arXiv:2603.06402v1 [cs.DS] 6 Mar 2026

横截秩、共形性与枚举

MartinSchirneck ✉卡尔斯
鲁厄理工学院

摘要

超图的横截秩是其极小击中集的最大大小。对于一个有 n 个顶点、 m 条边的超图和整数 k，判断其横截秩是否至少为 k，有一种自 20 世纪 70 年代以来就已知的算法，运行时间为 $O(m^{k+1}n)$。这基本上匹配了 Araújo、Bougeret、Campos 和 Sau 在 [Algorithmica 2023] 以及 Dublois、Lampis 和 Paschos 在 [TCS 2022] 中给出的 $(m+n)^{\Omega(k)}$ETH 下界。实践中遇到的许多超图，其边数远多于顶点数，$m \gg n$。这引出了一个问题：能否通过增加对 n 的依赖，来换取降低运行时间对 m 的依赖?

我们的第一个结果是一种算法，可在 $O(\Delta^{k-2}mn^{k-1})$ 时间内识别横截秩至少为 k 的超图，其中 $\Delta \leqslant m$ 是最大度。我们的主要技术贡献是一种“前瞻”方法，能够找到高阶扩展，即在给定集合的基础上至少增加两个顶点的极小击中集。我们证明，该方法还可用于枚举横截秩为 k^* 的超图的所有极小击中集，且延迟为 $O(\Delta^{k^*-1}mn^2)$。

接下来，我们探讨能否将计算横截秩的运行时间进一步降至 $\text{poly}(m)\cdot n^{k+O(1)}$。事实证明，这与组合算法和枚举领域的若干重大突破等价 。其中，这种改进能够实现，当且仅当可以在 $\text{poly}(m)\cdot n^{k+O(1)}$ 时间内识别 k-共形超图；并且，当且仅当能够以增量延迟枚举均匀超图的极大超团/独立集。

2012 年 ACM 学科分类 计算数学 → 超图；计算数学 → 枚举；计算理论 → 图算法分析；计算理论 → W分层

关键词与短语 共形度、枚举、击中集、超团、超图、独立集、横截秩。

资助 *Martin Schirneck*：作者获得德国研究基金会（DFG）资助，资助编号为 556899211（“枚举算法的设计、分析与工程”）。

1 引言

图算法领域有大量研究探讨如何判断给定输入是否属于某个特定的图类。我们已经有算法可以识别平面图、树宽至多为 67 的图，或独立数至少为 42 的图。然而，对于超图，已知方法要少得多，且大多数文献都聚焦于均匀超图，即所有超边都包含相同数量的顶点 [1, 21, 29], ，或者等价地说，每一列都包含相同数量的 1 的二元矩阵 [17]。我们的目标是推进非一致超图类识别算法这一领域的发展。具体而言，我们关注横截秩较大的超图，横截秩是任意包含关系意义下极小的击中集的最大大小。[1] 虽然我们的研究对象是非一致超图，但分析借鉴了均匀超图领域的思想。此外，我们还揭示了这一问题与枚举所有极小击中集问题之间的深刻联系。

横截秩最初用于衡量“局部改进”方法求解击中集问题时的最坏情况性能。它在参数化算法、核化、形式概念分析和计算生物学等领域都有应用 [31, 42], 。

[1] 横截集是击中集的另一种说法。横截超图是所有极小击中集构成的集合，而横截秩是该超图中超边大小的最大值。

2 横截秩、共形性与枚举

将横截秩称为如下判定问题：给定一个有 n个顶点、m条超边的超图以及一个非负整数 k，判断是否存在一个大小至少为 k 的极小击中集。顾名思义，该问题也称为最大极小击中集 [19]。它推广了经典优化问题的许多其他“最大—最小”变体，包括最大极小顶点覆盖 [10], 、最大极小反馈顶点集 [23], 、最大极小阻塞集（即需要击中图的所有最大独立集 [2]），以及上支配（最大极小支配集）及其变体上全支配数 [3]。

最大最小变体通常比其对应的最小化问题更难。关于上支配问题计算复杂性的已知下界可直接推广到横截秩。Cheston、Fricke、Hedetniemi 和 Jacobs [15]的结果因此表明，该问题是 NP 完全的。Bazgan、Brankovic、Casel、Fernau、Jansen、Klein、Lampis、Liedloff、Monnot 和 Paschos [3] 证明，该问题在以 k 为参数时是 W[1]-困难的，而且即使将横截秩近似到因子 $n^{1-\varepsilon}$以内也是 NP-hard。他们还证明，除非指数时间假设（ETH）不成立，否则不存在运行时间为 $(m+n)^{o(\sqrt{k})}$ 的横截秩算法。最近，Araújo、Bougeret、Campos 和 Sau [2] 以及 Dublois、Lampis 和 Paschos [24] 分别独立开展的两项工作将这一结果改进为 $(m+n)^{o(k)}$。

目前，识别具有较大极小击中集的超图时，效率最高的算法源自 Berge 和 Duchet 在 20 世纪 70 年代的一项结果 [6] 。这里，斯佩纳超图是指任意两条边都不存在包含关系的超图。[2]

▶**引理1** (Berge 和 Duchet [6]; 参见 [5]）。设 H 是一个斯佩纳超图，且 $k \geqslant 3$ 是一个整数。当且仅当存在k 条互不相同的边 E_1、E_2、.....、$E_k \in H$，使得 H 中的每条边 $E \in$ 都含有一个顶点 $x \in E$，该顶点至多出现在这些边 E_i中的一条中，超图 H 的横截秩至少为 k 。此外，若 D 表示所有此类 x构成的集合，则每个极小击中集 $T \subseteq D$ 的大小至少为 k。

