3}$$

这个方程只是方程 (2) 的一个 k-有界版本。根据命题 6，我们有

$$M = (I-K)^{-1} = (I-LR)^{-1} = I + L(I-RL)^{-1}R.$$

注意，$(I - RL)$ 是一个 $kn \times kn$ 矩阵。因此，要计算 M（从而完成 **编码**步骤），我们不再需要求逆一个 $m \times m$ 矩阵！相反，我们只需对一个大小为 $kn \times kn$的矩阵˜求逆。这正是定理4中 $O((kn)^\omega)$ 运行时间的主要来源。

从概念上讲，这一论证相当于通过将随机线性组合替换为随机“低秩组合”，来为图中的流向量赋值。也就是说，对于离开顶点 u的边 $e \in E_{\text{out}}(u)$，我们将 e 处的流向量定义为

$$\vec{e} = \sum_{i=1}^{k} \left(\sum_{f \in E_{\text{in}}(u)} L_i[f,u]\vec{f} \right) \cdot R_i[u,e],$$

→其中，内层求和遍历所有进入 u 的边 f， f 表示分配给边 f 的流向量，而 $L_i[f,u]$ 和 $R_i[u,e]$ 项对应于由索引 i 和某个（边、顶点）对唯一确定的随机域元素。

→这里，与第 3.1 节中描述的方法不同， f 在其对 $\vec{e}$ 的贡献中所对应的系数，并不能由边对 f 和 e 唯一确定。相反，如果边f 进入节点 u，那么对于每一个从 u流出的流向量，它所贡献的“权重”集合 $L_i[f,u]$ 都是相同的。然而，由于我们使用了 k 个不同的权重，这种受限的流向量传播规则仍足以计算 $\min(k, \lambda(s,t))$.

思考这种替代方法所产生影响的一种方式是：现在，对于任意顶点 v 和任意整数 $\ell \le k$，如果分配给 $E_{\text{in}}(v)$ 中各条边的 ℓ 个流向量的某个子集是独立的，那么我们预计，分配给$E_{\text{out}}(v)$ 中各条边的至多 ℓ 个流向量的每个子集也都是独立的。在先前框架中，这一结果甚至对 $\ell > k$ 也成立。通过

通过放宽用于确定流向量的方法，我们得到了一个较弱的条件，但这仍足以解决 k-全点对连通性（APC）问题。

这一修改使**编码** 步骤更加复杂（现在分为两部分：一部分是求矩阵的逆，另一部分是将该逆矩阵与其他矩阵相乘），但总体上加快了速度。为了加快**解码** 步骤，我们采用证明 [1, 定理 5.2] 中的一个观察结果的变体来论证，我们可以假设图中的每个顶点的入度和出度都为 k。根据命题 8 和方程 (3)，这意味着我们可以在 $\tilde{O}(k^\omega n^2)$时间内计算所有顶点对(s,t)的 $\min(k, \lambda(s,t))$。因此，我们的算法的瓶颈~来自**编码** 步骤，其运行时间为 $O((kn)^\omega)$。

3.3 全点对顶点连通度

~我们的出发点是 [1], 中的一个 $O((kn)^\omega)$时间算法，该算法计算所有非边顶点对(s,t)的 $\min(k, \nu(s,t))$。该算法基于第 3.1 节概述的流向量编码方案的一种变体。我们不再为边分配向量，而是为顶点分配流向量（直观上这是合理的，因为我们在 k-APVC 问题中处理的是顶点连通度）。其余构造类似：我们设想向 s 及其出邻居注入一些初始向量，然后在图中传播流，使得最终对于任意顶点 v，分配给 v 的流向量都是分配给 v的入邻居的流向量的随机线性组合。[6]

令 K 为一个 $n \times n$ 矩阵，其行和列由输入图中的顶点索引。对于每一对顶点 (u, v)，如果存在一条从 u 到 v 的边，我们就将 $K[u,v]$ 设为 $\mathbb{F}$ 中的均匀随机元素。否则，$K[u,v] = 0$。这些项恰好对应于流向量框架中随机线性组合所使用的系数。

现在定义矩阵

$$M = (I - K)^{-1}. \tag{4}$$

接下来，我们证明对于任意一对顶点 (s, t)，有

$$\operatorname{rank} M[V_{\text{out}}[s], V_{\text{in}}[t]] = \begin{cases} \nu(s,t) + 1 & \text{if } (s,t) \text{ is an edge} \\ \nu(s,t) & \text{otherwise.} \end{cases} \tag{5}$$

此前，[1, 引理 5 的证明。1] 概述了另一种论证，表明当 (s, t) 不是一条边时，秩 $M[V_{\text{out}}(s)$ $V_{\text{in}}(t)] = \nu(s, t)$。正如我们在附录 B.1 中讨论的，当 (s, t) 是一条边时，这一结论不一定成立。

我们使用方程 (5) 求解 k-APVC。在**编码** 步骤中，我们计算 M。根据~方程 (4)，我们可以通过求一个 $n \times n$ 矩阵的逆来完成，耗时 $O(n^\omega)$。在**解码** 步骤中，根据方程 (5) 和命题 8，我们可以为所有顶点对 (s, t) 计算 $\min(k, \nu(s, t))$，其渐近时间复杂度为

$$\sum_{s,t} (\deg_{\text{out}}(s) \deg_{\text{in}}(t) + k^\omega) = m^2 + k^\omega n^2$$

时间，其中求和遍历图中的所有顶点 s 和 t。对于**解码** 步骤，我们在此得到的运行时间上界过高——如果图中有许多高阶顶点，朴素地计算子矩阵的秩会太慢。

[6] 实际上，这种行为仅对顶点 $v \notin V_{\text{out}}[s]$. 成立。分配给 $V_{\text{out}}[s]$ 中顶点的流向量规则稍微复杂一些，具体取决于初始泵送向量的值。

为避免这种减速， [1] 采用一种简单技巧来降低图中的次数：我们可以添加几层 k 新节点，阻断原图中每个顶点的入边和出边。也就是说，对于 G中的每个顶点 s ，我们添加一个包含 k 个 S 新节点的集合，将 $E_{\text{out}}(s)$中的边替换为从 s 指向 S中所有节点的边，并添加从 S 中的每个节点指向原本位于 $V_{\text{out}}(s)$中的每个顶点的边。类似地，对于 G中的每个顶点 t ，我们添加一个包含k 个 T 新节点的集合，将 $E_{\text{in}}(t)$中的边替换为从 T 中的所有节点指向 t的边，并添加从原本位于 $V_{\text{in}}(t)$中的每个顶点指向 T中每个节点的边。

很容易验证，对于原图中所有满足 (s,t) 不是一条边的顶点对 (s,t)，该变换都能保持 $\min(k,\nu(s,t))$ 的值不变。此外，原图中的所有顶点在新图中的入度和出度都恰好为 k 。因此，现在可以将 **解码** 步骤实现为在 $\tilde{O}(k^\omega n^2)$ 时间内运行。不幸的是，这种构造使图中的顶点数量从 n 增加到 $(2k+1)n$。因此，在 **编码** 步骤中，我们所使用的矩阵 K 不再是 $n\times n$，而是~大小为 $(2k+1)n\times(2k+1)n$。现在，对 $I-K$ 求逆以计算 M需要 $O((kn)^\omega)$ 时间，这就是为什么 [1]的算法具有这一运行时间。

我们的改进

直观地说， [1]用于降低图中次数的修改似乎非常低效。这种变换会使图变大，以便“丢弃”有关大于 k 的连通度值的信息。与其以这种方式修改图，我们是否可以直接修改流向量方案本身，从而加快**解码** 步骤？我们的算法正是这样做的：它实质上修改了流向量矩阵，以模拟前述变换的效果，而无需显式地向图中添加新节点。

我们不直接处理方程 (4) 中的矩阵 M ，而是对每一对 (s,t) 个 f 顶点定义一个 $(k+1)\times(k+1)$ 矩阵

$$M_{s,t}=B_s\left(M[V_{\text{out}}[s],V_{\text{in}}[t]]\right)C_t$$

该矩阵由将 M 的一个子矩阵左乘和右乘小型随机矩阵 B_s 和 C_t 得到，其中分别具有 $k+1$ 行和列。由于 B_s 具有 $k+1$ 行，而 C_t 具有 $k+1$ 列，因此我们可以论证，以高概率，方程 (5) 蕴含

$$\operatorname{rank} M_{s,t}=\begin{cases}\min(k+1,\nu(s,t)+1) & \text{if } (s,t) \text{ is an edge}\\ \min(k+1,\nu(s,t)) & \text{otherwise.}\end{cases}$$

因此，我们可以根据秩 $M_{s,t}$ 的值计算 $\min(k,\nu(s,t))$。这一思路类似于用于高效计算矩阵秩的算法中的预处理方法（参见 [5] 及其中的参考文献）。从概念上讲，我们可以将这种方法视为对流向量框架的一种修改。令 $d=\deg_{\text{out}}(s)$。正如第 3.1 节所述，先前的工作

1. 首先为 d 维空间中的节点 $V_{\text{out}}(s)$分配互不相同的单位向量，然后2. 计算 $V_{\text{in}}(t)$中所有顶点的流向量的秩。

在我们的工作中，我们则

1. 首先生成 $(d+1)$ 个随机的 $(k+1)$ 维向量，赋给 $V_{\text{out}}[s]$ 中的节点，然后2. 计算 $(k+1)$ 个 $V_{\text{in}}[t]$ 中各顶点的流向量的随机线性组合的秩。