对于击中集而言，考虑完整超图$\mathcal{H}$ 还是只考虑其包含关系下极小的边，结果并无区别。找出这些边所需的运行时间为 $O(m^2 n)$，所得子超图是斯佩纳超图，因此可以应用引理1。给定$\{E_i\}_i$ 条边组成的集合，其中边来自$\mathcal{H}$，检查所得集合 D 是否为$\mathcal{H}$ 的击中集，可在 $O(mn)$ 的时间内完成。因此，遍历$\mathcal{H}$ 的所有 k-子超图，总运行时间为 $O(m^{k+1} n)$。我们在附录A中说明，采用更谨慎的预处理可以将运行时间降至 $O(m^{k+1} + m^k n)$，且其效率始终不低于 $O(m^{k+1} n)$。要为较大的横截秩构造一个实际见证，即一个大小至少为 k 的极小击中集，几乎不需要额外工作。一旦找到合适的集合 D，便可在 $O(mn)$ 的时间内计算出任意一个极小解$T \subseteq D$ （另见附录A）。

该算法在指数项中与 Araújo 等人 [2] 以及 Dublois、Lampis 和 Paschos [24]给出的下界相差至多一个常数因子。然而，就 m 而言，其复杂度很高，因此对于边数众多的超图来说，代价高得难以承受；而这正是许多实际应用中的典型情形 [4, 8, 38, 40]。请注意，在非一致超图中（与图不同） m 不受 n的多项式上界约束。ETH 排除了将运行时间中 m 和 n 的指数同时降低至 $o(k)$的可能性。尽管如此，两者之间的细粒度权衡仍完全

[2] Sperner 超图有时也称为 简单，例如在 [5], 中，但这一术语在 [30]所述的文献中并未得到一致使用。

尚未得到探索。例如，$m^{k-10} \cdot \text{poly}(n)$ 和 $O(m \cdot n^{3k/2})$ 这样的运行时间都与我们目前的认知相符，但在实践中会导致截然不同的表现。

我们研究是否可以降低对边数 m 的依赖，即使这可能会增加关于顶点数 n 的复杂度。我们的第一个结果是：对于边数非常多的超图，或每个顶点的度都很小的实例，可以使用更快的算法。令 $\Delta \leqslant m$ 表示最大度。

▶**定理 2**。横截秩至少为 k 的超图可以在时间 $O(\Delta^{k-2} m n^{k-1})$ 内被识别。对于 $k < 2$，运行时间为 $O(mn)$。

比较引理 1 和定理 2 可知，对 m 的依赖从 m^{k+1} 降至 $\Delta^{k-2} m$，但这会使关于顶点数的运行时间增加至 n^{k-1}。当 $m = \Omega((\Delta n)^{\frac{k-2}{k}})$ 时，这种权衡是有利的。

考虑极小击中集的扩展问题（Extension MinHS）：我们需要检查给定的顶点集合 X 是否包含于某个极小击中集中。Boros 、 Gurvich 和 Hammer [11] 证明了该问题是 NP 完全的，而 Bläsius、Friedrich、Lischeid、Meeks 和 Schirneck [9] 证明，当以部分解的基数 $|X|$ 为参数时，该问题是 W[3]-完全的。我们研究这样一种变体：判断是否存在一个包含 X 且至少多使用两个 顶点的极小击中集。我们称之为高阶扩展。本文第一部分的核心贡献是一种算法，它允许我们在寻找 X 的高阶扩展时进行前瞻 。其运行时间与原始扩展问题的现有最佳方法相同。这是建立定理 2 的关键要素。在此过程中，我们确定了高阶变体是扩展问题的核心难题。这立即表明它同样是 NP 完全和 W[3]-完全的，并且意味着若干细粒度下界。（详见第 2 节。）

前瞻思想也可应用于极小击中集的枚举 。寻找最小基数 击中集在经典意义上是 NP 难的 [35], ，但简单的贪心算法可以得到一个极小击中集。介于这两个极端之间的是枚举任务，也就是计算并列出超图的所有 极小击中集。极小解可能有指数多个，因此显然不可能存在多项式算法。我们希望实现输出多项式 时间，即运行时间随输入规模和解的数量呈多项式增长。枚举领域的这一 重大未解问题是，所谓的横截超图问题 是否存在输出多项式算法 [12, 20, 37, 39]。

虽然凭借我们目前的技术似乎还无法回答这个问题，但已有大量研究致力于找出此问题可高效求解的超图类别，参见例如 [25, 26, 27, 36]。对于横截秩 k^* 有界的超图，还能做到比单纯的输出多项式方法更好。延迟 是相邻两次输出之间的时间，包括首次输出前的预处理时间，以及最后一次输出后直至算法终止的时间。Bläsius 等人 [9] 利用 Extension MinHS 问题设计了一种延迟为 $O(m^{k^*+1} n^2)$ 的算法。该算法的关键性质是，它不 需要预先知道 k^* ，[3] 只有分析依赖于 k^*。Araújo 等人 [2]随后发现，可以将界收紧至 $O(\Delta^{k^*} mn^2)$；当度较小时，这种算法效率更高，但当 $\Delta = \Omega(m)$ 时，其运行时间达到最坏情况。我们在高阶扩展方面的改进，使我们得以突破一般超图的 $\Omega(m^{k^*+1}) \cdot \text{poly}(n)$ 延迟障碍。

[3] 如果已知 k^* 且它确实是一个常数，那么只需穷举所有大小不超过 k^* 的集合，就能得到一个平凡的多项式（因此也是输出多项式）算法。

4 横截秩、共形性与枚举

▶**定理 3.** 横截秩为 k^* 的超图的极小击中集可以用延迟 $O(\Delta^{k^*-1}mn^2)$ 枚举。该算法使用的空间为 $O(mn)$。

本文的第二部分探讨这样一个问题：是否能够将计算横截秩的运行时间中对 m 的依赖进一步降低，突破定理 2 的限制。理想情况下，我们希望运行时间关于 m 为多项式，其指数与 k 无关。当然，我们不能指望同时达到复杂度 $n^{o(k)}$。因此，我们研究运行时间为 poly(m) $\cdot\, n^{k+O(1)}$ 的方法所带来的影响。结果表明，这样的改进等价于 共形超图组合算法领域的若干其他突破，以及极大超团的枚举。

考虑一个顶点集合 S，存在一条超边 $E \in \mathcal{H}$ 满足 $S \subseteq E$。显然， S 的每个 k 元子集也都包含在 $\mathcal{H}$ 的一条超边中。如果反过来也成立，即 S 的所有 k 元子集都包含在某条超边中，就足以保证存在一条包含 S 中所有顶点的超边，那么这个超图就称为 k-共形的。$\mathcal{H}$ 的共形度 是使 $\mathcal{H}$ 成为 k-共形超图的最小 k。Berge 和 Duchet [6]早已观察到，横截秩较大的超图与共形度较大的超图之间存在密切联系（见引理 16）。因此，识别 k-共形超图是 coNP 完全的，而在以 k $[3, 16]$ 为参数时是 coW[1]-困难的。k-均匀超图 $\mathcal{G}$ 中的超团是一个集合 S，其中每个 S 的 k 元子集都等于 $\mathcal{G}$的一条超边。

我们将横截秩与共形度之间的联系扩展到枚举问题。若组合问题的解可以按增量延迟列出，我们就称其具有增量延迟：第i次延迟，即第 i个输出与第(i+1)个输出之间的时间，受输入规模和 i的多项式所界。本文始终用变量 i 表示目前已见到的解的数量。一般而言，增量延迟的概念介于输出多项式时间与多项式延迟之间；但对于击中集，已知任何输出多项式时间算法都可以转换为具有增量延迟 [7]的算法。这包括秩 r（最大超边大小）有常数上界的超图。然而，与横截秩的情况类似，文献中尚缺乏对 poly(m, n, i)-延迟的细致讨论。相比延迟为 $O(i^{r+2} \cdot n)$的方法，延迟为 $i^3 \cdot n^{r+O(1)}$ 的方法更受青睐；后者是当前最佳水平 [25]。我们证明了共形度的计算、有秩上界的超图中极小击中集的枚举，以及极大超团的枚举之间存在定量等价关系。当然，关于超团的一切结论，只需交换边与非边的角色，也同样适用于独立集。

▶**定理4**。 以下陈述等价。

- 横截秩至少为 k 的超图可以在poly(m)$\cdot\ n^{k+O(1)}$时间内识别。 k-共形超图可以在poly(m)$\cdot\ n^{k+O(1)}$ 时间内识别。秩为 r 的超图的极小击中集可以用增量延迟 poly(i)$\cdot\ n^{r+O(1)}$ 枚举。秩为 r 的均匀超图的极大超团/独立集可以用增量延迟poly(i)$\cdot\ n^{r+O(1)}$枚举。

一方面，这种等价性表明，在均匀超图中更快的枚举算法也能帮助解决非一致输入下的判定问题。另一方面，它为证明（条件性）延迟下界开辟了新的途径；这在枚举复杂度领域一向是个极难解决的问题 [12]。如今，要排除在均匀超图中快速列举极小击中集/极大超团的算法，归结为证明横截秩问题的下界。我们指出，定理4前两项中的超图

不需要具有有界秩。事实上，如果 r 是常数，Araújo 等人 [2] 展示了如何以快得多的速度计算横截秩，所需时间为 $2^{rk} \cdot \mathrm{poly}(m, n)$。

传统上，研究极小击中集的枚举时采用的是另一类判定问题：给定两个超图$\mathcal{H}$ 和$\mathcal{G}$，验证$\mathcal{G}$ 是否恰好由$\mathcal{H}$的极小击中集构成。已知当且仅当极小击中集能够以增量延迟 [7]枚举时，该问题才能在多项式时间内（以$\mathcal{H}$ 和$\mathcal{G}$的合并输入规模计）求解。对于秩为r 的超图$\mathcal{H}$，我们进一步强化了这一等价关系。

▶**定理 5.** 设 $\mathcal{H}$ 是一个秩为 r的超图。以下陈述等价。可以以增量延迟 $\mathcal{H}$ 的极小击中集，延迟为 $\mathrm{poly}(i) \cdot n^{r+O(1)}$。对于任意超图 $\mathcal{G}$，判定其是否包含且仅包含 $\mathcal{H}$ 的极小击中集，可以在 $\mathrm{poly}(|\mathcal{G}|) \cdot n^{r+O(1)}$ 的时间内完成。