这种替代方法足以解决 k-APVC 问题，同时避免了以往工作中缓慢的$O((kn)^\omega)$ **编码** 步骤。

因此，在我们的算法的**解码** 步骤中，我们通过计算$(k+1) \times (k+1)$ 矩阵 $M_{s,t}$ 的秩，以总计 $\tilde{O}(k^\omega n^2)$ 的时间，为每一对顶点 (s,t) 计算 $\min(k, \nu(s,t))$。

我们的**编码** 步骤比之前的工作更复杂，因为我们不仅需要计算逆矩阵 $(I-K)^{-1}$，还必须构造 $M_{s,\ t}$ 矩阵。朴素地分别计算每个 $M_{s,\ t}$ 矩阵速度太慢，因此我们最终采用间接方法，同时计算$M_{s,\ t}$ 矩阵的所有元素，仅需进行 $O(k^2)$ 次 $n \times n$ 矩阵乘法。这需要 $\tilde{O}(k^2 n^\omega)$ 时间，是我们算法的瓶颈。

4 流向量编码

本节中的论证与 [6, Section 2], 中的论证类似，但由于我们处理的是低秩随机矩阵，而非一般随机矩阵，因此需要更为复杂的证明。

固定输入图 G中的一个源点 s。令 $d = \deg_{\text{out}}(s)$ 表示离开 s的边数。令 $e_1, \ldots, e_d \in E_{\text{out}}(s)$ 表示从 s出发的边。

取一个素数 $p = \Theta(m^5)$。令 $\vec{u}_1, \ldots, \vec{u}_d$为 $\mathbb{F}_p^d$中的互异单位向量。

最终，我们将为 G 中的每条边 e 指定一个向量 $\vec{e} \in \mathbb{F}_p^d$，称之为流向量。这些流向量将由某个向量方程组确定。为描述这些方程，我们首先引入一些符号矩阵。

对于每个索引 $i \in [k]$，我们定义一个 $m \times n$ 矩阵 X_i，其行由 G 的边索引，列由 G的顶点索引，并且对于每条边 $e = (u,v)$，元素 $X_i[e,v] = x_{i,ev}$ 是不定元。X_i 中所有不属于此类型的元素均为零。

类似地，我们定义矩阵 $n \times m$ 和 Y_i，其行由 G 的顶点索引，列由 G 的边索引，使得对于每条边 $f = (u,\ v)$，对应的元素 $Y_i[u,f] = y_{i,\ uf}$ 是一个不定元。Y_i 中所有不属于此类型的元素均为零。

令 X 为通过水平拼接 X_i 矩阵形成的 $m \times kn$ 矩阵。类似地，令 Y 为通过垂直拼接 Y_i 矩阵形成的 $kn \times m$ 矩阵。然后我们定义矩阵

$$Z = XY = X_1Y_1 + \cdots + X_kY_k. \tag{6}$$

根据构造，Z 是一个 $m \times m$ 矩阵，其行和列的索引均为 G的边，使得对于任意边 $e = (u,v)$ 和 $f = (v,w)$，我们有

$$Z[e,f] = \sum_{i=1}^{k} x_{i,ev} y_{i,vf} \tag{7}$$

且 Z 的所有其他条目均设置为零。

考虑以下过程。我们为每个变量 $x_{i,ev}$ 和 $y_{i,uf}$赋予来自$\mathbb{F}_p$ 的独立均匀随机值。令 L_i, L, R_i, R和 K 分别为将这些值赋给 X_i, X, Y_i, Y和 Z后得到的 $\mathbb{F}_p$ 上的矩阵。特别地，我们有

$$K = LR. \tag{8}$$

现在，我们为每条边 e 分配一个流向量 $\vec{e} \in \mathbb{F}_p^d$，使其满足以下等式：

12 一种高效的全点对有界边连通度算法

1. 回顾一下， e_1，……， e_d 是从 s 出发的所有边，而 $\vec{u}_1$，……， $\vec{u}_d$ 是 $\mathbb{F}_p^d$ 中的互不相同的单位向量。对于每条边 $e_i \in E_{\text{out}}(s)$，我们要求其流向量满足

$$\vec{e}_i = \left(\sum_{f \in E_{\text{in}}(s)} \vec{f} \cdot K[f, e_i] \right) + \vec{u}_i. \tag{9}$$

2. 对于每条边 $e = (u, v)$ 满足 $u \neq s$，我们要求其流向量满足

$$\vec{e} = \sum_{f \in E_{\text{in}}(u)} \vec{f} \cdot K[f, e]. \tag{10}$$

先验上并不明显满足上述两个条件的流向量存在，但我们将在下文证明它们确实存在（且以高概率成立）。令 H_s 为 G 中各边作为列索引的 $d \times m$ 矩阵，其中每个索引 i 对应的列为 $\vec{u}_i$ e_i，其余列均为零向量。令 F 为 G 中各边作为列索引的 $d \times m$ 矩阵，其列 $F[*, e] = \vec{e}$ 是对应边的流向量。那么公式（9）和（10）可由如下简单的矩阵方程概括。

$$F = FK + H_s. \tag{11}$$

下面的引理表明，我们可以以较高概率求出上面方程中的 F。

▶ **引理 10。** 我们有 $\det(I - K) \neq 0$，其概率至少为 $1 - 1/m^3$。

证明。 由于输入图没有自环，根据方程 (7) 以及紧接其后的讨论，我们知道 $m \times m$ 矩阵 Z 的对角线元素为零。根据方程 (7)， Z 的每个元素都是次数至多为二的多项式，且常数项为零。因此，$\det(I - Z)$ 是一个关于 $\mathbb{F}_p$ 的次数至多为 $2m$ 的多项式，且常数项等于 1。特别地，这个多项式非零。根据施瓦茨-齐佩尔引理（命题 9），$\det(I - K)$ 非零的概率至少为

$$1 - 2m/p \geq 1 - 1/m^3$$

通过设置 $p \geq 2m^4$。 ◀

现在假设 $\det(I-K) \neq 0$（根据引理 10，这种情况以高概率发生）。在此假设下，我们可以在方程 (11) 中求解 F，得到

$$F = H_s(I - K)^{-1} = \frac{H_s \left(\text{adj}(I - K)\right)}{\det(I - K)}. \tag{12}$$

这个方程将使我们能够把流向量集合的秩与输入图中的连通度值联系起来。

▶ **引理 11.** 对于 G 中的任意顶点 t，以至少 $1 - 2/m^3$ 的概率，有

$$\text{rank}\, F[*, E_{\text{in}}(t)] \leq \lambda(s, t).$$

证明。 记 $\lambda = \lambda(s, t)$。从概念上说，该证明通过论证进入 t 的所有边所对应的流向量，都是 (s, t)-割中的边所对应的流向量的线性组合，来完成。 G。

设 C 是一个最小的（s， t）-割。根据门格尔定理， $|C| = \lambda$。

设 S 是从 s 出发且不经过 C中的边可以到达的节点集合，并设 T 是无需经过 C中的边即可到达 t 的节点集合。根据（s, t）-割的定义， S和 T 将 G中的顶点划分开。

令 E' 为边集 $e = (u, v)$，其中 $v \in T$。

设 $K' = K[E', E']$ 和 $F' = F[*, E']$.。最后，令 H' 为一个矩阵，其列以 E' 中的边为索引，使得与边 $e \in C$ 相关联的列为 $\vec{e}$，而所有其他列均等于 $\vec{0}$。

那么，根据公式（9）和（10），我们有

$$F' = F'K' + H'.$$

事实上，对于任意边 $e = (u, v) \in E'$，若 $u \in S$，则 $e \in C$，因此 $H'[*, e] = \vec{e}$，且不可能有边 $f \in E'$ 进入 u，所以 $(F'K')[*, e] = \vec{0}$。反之，若 $u \in T$，则 $H'[*, e] = \vec{0}$，但所有进入 u 的边 f 都属于 E'，因此根据方程（10），我们有 $(F'K')[*, e] = F'[*, e]$，如所需。

仿照引理 10 的证明，我们有 $\det(I - K') \neq 0$，其发生概率至少为 $1 - 1/m^3$。若此事件发生，我们可以在前一个方程中解出 F'，得到

$$F' = H'(I - K')^{-1}.$$

由于 H' 至多有 λ 个非零列，因此秩 $H \leq \lambda$。由上式可知，秩 $F' \leq \lambda$。根据定义， $E_{\text{in}}(t) \subseteq E'$。因此， $F[*, E_{\text{in}}(t)]$ 是 F'的一个子矩阵。结合前面的结果可知，秩 $F[*, E_{\text{in}}(t)] \leq \lambda$，这正是所需结论。所声称的概率界由对 $I - K$ 和 $I - K'$ 同时可逆这一事件取并集界得出。◀

▶ **引理 12.** 对于 G中的任意顶点 t， 以至少 $1 - 2/m^3$的概率，有

$$\operatorname{rank} F[*, E_{\text{in}}(t)] \geq \min(k, \lambda(s, t)).$$

证明。 记 $\lambda = \min(k, \lambda(s, t))$。直观而言，我们的证明表明，从 s 到 t 存在边不相交路径，会使得 $E_{\text{in}}(t)$ 中的某些边被赋予线性无关的流向量（以高概率），进而推出所需的下界。

根据门格尔定理， G 包含 λ 条边不相交路径 $P_1, \ldots, P_\lambda$，从 s到 t。

考虑符号矩阵 X_i 和 Y_i的变量赋值如下。对于每个索引 $i \leq \lambda$ 和边 $e = (u, v)$，如果 e 是 P_i 中的一条边，则将变量 $x_{i,ev} = 1$ 设为 1。类似地，对于每个索引 $i \leq \lambda$ 和边 $f = (u, v)$，如果 f 是 P_i中的一条边，则将变量 $y_{i,uf} = 1$ 设为 1。所有其他变量均设为零。特别地，如果 $i > \lambda$，则 X_i 和 Y_i 的所有元素均设为零。在此赋值下，矩阵 X_iY_i （其行和列均以图中的边为索引）具有如下性质：如果 f是路径 P_i上紧随 e 之后的边，则 $(X_iY_i)[e, f] = 1$，所有其他元素均设为零。

然后根据公式（6），我们看到，在此赋值下，如果 $Z[e, f] = 1$ 和 e 是某条路径 P_i 中相邻的边，则 f；而 Z 的所有其他项均设为零。对于这一特定赋值，由于这些 P_i 是边不相交路径，公式（9）和（10）表明，每条路径 P_i 的最后一条边都被赋予一个不同的 d 维单位向量。这些向量线性无关，因此在这种情况下，秩 $F[*, E_{\text{in}}(t)] = \lambda$。

对于此赋值，这意味着 $F[*, E_{\text{in}}(t)]$ 包含一个 $\lambda \times \lambda$ 满秩子矩阵。设 F' 为 $F[*, E_{\text{in}}(t)]$ 的一个具有此性质的子矩阵。由于 F' 满秩，对于上述赋值，我们有 $\det F' \neq 0$。

现在，在为变量赋值之前，$\operatorname{adj}(I - Z)$ 的每个条目都是次数至多为 $2m$ 的多项式。因此，根据公式（12），$\det F'$ 等于某个次数至多为 $2\lambda m$ 的多项式 P，除以 $(\det(I - Z))^\lambda$。我们知道 P 是一个非零多项式，因为我们看到

此外，对于变量的某些赋值，det F' 非零（如果 P 是零多项式，那么 det F' 在任何赋值下都会计算为零）。

根据引理 10，以至少 $1-1/m^3$ 的概率，对变量进行随机赋值后，det$(I-Z)$ 的取值将为非零值。假设该事件发生，根据施瓦茨-齐佩尔引理（命题 9），通过设置 $p \geq 2m^5$.，对 Z 中的变量进行随机赋值后，det $F' \neq 0$ 以至少 $1-(2\lambda m)/p \geq 1-1/m^3$ 的概率为非零值

因此，根据并集界， $F[*, E_{\text{in}}(t)]$ 的某个 $\lambda \times \lambda$ 子矩阵以至少 $1-2/m^3$ 的概率为满秩。这证明了所需结果。◀

▶ **引理 13**。 固定顶点 s 和 t。定义 $\lambda = 秩（I-K）^{-1}[E_{\text{out}}（s） \ E_{\text{in}}（t）]$. 以至少 $1-4/m^3$ 的概率，有 min（k， λ） $=$min（k， λ（s， t））。

证明。根据 H_s 的定义，结合公式（12），可得

$$F[*, E_{\text{in}}(t)] = (I-K)^{-1}[E_{\text{out}}(s), E_{\text{in}}(t)]. \tag{13}$$

根据引理 11 和 12 的并集界，以至少 $1-4/m^3$ 的概率，不等式 s

$$\lambda = \text{rank } (I-K)^{-1}[E_{\text{out}}(s), E_{\text{in}}(t)] = \text{rank } F[*, E_{\text{in}}(t)] \leq \lambda(s,t)$$

and

$$\lambda = \text{rank } (I-K)^{-1}[E_{\text{out}}(s), E_{\text{in}}(t)] = \text{rank } F[*, E_{\text{in}}(t)] \geq \min(k, \lambda(s,t))$$

同时成立。因此得出所需结果。◀

5 连通性算法

在本节中，我们介绍用于 k-全点对连通性 的算法。

图变换

我们首先按如下方式修改输入图 G 。对于 G中的每个顶点 v ，引入两个新节点 v_{out} 和 v_{in}。将原先属于 G 的每条边(u,v)替换为边$(u_{\text{out}}, v_{\text{in}})$。对于所有$u$ 和 v，添加 k 条从 v 到 v_{out}的平行边，以及 k 条从 v_{in} 到 v的平行边。我们将修改前图中存在的顶点称为原始顶点。

假设 G 原本有 n 个节点和 m 条边。那么，修改后的图有 $n_{\text{new}} = 3n$个节点和 $m_{\text{new}} = m+2kn$ 条边。对于任意原始顶点 s 和 t，新图中从s 到 t 的边不相交路径对应于原图中从 s 到 t 的边不相交路径。此外，对于任意整数 $\ell \leq k$，如果原图中包含 ℓ 条从 s到 t 的边不相交路径，那么新图中也包含 ℓ 条从 s 到 t 的边不相交路径。

因此，对于任意原始顶点 s 和 t，min$(k, \lambda(s,t))$的值在旧图和新图中保持不变。因此，只需在新图上求解 k-全点对连通性。在这个新图中，每个原始顶点的入度和出度都等于 k。此外，对于所有原始顶点 s 和 t，集合 $E_{\text{out}}(s)$与 $E_{\text{in}}(t)$两两不相交。

其他定义

我们使用第 4 节中定义的矩阵，但现在这些矩阵是相对于修改后的图定义的。具体而言，K、 L 和 R 现在分别是维度为 $m_{\text{new}} \times m_{\text{new}}$、 $m_{\text{new}} \times n_{\text{new}}$ 和 $n_{\text{new}} \times m_{\text{new}}$ 的矩阵。

˜定义 L 为矩阵：将所有原始顶点 s 对应的 $L[E_{\text{out}}(s) \ *]$˜ 垂直拼接而成。类似地，定义 R 为矩阵：将所有原始顶点 t 对应的 $R[*, E_{\text{in}}(t)]$ 水平拼接而成。

该算法

利用上述定义，我们在算法 1 中介绍求解 k-全点对连通性的方法。

算法 1 我们用于求解 k-全点对连通性问题的算法。

1: 计算 $n_{\text{new}} \times n_{\text{new}}$ 矩阵 $(I - RL)^{-1}$。 2: 计算 $kn_{\text{new}} \times kn_{\text{new}}$ 矩阵 $M = \tilde{L}(I - RL)^{-1}\tilde{R}$。 3: 对于每对原始顶点 (s, t)，计算

秩 $M[E_{\text{out}}(s)\ E_{\text{in}}(t)]$，并将其作为 $\min(k, \lambda(s, t))$ 的值输出。

▶ **定理14**。 以至少 $1 - 5/(m_{\text{new}})$ 的概率，算法 *1* 能正确求解 k-全点对连通性问题。

证明。 根据引理 10，以至少 $1 - 1/(m_{\text{new}})^4$ 的概率，矩阵 $I - K$ 可逆。接下来，我们假设 $I - K$ 可逆。

根据引理 13，以至少 $1 - 4/(m_{\text{new}})^3$，的概率，我们有

$$\text{rank}(I - K)^{-1}[E_{\text{out}}(s), E_{\text{in}}(t)] = \min(k, \lambda(s, t)) \tag{14}$$

对于任意给定的原始顶点 s 和 t，通过对所有 $n^2 \leq (m_{\text{new}})^2$ 对原始顶点（s，t）应用并集界，我们可以看出，方程（14）以至少 $1 - 4/(m_{\text{new}})$ 的概率对所有原始顶点 s 和 t 均成立。

由于 $I - K$ 是可逆的，根据方程 (8) 和命题 6，我们有

$$(I - K)^{-1} = (I - LR)^{-1} = I + L(I - RL)^{-1}R.$$

在方程（14）中使用上述方程可知，对于原始顶点 s 和 t，量 $\min(k, \lambda(s, t))$ 等于其秩。

$$(I + L(I - RL)^{-1}R)[E_{\text{out}}(s), E_{\text{in}}(t)] = L[E_{\text{out}}(s), *](I - RL)^{-1}R[*, E_{\text{in}}(t)]$$

其中我们利用了这样一个事实： $I[E_{\text{out}}(s)\ E_{\text{in}}(t)]$ 是全零矩阵，因为在修改后的图中，对于所有原始顶点对（s, t），$E_{\text{out}}(s)$和 $E_{\text{in}}(t)$都是不相交集合。

根据 L 和 R的定义，上述方程和讨论意味着

$$\min(k, \lambda(s, t)) = \text{rank}\ (\tilde{L}(I - RL)^{-1}\tilde{R})[E_{\text{out}}(s), E_{\text{in}}(t)] = \text{rank}\, M[E_{\text{out}}(s), E_{\text{in}}(t)]$$

这证明了算法 1 输出了正确的答案。

对 $I - K$ 可逆以及对所有 (s, t) 方程（14）均成立这些事件应用并集界，可知算法 1 至少以 $1 - 5/(m_{\text{new}})$ 的概率正确。◀

现在，我们已经准备好证明主要结果。

▶**定理4**。 对于任意正整数 k， k-全点对连通性可在 $O((kn)^\omega)$ 时间内求解。

证明。 根据定理14，算法 1 正确地解决了 k-全点对连通性（APC）问题。现在我们论证算法 1 可以通过实现，使其运行时间达到 $O((kn)^\omega)$ 。

在算法 1 的第 1 步中，我们需要计算 $(I - RL)^{-1}$。

根据 R 和 L 的定义，我们可以看出，要计算 RL，只需针对每一对指标 $(i, j) \in [k]^2$，计算乘积 $R_i L_j$。矩阵 $R_i L_j$ 是 $n_{\text{new}} \times n_{\text{new}}$，而它的