定理4的核心在于证明：识别具有较大横截秩的超图，与快速枚举秩经过适当选择的均匀超图的极大超团是等价的。对于简单图（2-均匀超图），已有一些非平凡的团枚举算法。Tsukiyama、Ide、Ariyoshi 和 Shirakawa [41]，以及后来 Johnson、Papadimitriou 和 Yannakakis [34]，展示了如何以延迟 $O(|V| \cdot |E|)$枚举图 $G = (V, E)$的极大团。我们将这一结果纳入框架，从而得到一种识别共形超图（即 2-共形超图）的算法。它优于此前基于引理1的、运行时间为 $O(m^4 n)$的过程。

▶**定理 6。** 共形超图可以在 $O(mn^3)$ 时间内被识别。

概要。 接下来，我们将对主要贡献进行技术概述。我们在第3节中固定基本记号。随后，第4节介绍前瞻及其应用，第5节证明判定问题与枚举问题之间的等价性。

2 概述与讨论

我们的第一项贡献是一个前瞻算法，用于枚举具有有界横截秩的超图中的极小击中集。相同的方法还可以用于更快地计算横截秩。然后，我们概述以下等价性背后的技术思想：识别具有较大横截秩的超图、具有较大共形度的超图，以及枚举超团和击中集之间的等价性。我们还简要讨论进一步改进的可能性，以及未来工作的开放问题。

2.1 高阶扩展的前瞻

我们的枚举算法改进了 Bläsius 等人 [9] 提出的算法。它构建一棵搜索树，并通过一个子程序对其进行剪枝；该子程序判断，对于一对不相交的顶点集合 X，$Y \subseteq V$，能否将 X 扩展为一个不包含 Y 中任何元素的极小击中集。对于任意 Y，这个扩展子程序的运行时间为 $O(\Delta^{|X|} mn)$。他们关于 $O(\Delta^{k^*} mn^2)$ 延迟界的主要论证是：对于横截秩为 k^* 的超图，该子程序只会在集合 X 满足 $|X| \leqslant k^*$ 时被调用。尽管 k^* 对算法而言事先未知，这一点仍然成立。文章 [9] 还给出了若干条件下界，表明该扩展算法几乎已达到最优。对于任意常数 $|X| \geqslant 2$ 和 $\varepsilon > 0$，若存在一种运行时间为 $O(m^{|X|-\varepsilon}) \cdot \mathrm{poly}(n)$ 的方法，就会违背强

指数时间假设（SETH）。而更强得多（可以说）的非确定性强指数时间假设（NSETH），则构成一道屏障，使运行时间甚至无法降至 $O(m^{|X|+1-\varepsilon}) \cdot \mathrm{poly}(n)$（精确表述见推论 15）。因此，从这一方面来看，改进的希望不大。

相反，我们着手解决树搜索的瓶颈。在根节点中，两个集合 X 与 Y均为空。搜索在每个节点检查(X, Y)是否可扩展。若是，算法便针对是否将 $V \backslash (X \cup Y)$中的下一个顶点加入部分解 X 或“禁止”顶点集合 Y这一决策进行分支。最坏情况出现在到达某个集合X 、且其基数恰好等于横截秩 k^*之时。若我们事先知道 k^*，便可在 $|X| = k^* - 1$处的瓶颈之前一层停止搜索。我们将迅速找到所有扩展得到的极小击中集，因为它们至多只再增加一个顶点。X 的所有其他超集对枚举而言都无关紧要，因为它们已经过大，不可能成为极小解。问题在于，若不计算横截秩 k^*，我们便无法检验 X是否具有正确的基数。

为了解决这一瓶颈，我们转而在搜索树的每一层中加入“前瞻”。主要困难在于判断 X 是否仍存在高阶扩展，即是否存在一个 $T \supseteq X$ 满足 $|T \backslash X| \geqslant 2$的极小击中集。随后，搜索必须扩展当前节点（$X$，$Y$），以找出其余的解。我们证明了高阶扩展的结构刻画，这意味着，判断是否存在这样的 T，至少与不带大小限制的扩展问题一样困难。尽管如此，我们还是给出了一个在相同 $O(\Delta^{|X|} mn)$时间内运行的算法。换言之，我们的前瞻是免费的。这正是我们能够实现延迟改进的原因，也有助于解决原始问题：识别横截秩至少为 k的超图。

人们可能会希望将这种方法以某种方式迭代下去。与其只考虑另外两个顶点，不如检查至多再包含 ℓ 个元素的极小扩展，并期待枚举延迟为 $\Delta^{k^*-\ell+1} mn^{\ell+O(1)}$，同时在计算横截秩方面也能获得类似的改进。如果可行，这种方法将绕过扩展问题的下界；仅通过增强树搜索，就能在对 Δ和 n的依赖之间实现平滑的权衡。然而，迭代式前瞻需要一种互补算法，用于判断 X 是否包含在基数至少为 $|X| + \ell$的极小击中集中。对于 $\ell > 2$而言，这似乎比扩展问题困难得多。

2.2 横截秩与共形度

可以设想，一种完全不同的方法能将识别横截秩至少为 k 的超图的运行时间一路降至 $\mathrm{poly}(m) \cdot n^{k+O(1)}$。我们证明，这等价于若干以超团代替击中集的其他新结果。考虑顶点集为 V 的超图$\mathcal{H}$，以及一个至少有 k个顶点的极小击中集 $S \subseteq V$。极小性意味着，对每个满足 $|S'| = k - 1$的子集 $S' \subsetneq S$，存在超边 $E \in \mathcal{H}$ 使得 $S' \cap E = \emptyset$。因此，S' 包含在补 $E = V \backslash E$中。由于S 击中$\mathcal{H}$的每一条超边，该集合本身不包含在任何补边中。这表明超图 $\mathcal{H} = \{V \backslash E \mid E \in \mathcal{H}\}$ 不是$(k-1)$-共形的。事实上，引理1可以改述为：横截秩至少为 k 等价于$\mathcal{H}$ 的共形度至少为 k （见引理 16）。

—

我们借鉴均匀超图分析中的思想，将这种等价关系推广到若干枚举问题。给定一个超边大小任意的超图 $\mathcal{H}$，其k-节 $[\mathcal{H}]_k$ 的构造方式是：将所有同时出现在 $\mathcal{H}$ 的某条超边中的 k 顶点集合，作为 $[\mathcal{H}]_k$的一条超边。显然，k-节是 k-均匀的。（在上面的例子中，每个集合 S' 都是$\mathcal{H}$的 $(k-1)$-节中的一条超边。）我们刻画了.....的共形度

$\mathcal{H}$ 与 $[\mathcal{H}]_k$的极大超团有关。关键观察是，计算共形度时，我们至多只需知道 $|\mathcal{H}|$ 个这样的超团，即使团的数量远超于此也是如此。这正是增量延迟枚举发挥作用之处。这类算法能够对连续两次输出之间的最长时间给出非平凡的界，因此也能界定我们得到第 $|\mathcal{H}|$个解所需的时间。

请回忆，图的团与补图的独立集之间存在一一对应关系。而独立集又可通过取集合补集与顶点覆盖（击中集）相对应。相同的论证可将均匀超图中极小击中集的任何延迟改进，转移到极大超团的枚举上。现在，为了完成这一系列归约，我们将证明，更快地计算横截秩的算法确实能够加速均匀超图中极小击中集的枚举。我们的结果甚至适用于具有有界秩这一更大的超图类；我们不需要所有超边都具有相同的大小。考虑这样一种算法：它已经找到了$\mathcal{H}$的若干个极小击中集，但仍需判断是否还有更多解有待发现。到目前为止找到的极小击中集自身构成了一个超图$\mathcal{G}$ 。基于 Eiter 和 Gottlob [25], 的工作，我们展示了如何利用有关$\mathcal{G}$的横截秩的信息，判断$\mathcal{G}$ 是否已经包含全部解。

与细粒度等价通常一样，它们可以被视为寻找更优算法的额外动力，也可以被视为这项工作不可能实现的条件性证据。目前尚无已知算法能在 $\text{poly}(m) \cdot n^{k+O(1)}$ 时间内计算横截秩，等价地说，即共形度。我们将其留作一个开放问题：能否找到这样的方法，或者在合理的复杂性假设下证明其不存在。可以说，一个更为务实的研究目标是找到一个运行时间为 $m^{k+1-\varepsilon} \cdot n^{o(k)}$ 的算法。

3 预备知识

超图、秩与补。 超图 是一个有限集合 V 以及一个子集族$\mathcal{H} \subseteq 2^V$。 V的元素是顶点，$\mathcal{H}$ 包含（超）边。通常我们将超图与其边集等同起来。可以假定本文中的所有超图都具有相同的顶点集。我们用 $n = |V|$ 表示其大小。超图的边数可以不同；若不致产生歧义，我们用 $m = |\mathcal{H}|$ 表示。顶点 $v \in V$ 的度 为 $\deg(v) = |\{E \in \mathcal{H} \mid v \in E\}|$，即包含 v 的边数。H 的最大度 为 $\Delta = \max_{v \in V} \deg(v)$。

超边最大基数为 秩 $\text{秩}(\mathcal{H}) = \max_{E \in \mathcal{H}} |E|$ 。如果秩由某个非负整数 r 限定，我们就称 $\mathcal{H}$ 为 r- 超图。如果所有边的大小都相同，均为 r，则称其为 r- 均匀 。2- 均匀超图就是通常的（无向、简单）图。若超图 $\mathcal{H}$ 中任意两条边都不互相包含，则称 H 为斯佩纳图。注意，均匀超图（无论秩是多少）始终都是斯佩纳图。

我们需要区分超图（V,$\mathcal{H}$）的两种不同的补 的概念。我们将$\overline{\mathcal{H}} = \{V \setminus E \mid E \in \mathcal{H}\}$ 定义为包含补边的超图。在$\mathcal{H}$ 为 r-均匀时这一特定情形下，我们令$\mathcal{H}^{\mathsf{C}} = \{E \in \binom{V}{r} \mid E \notin \mathcal{H}\}$ 由所有这样的 r-顶点子集组成：它们不是$\mathcal{H}$的边。

击中集与横截秩。 设 $(V,\mathcal{H})$ 是一个（秩不一定有界的）超图， $X \subseteq V$ 是一个顶点集合。子超图 $\mathcal{H}(X)$ 由所有包含 X 中某个元素的边$E \in \mathcal{H}$ 组成。其余的边属于 $\mathcal{H}^{V \setminus X} = \mathcal{H} \setminus \mathcal{H}(X)$。这一记法表明， $E \in \mathcal{H}^{V \setminus X}$ 等价于 $E \subseteq V \setminus X$。

我们称一组顶点 $T \subseteq V$ 为击中集，或横截集，如果它与每条超边都有交集，也就是说，对所有 $E \in \mathcal{H}$， $E \cap T \neq \emptyset$ 都成立；等价地，$\mathcal{H}(T) = \mathcal{H}$。该