行和列通过图中的顶点进行索引。给定顶点 u 和 v，令 $E(u,v)$ 表示从 u 到 v 的平行边集合。根据 R_i 和 L_j 矩阵的定义，我们可以看到，对于任意顶点 u 和 v，有

$$(R_iL_j)[u,v] = \sum_{e\in E(u,v)} R_i[u,e]L_j[e,v]. \tag{15}$$

如第 2 节所述，对于所有顶点 u 和 v，我们可以假设 $|E(u,v)| \le k$。

对于每个顶点 u，定义 $k \times \deg_{\text{out}}(u)$ 矩阵 R'_u，其行由 $[k]$ 索引，列由从 u 出发的边索引，定义为

$$R'_u[i,e] = R_i[u,e]$$

对于所有 $i \in [k]$ 和 $e \in E_{\text{out}}(u)$。

类似地，对于每个顶点 v，定义 $\deg_{\text{in}}(v) \times k$ 矩阵 L'_v，定义为

$$L'_v[e,j] = L_j[e,v]$$

对于所有 $e \in E_{\text{in}}(v)$ 和 $j \in [k]$。

最后，对于每一对顶点 (u,v)，定义 $R'_{uv} = R'_u[*, E(u,v)]$ 和 $L'_{uv} = L'_v[E(u,v), *]$。然后根据公式（15），我们有

$$(R_iL_j)[u,v] = R'_{uv}L'_{uv}[i,j].$$

因此，要计算 R_iL_j 乘积，只需在 $O(km_{\text{new}})$ 时间内构建 R'_u 和 L'_v 矩阵，然后计算 $R'_{uv}L'_{uv}$ 乘积。我们可以通过计算成对 $k \times k$ 矩阵的 $(n_{\text{new}})^2$ 乘积来实现这一点。由于对于每一对顶点 (u,v)，从 u 到 v，$km_{\text{new}} \le k^2n^2$最多有 k 条平行边，因此我们可以在 $\tilde{O}(n^2k^\omega)$ 时间内计算所有 R_iL_j 乘积，从而得到整个矩阵 RL。

然后，我们可以通过修改 RL 的 $O(kn)$ 个条目来计算 $I-RL$。最后，根据命题 7，我们可以在 $\tilde{O}((kn)^\omega)$ 时间内计算 $(I-RL)^{-1}$。

因此 ov 总体而言，算法 1 的步骤 1 用时为 $\tilde{O}((kn)^\omega)$ tim e.

在算法 1 的第 2 步中，我们需要计算 $M = \tilde{L}(I-RL)^{-1}\tilde{R}$。

回想一下，L 是一个 $kn \times n_{\text{new}}$ 矩阵。根据定义，L 的每一行都只有一个非零元素。同样，R 是一个 $n_{\text{new}} \times kn$ 矩阵，每一列都只有一个非零元素。

因此，我们可以计算 M，并完成算法 1 的步骤 2。 $\tilde{O}((kn)^\omega)$ 时间。

最后，在算法 1 的第 3 步中，我们需要计算

$$\operatorname{rank} M[E_{\text{out}}(s), E_{\text{in}}(t)] \tag{16}$$

对于图中的每一对原始顶点 (s,t)。在修改后的图中，每个原始顶点的入度和出度均为 k，因此每个 $M[E_{\text{out}}(s), E_{\text{in}}(t)]$ 都是一个 $k \times k$ 矩阵。对于任意固定的 (s,t)，根据命题 8，我们可以在 $O(k^\omega)$ 时间内计算 $M[E_{\text{out}}(s), E_{\text{in}}(t)]$ 的秩。

因此，我们可以针对所有原始顶点对 (s,t) 计算方程 (16) 中的秩，并在 $\tilde{O}(k^\omega n^2)$ 时间内完成算法 1 的第 3 步。

因此，我们可以在总体时间 $O((kn)^\omega)$ 内解决 k-全点对连通性，正如所述。 ◀

6 顶点连通度编码

取一个素数 $p = \tilde{\Theta}(n^5)$。令 K 为一个 $n \times n$ 矩阵，其行和列均以 G的顶点为索引。对于每一对顶点 (u,v)，若 (u,v) 是 G中的一条边，则令 $K[u,v]$ 为 $\mathbb{F}_p$中的一个均匀随机元素。否则，$K[u,v] = 0$。

回顾第 2 节，对于给定的顶点 $v \in G$，令 $V_{\text{in}}[v] = V_{\text{in}}(v) \cup \{v\}$ 为由 v 及 v的所有入邻点组成的集合，并令 $V_{\text{out}}[v] = V_{\text{out}}(v) \cup \{v\}$ 为由v 及 v的所有出邻居组成的集合。下述命题[7] 基于 [6, 第 2]节和 [1, 第 5]节中的思想。我们在附录B.2中给出该结果的完整证明。

▶ **命题15**。对于 G 中任意顶点 s 和 t，以至少 $1 - 3/n^3$ 的概率，矩阵 $(I - K)$ 可逆，并且我们有

$$\text{rank } (I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}[t]] = \begin{cases} \nu(s,t)+1 & \textit{if } (s,t) \textit{ is an edge} \\ \nu(s,t) & \textit{otherwise.} \end{cases}$$

命题15表明，只需计算某些子矩阵 $(I - K)^{-1}$ 的秩，就能在 G 中计算顶点连通度。然而，这些子矩阵可能会相当大；若要快速计算顶点连通度，这就不理想了。为了解决这个问题，我们将说明如何缩小 $(I - K)^{-1}$ 的规模，同时保留关于 $\nu(s, t)$ 的值的相关信息。

▶ **引理16**。设 M 是一个定义在 $\mathbb{F}_p$ 上的 $a \times b$ 矩阵。设 Γ 是一个 $(k+1) \times a$ 矩阵，其元素从 $\mathbb{F}_p$ 中均匀随机选取。那么，以至少 $1 - (k+1)/p$ 的概率，我们有

$$\text{rank } \Gamma M = \min(k+1, \text{rank } M).$$

证明。由于 ΓM 有 $k+1$ 行，秩$(\Gamma M) \leq k+1$。

同理，由于 ΓM以 M为因子，rank$(\Gamma M) \leq$rank M。因此

$$\text{秩 } \Gamma M \leq \min(k+1, \text{秩 } M)。 \tag{17}$$

因此，只需证明 ΓM的秩至少为 $\min(k+1,$ rank $M)$。

令 $r = \min(k+1,$ rank $M)$。那么，存在行索引和列索引的子集 S和 T，使得 $|S| = |T| = r$ 和 $M[S, T]$的秩为 r。现在，设 U是 Γ中任意一个包含 r行的集合。考虑矩阵 $M' = (\Gamma M)[U, T]$。

我们可以将 M' 的每个元素视为关于 Γ各元素的次数至多为 1 的多项式。这意味着 det M'是关于 Γ各元素的次数至多为 r的多项式。此外，如果子矩阵 $\Gamma[U, T] = I$恰好是单位矩阵，那么 $M' = M[S, T]$。这说明 det M'是关于 Γ各元素的非零多项式，因为对 Γ 各元素进行某种赋值后，该多项式的取值 det $M[S, T] \neq 0$非零（这里用到了 $M[S, T]$满秩这一事实）。

因此，根据施瓦茨-齐佩尔引理（命题 9），矩阵 ΓM 的秩至少为 r，其概率至少为 $1 - r/p$。

结合方程 (17)，即可得到所需结果。 ◀

现在，我们为图中的每个顶点 u 分配一个 $(k+1)$ 维列向量 b_u和一个 $(k+1)$ 维行向量 $\vec{c_u}$。

令 B为将所有 b_u向量水平拼接而成的 $(k+1) \times n$矩阵，并令 C为将所有 $\vec{c_u}$向量垂直拼接而成的 $n \times (k+1)$矩阵。对于每一对不同的顶点 (s, t)，定义 $(k+1) \times (k+1)$ 矩阵

$$M_{s,t} = B[*, V_{\text{out}}[s]] \left((I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}[t]] \right) C[V_{\text{in}}[t], *]. \tag{18}$$

以下结果是我们用于 k-APVC 的算法的基础。

[7] 此处陈述的结果不同于 [1, 第 5] 节中使用的类似结论。我们将在附录 B.1 中讨论这一差异及相关的细微问题。

▶ **引理 17.** 对于 G 中的任意顶点 s 和 t，其概率至少为 $1-5/n^3$，我们有

$$\operatorname{rank} M_{s,t} = \begin{cases} \min(k+1, \nu(s,t)+1) & \textit{if } (s,t) \textit{ is an edge} \\ \min(k+1, \nu(s,t)) & \textit{otherwise.} \end{cases}$$

证明。 固定顶点 s 和 t。那么，根据命题15，我们有

$$\operatorname{rank} (I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}[t]] = \begin{cases} \nu(s,t)+1 & \text{if } (s,t) \text{ is an edge} \\ \nu(s,t) & \text{otherwise} \end{cases}$$

其概率至少为 $1-3/n^3$。假设上述方程成立。

那么，在引理16中令 $\Gamma = B[*, V_{\text{out}}[s]]$ 和 $M = (I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}[t]]$，可知其概率至少为 $1-1/n^3$，并且有

$$\operatorname{rank} B[*, V_{\text{out}}[s]](I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}(t)] = \begin{cases} \min(k+1, \nu(s,t)+1) & \text{if } (s,t) \text{ is an edge} \\ \min(k+1, \nu(s,t)) & \text{otherwise.} \end{cases}.$$

假设上述方程成立。

最后，在引理16中令 $\Gamma = C^\top[*, V_{\text{in}}(t)]$ 和 $M = (B[*, V_{\text{out}}[s]](I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}(t)])^\top$，并进行转置，可知其概率至少为 $1-1/n^3$，并且有

$$\operatorname{rank} B[*, V_{\text{out}}[s]] \left((I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}(t)]\right) C[V_{\text{in}}(t), *] = \min(k+1, \nu(s,t)+1)$$

如果存在一条从 s 到 t的边，并且

$$\operatorname{rank} B[*, V_{\text{out}}[s]] \left((I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}(t)]\right) C[V_{\text{in}}(t), *] = \min(k+1, \nu(s,t))$$