8 横截秩、共形性与枚举

击中集若其任何真子集都不是击中集，则称为（按包含关系）极小的。等价地，击中集是极小的，当且仅当对每个 $t \in T$，都存在一条私有边 $E \in \mathcal{H}$，满足 $E \cap T = \{t\}$。如果某个 $t \in T$ 没有私有边，那么集合 T 就是冗余的。

$\mathcal{H}$ 的极小击中集构成一个定义在相同顶点集 V 上的超图，即横截超图Tr($\mathcal{H}$)。无论 $\mathcal{H}$ 是否具有斯佩纳性质，它都是斯佩纳的。对于两个不相交集合 X，$Y \subseteq V$，我们用 Tr($\mathcal{H}$)$[X, Y]$ 表示 Tr($\mathcal{H}$) 中满足 $X \subseteq T \subseteq V \backslash Y$ 的所有极小击中集所构成的子超图。

H 的横截秩是 rank(Tr($\mathcal{H}$))，也就是极小横截的最大基数。我们考虑相应的搜索问题。

横截秩

输入： 一个超图$(V,\ \mathcal{H})$ 和一个非负整数 k。

输出： 要么给出 $T \in \mathrm{Tr}(\mathcal{H})$，其中 $|T| \geqslant k$；要么正确输出$\mathrm{rank}(\mathrm{Tr}(\mathcal{H})) < k$.

对于 $k \leqslant 2$，问题是显然的。超图 $\mathcal{H}$ 的横截秩至少为 1，当且仅当 $\emptyset \notin \mathcal{H}$。其横截秩至少为 2，当且仅当它包含两条关于集合包含关系不可比较的边。

超团与共形性。设 (V,$\mathcal{H}$) 是一个 r-均匀超图。$\mathcal{H}$ 中的超团是一个至少含有 r 个顶点的集合 $C \subseteq V$，其中每个满足 $|S| = r$ 的子集 $S \subseteq C$ 都是超边，$S \in \mathcal{H}$。若一个超团的所有真超集都不再是超团，则称其为（按包含关系而言）极大 超团。所有极大超团构成的集合记为 $\mathcal{C}$($\mathcal{H}$)。反过来，如果 C 不包含 $\mathcal{H}$ 的任何一条边，那么它就是一个独立集。

现在设 (V,$\mathcal{H}$) 是一个任意超图，k 是一个正整数。若对于每个集合 $S \subseteq V$，以下两个条件等价，则称$\mathcal{H}$ 是 k-共形的 。

1. 存在一个超边 $E \in \mathcal{H}$ 使得 $S \subseteq E$。
2. 对于每个子集 $S' \subseteq S$ 满足 $|S'| \leqslant k$ 存在一个超边 $E' \in \mathcal{H}$ 满足 $S' \subseteq E'$。

如果$\mathcal{H}$ 是 k-共形的，那么对于所有 $k' \geqslant k$，它也是 k'-共形的。$\mathcal{H}$ 的共形度 是使$\mathcal{H}$ 为 k-共形的最小 k 。

共形度

输入： 一个超图 (V,$\mathcal{H}$) 以及一个非负整数 k。

输出： 正确输出 $\mathcal{H}$ 是 k-共形的，或者给出一个集合 $S \subseteq V$ 使得 $\binom{S}{\leqslant k}$ 中的每个子集都包含在 $\mathcal{H}$的某条超边中，但 $S \nsubseteq E$ 对于所有 $E \in \mathcal{H}$。

参数化复杂性与（强）指数时间假设。我们假定读者对参数化复杂性已有一定了解，此处仅回顾最重要的概念。更详尽的阐述请读者参阅教科书 [18, 22, 28]。考虑序对 $(I, k) \in \{0,1\}^* \times \mathbb{N}_0$，其中 I是输入实例（的适当编码），而非负整数 k 是参数。参数化问题 Π 是$\{0,1\}^* \times \mathbb{N}_0$ 的一个子集。若存在可计算函数 f，使得包含关系 $(I, k) \in \Pi$ 可在时间 $f(k) \cdot \mathrm{poly}(|I|)$ 内判定（称为 FPT 时间），则该问题是固定参数可处理 （固定参数可处理）。参数化问题 Π与 P 之间的参数化归约是一个可在 FPT 时间内计算的函数，它将 Π 的实例 (I, k) 映射为 P 的等价实例 (I', k')，且存在可计算函数 g，满足 $k' \leqslant g(k)$。

这些归约形成了一个称为 W-层级的复杂性类体系。加权电路可满足性问题是指：给定一个布尔电路 C 和一个非负整数 k，判断 C 是否存在汉明重量为k的满足赋值。该问题的参数为 k。C 中的大门是指扇入严格大于 2 的门。C 的纬度 是其任意输入输出路径上大门数量的最大值，这些路径均经过

C。对于任意正整数 t，类 W[t] 由所有可归约到加权电路可满足性问题的参数化问题构成，其中电路深度为常数且纬度为 t。人们猜想 W 层级中的所有包含关系 FPT $\subseteq$ W[1] $\subseteq$ W[2] $\subseteq$ W[3] $\subseteq \ldots$ 都是严格的。类 coW[t] 包含类 W[t]中问题的补。W 层级之外还有类 para-NP，它包含所有可在 FPT 时间内由非确定性算法求解的参数化问题。当且仅当 P = NP 时，FPT = para-NP，参见 [28]。如果一个参数化问题在常数参数值下已经是 NP 完全的，那么它就是 para-NP 完全的。类 para-coNP 是使用余非确定性算法和 coNP 类得到的对应概念。

比 W 层级是严格的更强的困难性假设是指数时间假设 (ETH) [32, 33]。它指出，对于含有 n 个变量、处于合取范式且子句长度为 3 的布尔公式，3-SAT 问题无法在 $2^{o(n)}$ 时间内求解。已知 ETH 蕴含 W[1] $\neq$ FPT [14]。反过来，ETH 可由强指数时间假设 (SETH) [33] 推出；SETH 声称，对于每个常数 $\varepsilon > 0$，都存在一个子句长度 ℓ，使得 ℓ-SAT 无法在$O(2^{(1-\varepsilon)n})$ 时间内求解。最后，非确定性强指数时间假设 (NSETH) 是 SETH 和 coNP $\neq$ NP [13] 的一种常见推广。它表示如下猜想：对于每个 $\varepsilon > 0$，都存在一个 ℓ，使得不存在不使用随机性的 *co*-非确定性算法能在 $O(2^{(1-\varepsilon)n})$ 时间内判定 ℓ-SAT。请注意，尽管 NSETH 刻画了当前最坏情况 SAT 求解的前沿，但它是一个非常强的假设，因此并非完全可信。它的价值在于，无论证明还是否定 NSETH，都会对复杂性理论产生有趣的影响 [13]。

4 横截秩的前瞻

本节介绍一种算法：给定一个非空超图$\mathcal{H} \neq \emptyset$，其横截秩（未知）为 k^*，最大度为 Δ，该算法以 $O(mn)$空间枚举$\mathcal{H}$ 的极小击中集，延迟为 $O(\Delta^{k^*-1}mn^2)$。首先回顾 Bläsius 等人 [9], 提出的方法，包括 Araújo 等人 [2]的分析。

4.1 Bläsius 等人的枚举算法

该算法 [9] 通过反复调用扩展判定预言机来构造并剪枝搜索树；扩展判定预言机是一个子程序，用于判断顶点集合 X 是否包含于某个极小击中集 $T \subseteq X$中。我们称这样的集合 T 为 X 的极小扩展 。该算法的正确性取决于以下引理。

▶ **引理7** ([9, 11])。 设(V,H) 是一个超图， $X \subseteq V$是一个顶点集合。当且仅当存在一个边族$\{E_x\}_{x\in X} \subseteq H$，使得集合 X具有极小扩展
1. 对于每个顶点 $x \in X$，我们有 $E_x \cap X = \{x\}$；**2.** 对于每条超边 $E \in \mathcal{H}$ 包含于 $\bigcup_{x\in X} E_x$，我们有 $E \cap X \neq \emptyset$。

回想一下，极小击中集的每个顶点 x 都需要一条仅被 x击中的私有边。引理7的第一个条件使 E_x 成为 x 关于部分解 X的一条候选私有边。当 X 从上下文中显而易见时，我们用 $\mathcal{H}_x = \{E \in \mathcal{H} \mid E \cap X = \{x\}\}$ 表示 x的所有候选私有边所构成的子超图。第二个条件进而确保 X的不同元素所对应的候选 E_{x_1}， E_{x_2}，.....互不干扰。

除了部分解 X 之外，我们还需要纳入一个集合 Y，其中的顶点已被排除在搜索之外。采用记号 Tr($\mathcal{H}$)$[X,Y]$ 表示所有包含 X 但避开 Y 的 $\mathcal{H}$ 的极小击中集，我们得到如下结果。

▶ **推论 8.** 设 (V,H) 为一个超图，且 X、$Y \subseteq V$ 为两个不相交的顶点集合。存在极小击中集 $T \in \mathrm{Tr}(H)[X,Y]$，当且仅当存在边族 $\{E_x\}_{x\in X} \subseteq H$，使得

1. 对于每个顶点 $x \in X$，我们有 $(E_x \backslash Y) \cap X = \{x\}$；
2. 对于每条边 $E \in \mathcal{H}$，若 $(E \backslash Y) \cap X = \emptyset$，则 $(E \backslash Y) \nsubseteq \bigcup_{x\in X}(E_x \backslash Y)$。

对于子超图 $\mathcal{G} \subseteq \mathcal{H}$，我们用 $\mathcal{G}^* = \{E \backslash Y \mid E \in \mathcal{G}\}$ 表示移除 Y 后相应的边集合。例如，$\mathcal{H}_x^* = \{E \backslash Y \mid E \in \mathcal{H},\ E \cap X = \{x\}\}$ 是 x 的约化候选私有边，而 $(\mathcal{H}^{V\backslash X})^* = \{E \backslash Y \mid E \in \mathcal{H},\ E \cap X = \emptyset\}$ 是尚未被 X 击中的约化边。我们有时也用 E^* 表示边 $E \backslash Y$ 本身。

推论 8 给出了如下解法，用于求解给定不相交集合对 X 和 Y 的扩展问题。若其中一个超图 $\mathcal{H}_x^*$ 为空，则 X 是冗余的，因此没有极小扩展。否则，若 $(\mathcal{H}^{V\backslash X})^* = \emptyset$（即，$\mathcal{H}^*$ 的所有边都被 X 击中），则 X 是非约化边所构成的原始超图 $\mathcal{H}$ 的极小击中集。如果这些合理性检查仍未能确定该实例的结果，则测试笛卡尔积中所有边组合 $(E_{x_1}^*, E_{x_2}^*, \ldots, E_{x_{|X|}}^*) \in \prod_{x\in X} \mathcal{H}_x^*$，检查它们的并集 $\bigcup_{x\in X} E_x^*$ 是否完整包含 $(\mathcal{H}^{V\backslash X})^*$ 中的某条边。当且仅当存在一种组合不满足这一条件时，X 才能扩展为一个不使用 Y 的 $\mathcal{H}$ 极小击中集。