否则。因此，根据并集界，所需结果以至少 $1-5/n^3$ 的概率成立。◀

7 顶点连通度算法

令 A 为带自环的图 G 的邻接矩阵。也就是说，A 是一个 $n \times n$ 矩阵，其行和列由 G 的顶点索引；对于任意一对顶点 (u, v)，若 $v \in V_{\text{out}}[u]$（等价地，$u \in V_{\text{in}}[v]$），则 $A[u,v] = 1$；否则 $A[u,v] = 0$。

回顾 b_u 和 $\vec{c}_u$ 向量，以及第 6 节中的 K，B，C 和 $M_{s,t}$ 矩阵的定义。对于每个 $i \in [k+1]$，令 P_i 为 $n \times n$ 对角矩阵，其行和列由 G 的顶点索引，且满足 $P_i[u,u] = \vec{b}_u[i]$。类似地，令 Q_i 为 $n \times n$ 对角矩阵，其行和列由 G 的顶点索引，且满足 $Q_i[u,u] = \vec{c}_u[i]$。

基于这些定义，我们在算法 2 中给出求解 k-APVC 的方法。

算法 2 我们用于求解 k-APVC 的算法。

1: 计算 $n \times n$ 矩阵 $(I-K)^{-1}$.2: 对于每对 $(i,j) \in [k+1]^2$ 索引，计算 $n \times n$ 矩阵

$$D_{ij} = AP_i(I-K)^{-1}Q_jA^\top.$$

3: 对于每一对顶点 (s, t)，令 $F_{s,t}$ 为一个 $(k+1) \times (k+1)$ 矩阵，其 (i, j) 项等于 $D_{ij}[s,t]$。如果 (s, t) 是一条边，则输出 (秩 $F_{s,t}$) -1 作为 $\min(k, \nu(s, t))$ 的值。否则，输出 $\min(k,$ 秩 $F_{s,t})$ 作为 $\min(k, \nu(s, t))$ 的值。

算法 2 的主要思想是利用引理 17，将计算给定顶点对 (s, t) 的 $\min(k, \nu(s, t))$，转化为计算相应 $(k+1) \times (k+1)$ 的秩。

矩阵 $M_{s,\ t}$。为了提高这一方法的效率，我们通过一种略为间接的论证，同时计算所有 $M_{s,\ t}$ 矩阵的各个元素。

▶ **定理 18.** 以至少 $1 - 5/n$ 的概率，算法 *2* 能够正确求解 *k-APVC*。

证明。 我们利用以下断言证明算法 2 的正确性。

▷ 断言 19. 对于所有指标对 $(i,\ j) \in [k+1]^2$ 以及所有顶点对 $(s,\ t)$，有

$$M_{s,t}[i,j] = D_{ij}[s,t],$$

其中，D_{ij} 是算法 2 第 2 步中计算得到的矩阵。

证明。 通过展开 D_{ij} 在算法 2 第 2 步中的表达式，我们有

$$D_{ij}[s,t] = \sum_{u,v} A[s,u]P_i[u,u]\left((I-K)^{-1}[u,v]\right)Q_j[v,v]A[v,t],$$

其中求和是对图中的所有顶点 u, v 进行的（这里，我们利用了 P_i 和 Q_j 是对角矩阵这一事实）。根据 A、P_i 和 Q_j 矩阵的定义，我们有

$$D_{ij}[s,t] = \sum_{\substack{u\in V_{\text{out}}[s]\\ v\in V_{\text{in}}[t]}} \vec{b}_u[i]\left((I-K)^{-1}[u,v]\right)\vec{c}_v[j]. \tag{19}$$

另一方面，根据方程 (18) 对 $M_{s,t}$ 的定义可知

$$M_{s,t}[i,j] = \sum_{\substack{u\in V_{\text{out}}[s]\\ v\in V_{\text{in}}[t]}} B[i,u]\left((I-K)^{-1}[u,v]\right)C[v,j].$$

由于 $B[i,u] = \vec{b}_u[i]$ 和 $C[v,j] = \vec{c}_v[j]$，上面的方程和方程 (19) 表明 t

$$M_{s,t}[i,j] = D_{ij}[s,t]$$

对所有 (i,j) 和 (s,t) 均成立，正如所需。 ◁

根据断言 19，算法 2 第 3 步中计算得到的矩阵 $F_{s,t}$ 等于 $M_{s,t}$。因此，根据引理 17，对于任意固定的顶点对 (s,t)，我们有

$$\operatorname{rank} F_{s,t} = \begin{cases} \min(k+1, \nu(s,t)+1) & \text{if } (s,t) \text{ is an edge} \\ \min(k+1, \nu(s,t)) & \text{otherwise.} \end{cases} \tag{20}$$

其概率至少为 $1 - 5/n^3$。然后，对所有顶点对 (s,t) 应用并集界，可知方程 (20) 对所有顶点对 (s,t) 均成立的概率至少为 $1 - 5/n$。

假设此事件发生。那么，如果 (s,t) 是一条边，根据方程 (20)，我们会正确地返回

$$(\operatorname{rank} F_{s,t}) - 1 = \min(k+1, \nu(s,t)+1) - 1 = \min(k, \nu(s,t))$$

作为这对顶点的答案。

类似地，如果 (s,t) 不是一条边，根据方程 (20)，我们会正确地返回

$$\min(k, \operatorname{rank} F_{s,t}) = \min(k, k+1, \nu(s,t)) = \min(k, \nu(s,t))$$

作为这对顶点的答案。这就证明了我们所需的结果。 ◀

定理 18 得证后，我们就可以证明关于顶点连通度的结果。

▶**定理 5**。对于任意正整数 k， *k-APVC* 可在 $\tilde{O}(k^2n^\omega)$ 时间内求解。

证明。由定理 18 可知，算法 2 能正确求解 k-APVC问题。下面说明，算法 2 可以在 $\tilde{O}(k^2n^\omega)$ 时间内实现。

在算法 2 的第 1 步中，我们需要计算 $(I-K)^{-1}$。由于 K 是一个 $n\times n$ 矩阵，根据命题 7，我们可以在 $O(n^\omega)$ 时间内完成这一步。

在算法 2 的第 2 步中，我们需要对每一对 $(i,\ j)\in[k+1]^2$ 计算 D_{ij}。对于每个固定的数对 $(i,\ j)$，D_{ij} 矩阵定义为五个已知其元素的 $n\times n$ 矩阵的乘积，因此这一步总共需要 $\tilde{O}(k^2n^\omega)$ 时间。

在算法 2 的第 3 步中，我们需要构造每个 F_{st} 矩阵并计算其秩。由于每个 F_{st} 矩阵的维度为 $(k+1)\times(k+1)$，且只需读取我们已计算出的 D_{ij} 矩阵中的元素即可填充其各项，根据命题 8，这一步可在 $\tilde{O}(k^\omega n^2)$ 时间内完成。

将各步骤的运行时间相加，并注意到 $k\le n$，可知算法 2 能在 $\tilde{O}(k^2n^\omega)$ 时间内解决 k-APVC，正如所述。◀

8 结论

本文提出了分别以 $\tilde{O}((kn)^\omega)$ 和 $\tilde{O}(k^2n^\omega)$ 时间解决 k-全点对连通性和 k-APVC 的算法。关于这些问题的精确时间复杂度，仍有许多开放问题。以下列出一些我们特别感兴趣的开放问题：

1. 与我们的工作最相关的问题：能否更快地解决 k-全点对连通性或 k-APVC？能否在 $\tilde{O}(k^2n^\omega)$ 时间内解决 k-全点对连通性，与我们解决 k-APVC 的算法一样快？是否存在一些中等规模的参数值 $k\ge n^{\Omega(1)}$，使得这些问题能够在 $O(n^\omega)$ 时间内解决，从而达到常数 k时的运行时间？ 2. 能否为 k-全点对连通性和 k-APVC 得到更好的条件下界？目前，没有条件下界能够排除这些问题例如在 $\tilde{O}(kn^\omega)$ 时间内得到解决的可能性。对于全点对连通性和 APVC 问题，能否将已知的 $n^{\omega(1,2,1)-o(1)}$条件下界改进[8] 为 $n^{4-o(1)}$ 条件下界？ 3. 最近，[13] 证明了 APVC 问题存在一个非确定性验证器，其运行时间为 $O(n^{\omega(1,2,1)})$。全点对连通性是否存在运行时间相同的非确定性验证器？对于 k-全点对连通性和 k-APVC 问题，是否存在运行速度快于定理 4 和定理 5 中算法的非确定性验证器？

参考文献

1 Amir Abboud、Loukas Georgiadis、Giuseppe F. Italiano、Robert Krauthgamer、Nikos Parotsidis、Ohad Trabelsi、Przemysław Uznański 和 Daniel Wolleb-Graf。更快的全点对有界最小割算法。载于第*46*届国际自动机、语言与编程研讨会，*ICALP 2019*，*2019*年*7*月*9*日至*12*日，希腊帕特雷，LIPIcs第*132*卷，第7:1–7:15页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2019年。 doi:10.4230/LIPIcs.ICALP.2019.7。2 Amir Abboud、Robert Krauthgamer 和 Ohad Trabelsi。APMF $<$ APSP？近二次时间内无权图的Gomory-Hu树。载于第*62*届*IEEE*计算机科学基础年会，*FOCS 2021*，美国科罗拉多州丹佛，*2022*年*2*月*7*日至*10*日，第1135–1146页。IEEE，2021年。 doi:10.1109/FOCS52979.2021.00112。

[8] 有一些证据表明，要证明更好的下界可能并不容易 [13]。

3 Josh Alman 和 Virginia Vassilevska Williams。一种改进的激光法与更快的矩阵乘法。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522–539页。SIAM，2021年。 doi:10.1137/1.9781611976465.32。4 Li Chen、Rasmus Kyng、Yang P. Liu、Richard Peng、Maximilian Probst Gutenberg 和 Sushant Sachdeva。近线性时间内的最大流与最小费用流。载于2022年IEEE第63届计算机科学基础年会（FOCS），第612–623页，2022年。 doi:10.1109/FOCS54457.2022.00064。5 Ho Yee Cheung、Tsz Chiu Kwok 和 Lap C hi Lau。快速矩阵秩算法及其应用。ACM期刊，60(5):1–25，2013年10月。 doi:10.1145/2528404。6 Ho Yee Cheung、Lap Chi Lau 和 Kai Man Leung。图连通度、网络编码与扩展图。SIAM计算杂志，42(3):733–751，2013年1月。 doi:10.1137/110844970。7 M. J. Fischer 和 A. R. Meyer。布尔矩阵乘法与传递闭包。载于第12届开关与自动机理论年会（SWAT 1971）。IEEE，1971年10月。doi:10.1109/swat.1971.4。8 François Le Gall 和 Florent Urrutia。利用Coppersmith-Winograd张量的幂改进矩形矩阵乘法。载于第29届ACM-SIAM离散算法研讨会论文集，SODA '18，第1029–1046页，美国，2018年。工业与应用数学学会。9 Loukas Georgiadis、Daniel Graf、Giuseppe F. Italiano、Nikos Parotsidis 和 Przemysław Uznański。在 $O(n^{\omega}logn)$ 时间内实现全点对二可达性。载于第44届自动机、语言与程序设计国际研讨会（ICALP 2017），LIPIcs第80卷，第74:1–74:14页，德国达格施图尔，2017年。Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik。URL： http://drops.dagstuhl.de/opus/volltexte/2017/7451， doi:10.4230/LIPIcs.ICALP.2017.74。10 Zhiyi Huang、Yaowei Long、Thatchaphol Saranurak 和 Benyu Wang。点连通性问题的紧条件下界，2022年。URL: https://arxiv.org/abs/2212.00359，doi:10.48550/ARXIV.2212.00359。11 Robert Krauthgamer 和 Ohad Trabelsi。全点对最大流的条件下界。ACM算法汇刊，14(4)，2018年8月。 doi:10.1145/3212510。12 Rajeev Motwani 和 Prabhakar Raghavan。随机化算法。剑桥大学出版社，1995年8月。 doi:10.1017/cbo9780511814075。13 Ohad Trabelsi。（几乎）排除全点对最大流的SETH下界，2023年。URL:https://arxiv.org/abs/2304.04667，doi:10.48550/ARXIV.2304.04667。14 Xiaowei Wu 和 Chenzi Zhang。计算有向图中所有低s-t边连通度的高效算法。载于2015年计算机科学数学基础，第577–588页。Springer Berlin Heidelberg，2015年。 doi:10.1007/978-3-662-48054-0_48。

细粒度复杂性中的一个猜想

在布尔矩阵乘法（BMM）问题中，给定元素属于{0,1}的 $n \times n$ 矩阵 A 和 B，任务是针对每一对 $(i, j) \in [n]^2$，判断是否存在索引 k 使得 $A[i, k] = B[k, j] = 1$。利用矩阵乘法，我们可以在 $O(n^{\omega})$时间内解决 BMM。

布尔矩阵乘法假设[9] 认为这基本上是最优的，并断言不存在常数 $\delta > 0$，使得布尔矩阵乘法能够在 $O(n^{\omega-\delta})$ 时间内求解。

[9] 文献中也提到了组合布尔矩阵乘法假设，这是一种非正式猜想，认为不存在任何"组合"算法能在 $O(n^{3-\delta})$时间内完成 BMM，其中 $\delta > 0$为任意常数。

设 k 为正整数。在 kSAT 问题中，给定一个关于 n 个变量的 k-CNF（可写成若干子句合取的布尔公式，其中每个子句至多是 k 个布尔文字的析取），任务是确定是否存在一个变量赋值，使其满足该 k-CNF。

强指数时间假设（SETH）猜想，对于任意常数 $\delta > 0$，存在某个正整数 k，使得 kSAT 无法在 $2^{(1-\delta)n}\mathrm{poly}(n)$ 时间内求解。

在 4-团问题中，给定一个图 G，任务是判断其中是否包含一个由四个顶点组成的团（即四个两两相邻且互不相同的顶点）。

令 $\omega(1,2,1)$ 为最小的正实数，使得我们可以在 $n^{\omega(1,2,1)+o(1)}$ 时间内将一个 $n \times n^2$ 矩阵与一个 $n^2 \times n$ 矩阵相乘。已知 4-团问题可以在 $O\left(n^{\omega(1,2,1)}\right)$ 时间内求解。4-团猜想[10] 断言，这一运行时间本质上是最优的，也就是说，对于任意常数 $\delta > 0$，4-团问题都无法在 $O\left(n^{\omega(1,2,1)-\delta}\right)$ 时间内求解。

B 顶点连通度编码方案

在附录 B.1 中，我们介绍 [1] 在计算顶点割方面取得的结果，并解释 [1] 中的主张与本文论证的不同之处。在附录B.2中，我们给出命题15的证明。

B.1 对先前顶点割算法的讨论

~在 [1], 中，作者提出了一种时间复杂度为 $O((kn)^{\omega})$ 的算法，用于求解 k-有界全点对最小顶点割问题。给定一个包含 n 个顶点的有向图 G，该问题要求对每一对顶点（s, t），若最小（s, t）顶点割的大小小于 k，则返回其大小。对于最小（s, t）顶点割大小至少为 k 的顶点对（s, t），我们只需返回值 k。

当（s, t）不是一条边时，门格尔定理表明，最小（s, t）顶点割的大小等于从 s 到 t 的内部顶点不相交路径的最大数量。因此，对于此类顶点对（s, t），k-有界全点对最小顶点割问题只需像 k-APVC 问题那样计算 $\min(k, \nu(s, t))$。

然而，当 (s,t) 是一条边时，正如第 2 节所讨论的，不存在 (s,t)-顶点割。这是因为无论删除 s 和 t 以外的哪些顶点，所得图中始终存在一条从 s 到 t、长度为一的路径。

在这种情况下，将“最小 (s,t)-顶点割的大小”定义为 ∞ 可能是合理的。按照这一约定，对于作为边的顶点对 (s,t)，我们只需在 k-有界全点对最小顶点割问题中返回 k 的值。这正是 [1] 的算法所做的。

更具体地说，[1, 引理 5 的证明.1] 中概述的论证表明，

$$\operatorname{rank}\,(I-K)^{-1}[V_{\mathrm{out}}(s), V_{\mathrm{in}}(t)] \tag{21}$$

以高概率等于所有非边的顶点对 (s,t) 的最小顶点割大小（其中 K 的定义与第3.3节中的相同）。

来自 [1, 定理 5 的证明。2] 的算法首先修改图，然后针对新图计算每一对原始顶点（s, t）的公式（21）的值，以求解 k-有界全点对最小顶点割问题。

[10]该猜想在文献中也有一个非正式的对应说法，即对于任意常数 $\delta > 0$，不存在运行时间为 $O(n^{4-\delta})$ 的 4-Clique“组合”算法。

如第3.3节所述，该图通过以下方式进行变换：对每个顶点 s ，引入一组 S k 个新节点；添加从 s 到 S中每个节点的边；添加从S 中的所有节点到原图中 s 的所有出邻居的边；并删除 s 到其原有出邻居的所有边。类似地，该变换还对每个顶点 t 引入一组 T k个新节点；添加从 t 的所有入邻居到 T中各节点的边；添加从 T 中的所有节点到 t的边；并删除原图中从其入邻居指向 t 的所有边。

如果原图中 (s, t) 不是一条边，那么新图中 (s, t) 仍然不是一条边。在这种情况下，新图中从 s 到 t 的顶点连通度结果为 $\min(k, \nu(s, t))$。由于 (s, t) 不是一条边， $\nu(s, t)$ 的值等于最小 (s, t)-顶点割的大小。因此，在这种情况下，返回公式（21）的值即可得到 k-有界全点对最小顶点割问题的正确答案。

现在假设原图中 (s, t) 是一条边。那么新图中 (s, t) 将不是一条边，因为 s 的新出邻居与 t 的新入邻点互不相同。然而，该变换确保新图中存在从 s 到 t 的 k 条内部顶点不相交路径，因为 s 的每个 k 个新出邻居都与 t 的每个 k 个新入邻点之间有一条边。因此，在这种情况下，返回公式（21）的值就等同于返回值 k。

因此，当（s, t）是一条边时， [1, 证明定理 5.2] 的算法能够为 k-有界全点对最小顶点割问题给出正确答案；按照约定，当（s, t）是一条边时，最小（s, t）-顶点割的大小为 ∞。

˜总而言之，来自 [1] 的算法在 $O((kn)^\omega)$ 时间内运行，并针对所有满足 (s,t) 不是一条边的顶点对 (s,t)，计算 $\min(k, \nu(s,t))$；但对于 (s,t) 是一条边的顶点对，则不会返回有关 $\nu(s,t)$ 值的任何信息。

从顶点割到顶点连通度

公式（21）中的量与我们在命题15中使用的表达式不同。在命题15中，我们用 $V_{\text{out}}[s]$ 和 $V_{\text{in}}[t]$ 对行和列进行索引，而不是分别用 V_{out}（s）和 V_{in}（t）。为了求解 k-APVC，我们需要使用一个不同于公式（21）中所用的子矩阵，因为当（s, t）是一条边时，公式（21）中的表达式不一定等于 ν（s, t）。

例如，假设 G 是一个图，其中 $\deg_{\text{in}}(s) = 0$， $\deg_{\text{out}}(t) = 1$，存在一条从 s 到 t的边，并且 $\nu(s, t) = 1$。那么， [1, 引理 5 的证明1] （使用第3.3节的术语）中的证明概要建议，将单位向量泵送到 $V_{\text{out}}(s)$ 中的节点，使用这些初始向量确定所有节点的流向量，然后计算分配给 $V_{\text{in}}(t)$ 中节点的流向量的秩。在此例中，由于 s 的入度为零，且没有初始向量被泵送到该节点，因此为其分配的流向量为 $\vec{s} = \vec{0}$。由于 $V_{\text{in}}(t) = \{s\}$ 在此例中， $V_{\text{in}}(t)$ 中流向量的秩仅为零，尽管 $\nu(s, t) = 1$。

当 (s, t) 是图的一条边时，由 [1, 引理 5 的证明1]所提出的证明中会更普遍地产生这一问题。直观地说，这是因为从 s 到 t 的最大内部顶点不相交路径集合总会包含一条由从 s 到 t 的单条边组成的路径；但是，如果我们不在 s 处泵出一个向量，这条路径就不会对进入 t的流向量的秩作出贡献。

为了解决这一问题并正确计算所有顶点对的顶点连通度，我们对公式（21）进行了适当修改（在源点 s 处泵出一个初始向量，并允许分配给 t 的流向量计入秩），这就是为什么命题15的表述涉及计算另一个子矩阵的秩。

上述问题在处理边连通度时不会出现，根本原因在于此时总存在一组边，移除它们即可破坏所有从 s 到 t 的路径。

B.2 顶点连通度编码的证明

本节给出命题15的证明，并在下文重述其陈述。

▶ **命题15**。 对于 G中的任意顶点 s 和 t，以至少 $1-3/n^3$的概率，矩阵$(I-K)$ 可逆，且有

$$\text{rank } (I-K)^{-1}[V_{\text{out}}[s], V_{\text{in}}[t]] = \begin{cases} \nu(s,t)+1 & \textit{if } (s,t) \textit{ is an edge} \\ \nu(s,t) & \textit{otherwise.} \end{cases}$$

我们对命题15的证明与 [1, 第 5] 节中的概要相似，并基于 [6, 第 2], 节中的思路，但作了若干修改，以解决处理顶点连通度时出现的问题。

在输入图 G中固定一个源点 s 和目标节点 t 。记 $v_0 = s$。假设 s 的出度为 $d = \deg_{\text{out}}(s)$。令 $v_1, \ldots, v_d$ 表示 s 在 $V_{\text{out}}(s)$中的出邻居。

取一个素数 $p = \Theta(n^5)$（这是第 6 节中的同一个素数 p）。令 $\vec{u}_0, \ldots, \vec{u}_d$ 表示 $\mathbb{F}_p^{d+1}$中的互异单位向量。

最终，我们将为 v 中的每个顶点分配一个向量 $\vec{v} \in \mathbb{F}_p^{d+1}$，并将其称为 流向量。这些流向量由一个向量方程组确定。为了描述这些方程，我们首先需要引入一些符号矩阵。

设 Z 是一个 $n \times n$ 矩阵，其行和列由 G 中的顶点编索引。如果 (u, v) 是一条边，我们设置 $Z[u, v] = z_{uv}$ 为一个不定元，否则 $Z[u, v] = 0$. 考虑以下过程。我们从域 $\mathbb{F}_p$ 中为每个变量 z_{uv} 指定独立均匀随机值。令 K 为根据对 Z 中变量的赋值所得到的 $n \times n$ 矩阵（这与第 6 节中 K 的定义一致）。

现在，对于每个顶点 v，我们指定一个流向量 $\vec{v}$，使其满足以下等式：**1.** 回想一下，$V_{\text{out}}[s] = \{v_0, \ldots\ldots, v_d\}$。对于每个顶点 v_i，我们要求其流向量满足

$$\vec{v}_i = \left(\sum_{u \in V_{\text{in}}(v_i)} \vec{u} \cdot K[u, v_i] \right) + \vec{u}_i. \tag{22}$$

2. 对于每个顶点 $v \notin V_{\text{out}}[s]$，我们要求其流向量满足

$$\vec{v} = \sum_{u \in V_{\text{in}}(v)} \vec{u} \cdot K[u, v]. \tag{23}$$

这些流向量是否存在并不明显，但我们将在下文证明它们确实存在（以较高概率，对 z_{uv} 变量的随机赋值而言）。令 H_s 表示 $(d+1) \times n$ 矩阵，其列由 G 中的顶点索引，其中对每个索引 $0 \le i \le d$，与 v_i 对应的列为 $\vec{v}_i$，其余列均为零向量。令 F 表示 $(d+1) \times n$ 矩阵，其列由 G 中的顶点索引，其中与顶点 v 对应的列为相应的流向量 $\vec{v}$。

因此，方程 (22) 和 (23) 可由矩阵方程表示

$$F = FK + H_s. \tag{24}$$

我们将证明满足上述条件的流向量 $\vec{v}$ 确实存在，方法是证明我们可以求解方程 (24) 中的 F。为此，我们使用以下引理。

▶ **引理 20.** 我们有 $\det(I-K) \neq 0$，其概率至少为 $1 - 1/n^3$。

证明。 的每个条目 Z 都是次数至多为一且常数项为零的多项式。由于输入图没有自环， Z 的对角线条目全为零。

因此，$\det(I - Z)$ 是次数至多为 n 且常数项为 1 的多项式（这意味着该多项式非零）。所以，根据施瓦茨-齐佩尔引理（命题 9），$\det(I - K)$ 非零的概率至少为 $1 - n/p \geq 1/n^4$（取 $p \geq n^5$），如所述。◀

从现在起，假设 $\det(I - K) \neq 0$（根据引理 20，这种情况以高概率发生）。在此情况下，流向量定义良好，并且根据方程 (24)，它们作为列出现于

$$F = H_s(I - K)^{-1} = \frac{H_s\left(\operatorname{adj}(I - K)\right)}{\det(I - K)}. \tag{25}$$

▶ **引理 21.** 对于 G 中的任意顶点 t，以至少 $1 - 1/n^3$ 的概率，我们有

$$\operatorname{rank} F[*, V_{\text{in}}[t]] \leq \begin{cases} \nu(s,t) + 1 & \textit{if } (s,t) \textit{ is an edge} \\ \nu(s,t) & \textit{otherwise.} \end{cases}$$

证明。 缩写 $\nu = \nu(s,\ t)$。从概念上讲，该证明通过论证 t 的所有入邻点所分配的流向量，都是图 G 中最小 $(s,\ t)$ 顶点割中的节点所分配的流向量的线性组合来实现，其中边 $(s,\ t)$ 已被移除。

令 G' 为删除边 $(s,\ t)$ 后得到的图 G（如果 $(s,\ t)$ 不是一条边，则 $G' = G$）。令 C' 为 G' 的一个最小 $(s,\ t)$ 顶点割。这意味着 C' 是一个大小最小的节点集合（其中 $s,\ t \notin C'$），删除 C' 后，G' 中将不存在从 s 到 t 的路径。从 G' 中移除 C' 中的节点后，令 S 为从 s 可达的顶点集合，令 T 为能够到达 t 的顶点集合。

如果 $(s,\ t)$ 是一条边，则令 $C = C' \cup \{s,\ t\}$。否则，令 $C = C'$。

根据门格尔定理，若 $(s,\ t)$ 不是 G 中的一条边，则 $|C| = \nu$；否则 $|C| = \nu + 1$。这是因为，除了边 $(s,\ t)$ 之外，我们还能找到 $|C'|$ 条从 s 到 t 的内部顶点不相交路径，因此 $\nu(s,\ t) = |C'| + 1 = |C| - 1$。

令 V' 为属于 T 或与 T 中某个顶点有边相连的顶点集合。然后令 $K' = K[V', V']$ 和 $F' = F[*, V']$。现在定义矩阵 $H' = F' - F'K'.$。根据定义，H' 的列以 V' 中的顶点为索引。

▷ 断言 22. 对所有 $v \in V' \setminus C$，都有 $H'[*, v] = \vec{0}$。

证明。 取 $v \in V' \setminus C$。

注意，$v \neq s$。事实上，若 (s,t) 是一条边，则 $s \in C$，因此由于 $v \notin C$，必然有 $v \neq s$。若 (s,t) 不是一条边，则根据顶点割的定义，s 不可能与 T 有边相连，而且 s 不属于 T，这意味着 $s \notin V'$，从而再次必然有 $v \neq s$。

我们首先处理 $v \neq t$ 的情形。

情形 1： $v \neq t$

在这种情况下，根据顶点割的定义，$v \in T$。因此，v 的入邻点都是 V' 的成员。根据本证明开头的讨论，$v \neq s$。

我们声称 $v \notin V_{\text{out}}(s)$。确实，反设 $v \in V_{\text{out}}(s)$。根据情形假设，$v \neq t$。根据定义，$v \notin C$ 且 v 有一条边连接到 T。这意味着存在一条从 s 到 T中某个顶点的长度为二的路径，且不经过 C中的任何顶点。这与(s,t)-顶点割的定义矛盾。因此，必然有 $v \notin V_{\text{out}}(s)$，如所声称。

因此，在这种情况下，我们已经证明，对于 $v \in V' \setminus C$，有 $V_{\text{in}}(v) \subseteq V'$ 且 $v \notin V_{\text{out}}(s)$。由等式 (23) 以及 F' 和 K' 的定义可得

$$(F'K')[*, v] = \sum_{u \in V_{\text{in}}(v) \cap V'} \vec{u} \cdot K'[u, v] = \sum_{u \in V_{\text{in}}(v)} \vec{u} \cdot K[u, v] = F[*, v] = F'[*, v] \tag{26}$$

这就证明了

$$H'[*,v] = F'[*,v] - (F'K')[*,v] = \vec{0}$$

正如所需。剩下需要处理的是 $v \in V' \setminus C$ 具有 $v = t$ 的情况。

情况 2： $v = t$

在这种情况下，根据 C 的定义， s 与 t 之间一定没有边。

因此 $t \notin V_{\text{out}}(s)$。显然，根据定义，我们有 $V_{\text{in}}(t) \subseteq V'$。所以，方程 (26) 中的计算同样适用于 $v = t$，这意味着 $H'[*,t] = \vec{0}$。

因此，对于所有 $v \in V' \setminus C$，我们都有 $H'[*,v] = \vec{0}$，正如所需。 ◁

根据断言 22， H' 至多有 $|C|$ 个非零列。因此，秩 $H' \le |C|$。

与引理 20 的证明类似的推理表明，矩阵 $I - K'$ 以至少 $1 - 1/n^3$ 的概率可逆。因此，我们可以在方程 $H' = F' - F'K'$ 中求解 F'，从而获得。 t

$$F' = H'(I - K')^{-1}.$$

由于秩 $H' \le |C|$，上述方程也意味着秩 $F' \le |C|$。

根据定义， $V_{\text{in}}[t] \subseteq V'$，因此 $F[*, V_{\text{in}}[t]]$ 是 F'的一个子矩阵。由此可得

秩 $F[*, V_{\text{in}}[t]] \le |C|$.

由于 $|C| = \nu + 1$ if $(s,\ t)$ 是一条边，否则为 $|C| = \nu$，所需结果得证。◀

▶ **引理 23.** 对于 G 中的任意顶点 t，我们以至少 $1 - 2/n^3$ 的概率有

$$\operatorname{rank} F[*, V_{\text{in}}[t]] \ge \begin{cases} \nu(s,t) + 1 & \text{if } (s,t) \text{ is an edge} \\ \nu(s,t) & \text{otherwise.} \end{cases}$$

证明。 简记 $\nu = \nu(s,t)$。直观而言，我们的证明将论证，从 s 到 t 的内部顶点不相交路径的存在，会使 $V_{\text{in}}[t]$ 中的某些节点被赋予线性无关的流向量（以高概率成立），进而推出所需的下界。根据定义， G 包含 ν条从 P_1、.....、 P_ν到 s的内部顶点不相交路径 t。

考虑对矩阵 Z 的 z_{uv} 变量作如下赋值。对于每个 (u,v)，如果 (u,v) 是其中一条 P_i 路径上的边，则令 $z_{uv} = 1$。否则，令 $z_{uv} = 0$。对于每个顶点 v，令 $\vec{v}$ 表示在此赋值下与 v 对应的流向量。为了证明所需的秩下界，首先确定其中一些流向量的值会有所帮助。

令 $\vec{u}_0$ 表示最初在顶点 s 处泵出的单位向量（如方程（22）所示）。

取任意一条 P_i 路径。

首先，假设 P_i 是一条长度至少为二的路径。在这种情况下，令 a_i 表示 P_i中的第二个顶点，令 b_i 表示 P_i 中的倒数第二个顶点（注意，若 P_i 的长度恰好为二，则 $a_i = b_i$）。根据定义， $a_i \in V_{\text{out}}$（s）。令 $\vec{u}_i$ 表示最初在节点 a_i 处泵出的单位向量（如方程（22）所示）。那么，根据我们对 z_{uv}变量的赋值选择，由方程（22）可得

$$\vec{a_i} = \vec{u}_0 + \vec{u}_i.$$

接着，对 P_i上从 a_i 到 b_i 的路径上的顶点反复应用等式 (23)，我们发现

$$\vec{b_i} = \vec{u}_0 + \vec{u}_i \tag{27}$$

也同样成立。上述方程刻画了长度至少为二的 P_i 路径的倒数第二个顶点所对应的流向量。接下来考虑 P_i 的长度为一的情形。若 P_i 的长度为一，则它仅由一条从 s 到 t 的边构成。于是根据公式（22），我们有

$$\vec{s} = \vec{u}_0. \tag{28}$$

现在我们可以证明，对于这些变量的这一特定赋值，$V_{\text{in}}[t]$ 中的节点对应的流向量合在一起达到所需的秩下界。

▷ 命题 24. 对于如下赋值：若 $z_{uv} = 1$（u, v）是某条路径上的一条边，则 P_i；否则 $z_{uv} = 0$，$V_{\text{in}}[t]$ 中的流向量的秩至少为 $\nu + 1$；若（s, t）是一条边，则至少为 $\nu + 1$，否则至少为 ν。

证明。 我们根据 $(s,\ t)$ 是否为一条边分情况讨论。

情形 1：$(s,\ t)$ 不是一条边

假设（s, t）不是一条边。那么每条 P_i 路径的长度至少为 2。等式（27）表明，V_{in}（t）中的流向量包括 $\vec{u}_0 + \vec{u}_1$，$\vec{u}_0 + \vec{u}_2$，.....，$\vec{u}_0 + \vec{u}_\nu$。由于这些 $\vec{u}_i$ 是互不相同的单位向量，因此这些流向量的秩至少为 ν，符合要求。

情况 2：$(s,\ t)$ 是一条边

反之，假设 $(s,\ t)$ 是一条边。那么，在我们的最大顶点不相交路径集合中，其中一条路径必定是从 s 到 t 的直接边。不失一般性地，设 P_ν 是这条长度为一的路径（因此，对于所有 $1 \le i \le \nu - 1$，P_i 是一条长度为二的路径）。

在这种情况下，$t \in V_{\text{out}}$（s）。令 $\vec{u}_\nu$ 表示在 $\vec{t}$ 处泵出的单位向量。

将方程（27）和（28）代入公式（22），可得

$$\vec{t} = \vec{s} + \left(\sum_{i=1}^{\nu-1} \vec{b}_i\right) + \vec{u}_\nu = \nu \cdot \vec{u}_0 + \left(\sum_{i=1}^{\nu-1} \vec{u}_i\right) + \vec{u}_\nu.$$

结合上述方程与方程（27）和（28），我们看到，分配给 $V_{\text{in}}[t]$ 中节点的流向量（包括 $\vec{b}_1$、...、$\vec{b}_{\nu-1}$、$\vec{s}$ 和 $\vec{t}$）张成了包含互不相同的单位向量 $\vec{u}_0$、$\vec{u}_1$、.....、$\vec{u}_\nu$ 的空间，因此其秩至少为 $\nu + 1$，如所需。◁

为方便起见，在本证明的其余部分中，令 $\tilde{\nu}$ 表示：当 $(s,\ t)$ 是一条边时取 $\tilde{\nu} = \nu + 1$，否则取 $\tilde{\nu} = \nu$。

根据命题 24，$F[*, V_{\text{in}}(t)] = \tilde{\nu}$ 的秩为 $F[*, V_{\text{in}}$。因此，$F[*, V_{\text{in}}(t)]$ 包含一个满秩 $\tilde{\nu} \times \tilde{\nu}$ 子矩阵。

令 F' 为这样一个子矩阵。

现在，在为 z_{uv} 变量赋值之前，$\text{adj}(I - Z)$ 的每个元素都是次数至多为 n 的多项式。因此，根据方程 (25)，$\det F'$ 等于某个次数至多为 $n\tilde{\nu}$ 的多项式 P 除以多项式 $(\det(I - Z))^{\tilde{\nu}}$。注意，$\det(I - Z)$ 是非零多项式，因为其常数项等于 1。

根据 F' 的定义，我们知道，在按照命题 24 的陈述为变量赋值后，有 $\det(F') \neq 0$。由于作为一个多项式

$$\det(F') = P / \left(\det(I - Z)\right)^{\tilde{\nu}}, \tag{29}$$

必然有 P 是一个非零多项式（因为如果 P 是零多项式，那么无论如何赋值，$\det(F')$ 的值都会是零）。

根据引理 20，以至少 $1 - 1/n^3$ 的概率，行列式 $\det(I - Z) \neq 0$ 将在对 z_{uv} 变量进行均匀随机赋值时非零。假设这一事件发生，根据施瓦茨-齐佩尔引理（命题 9），对这些变量进行均匀随机求值，还将以至少 $P \neq 0$ 的概率具有

$$1 - (2\tilde{\nu}n)/p \geq 1 - 1/n^3$$

通过设置 $p \geq 2n^5$。

因此，根据并集界，从 $\mathbb{F}_p$ 中为 z_{uv} 个变量进行均匀随机赋值，可使 P 和 $\det(I - Z)$ 同时非零的概率至少为 $1 - 2/n^3$。

如果这种情况发生，根据方程 (29)，我们有 $\det(F') \neq 0$。这意味着 $F[*, V_{\text{in}}(t)]$ 的某个特定子矩阵将以所声称的概率满秩，从而证明了所需结果。◀

确立这些引理后，我们便可以立即证明本节的主要结果。

命题15的证明。固定一对不同的顶点 (s, t)。

将 H_s 的定义代入方程 (25)，可得

$$F[*, V_{\text{in}}[t]] = (I - K)^{-1}[V_{\text{out}}[s], V_{\text{in}}[t]].$$

根据引理 21 和 23 的并集界，我们可以得出

$$\operatorname{rank}\,(I - K)^{-1}[V_{\text{out}}[s], V_{\text{in}}[t]] = \operatorname{rank} F[*, V_{\text{in}}[t]] = \begin{cases} \nu(s,t) + 1 & \text{if } (s,t) \text{ is an edge} \\ \nu(s,t) & \text{otherwise} \end{cases}$$

其概率至少为 $1 - 3/n^3$。 ◀