在顶点集 V 上固定一个任意全序 $\leqslant$。该算法构造一棵树，其中每个节点都标有一个不相交对 (X,Y)，使得 $X \cup Y$ 关于序 $\leqslant$ 是 V 的前缀。根节点标为 $(\emptyset,\emptyset)$。在任意节点 (X,Y) 处，检验 X 是否在不含 Y 的情况下可扩展。若 X 没有极小扩展，则该分支中的搜索停止并回溯。若检验判定 X 本身已是极小击中集，则输出该解，并同样对该分支进行剪枝。否则，X 确实在不含 Y 的情况下可扩展，搜索便针对将下一个顶点 $v_{\min} = \min_{\leqslant} V \backslash (X \cup Y)$ 纳入何处这一决策进行分支，方法是创建两个子节点 $(X \cup \{v_{\min}\},\ Y)$ 和 $(X, Y \cup \{v_{\min}\})$。

该算法无需知道横截秩。根据设计，X 的基数从不超过 k^*。任何具有 $|X| = k^*$ 的集合要么本身就是一个（最大）极小击中集，要么已经过大而不可扩展。最坏情况出现在如下情形：一个含有 k^*-1 个顶点的集合被验证为可扩展，但下一个顶点 $v_{\min}$ 使得 $X \cup \{v_{\min}\}$ 既非冗余也非极小。于是必须检查全部 $\prod_{x\in X\cup\{v_{\min}\}} |\mathrm{H}_x^*| = O(\Delta^{k^*})$ 种可能组合，仅仅为了验证总存在某条超边 $E^* \in (\mathrm{H}^{V\setminus(X\cup\{v_{\min}\})})^*$，满足 $E^* \subseteq \bigcup_{x\in X\cup\{v_{\min}\}} E_x^*$。每次检查耗时 $O(|(\mathrm{H}^{V\setminus(X\cup\{v_{\min}\})})^*| \cdot |V|) = O(mn)$。$O(\Delta^{k^*} mn^2)$ 延迟源于两次输出之间隔着 $O(n)$ 个节点。

4.2 高阶扩展

我们的改进源于通过前瞻来避免这一瓶颈，并检查 X 是否存在至少多出 2 个顶点的极小扩展。考虑不相交集合 X 和 Y，我们已知存在一个极小击中集 $T \in \mathrm{Tr}(\mathcal{H})[X,Y]$。显然，可以检查 $T = X$ 本身是否已经是极小的；我们称之为 *0* 阶扩展。找出所有 *1*-扩展也很容易，它们是恰好比 X 多一个元素的极小解。然而，极小扩展 T 可能比 $|X|+1$ 大得多。这些就是高阶扩展。事实证明，判定它们是否存在是扩展问题的核心难点，见第 4.4 小节。下面的引理对于理解高阶扩展的计算复杂性，以及在不牺牲运行时间的情况下实现前瞻思想，都至关重要。我们先讨论 $Y = \emptyset$ 的情形。

▶ **引理 9.** 设(V, H)是一个超图，$X \subseteq V$是一个顶点集合。定义$S = \bigcap H^V \backslash X$。若一个极小击中集$T \in \mathrm{Tr}(H)$满足$T \supseteq X$，则$|T \backslash X| > 1$当且仅当$T \subseteq V \backslash S$。

证明。 集合$S = \bigcap \mathcal{H}^{V\backslash X}$包含出现在所有尚未被$X$击中的边中的那些顶点。对于任意这样的$s \in S$，集合$X \cup \{s\}$是$\mathcal{H}$的一个（不一定是极小的）击中集。因此，任何真超集$T \supsetneq X \cup \{s\}$都不是一个极小击中集。如果$X$存在高阶扩展$T$，那么我们必有$s \notin T$。

反过来，每个满足$X \subseteq T \subseteq V \backslash S$的$T \in \mathrm{Tr}(\mathcal{H})$都比$X$至少多两个元素。它不能只用一个顶点击中$\mathcal{H}^{V\backslash X}$中的所有边，因为这些边都属于$S$。◀

对于包含非空集合Y的一般情形，容易验证：T是X的一个避开Y的极小扩展，当且仅当它是在约化边超图$\mathcal{H}^* = \{E \backslash Y \mid E \in \mathcal{H}\}$中的一个（大小任意的）极小扩展。

▶ **推论 10.** 设(V,H)是一个超图，$X, Y \subseteq V$是两个不相交集合。令$S = \bigcap (H^V \backslash X)^*$为与$X$不相交的约化边的交集。一个极小击中集$T \in \mathrm{Tr}(H)$满足$X \subseteq T \subseteq V|Y$，当且仅当$|T|X| > 1$满足$(Y \cup S)$。

现在我们实现前瞻。我们不再只是检查X在不包含Y的情况下是否可扩展，而是设计一种算法，输出X中避开Y的所有 0 阶和 1 阶扩展，然后判断$\mathrm{Tr}(\mathcal{H})[X, Y]$中是否还存在高阶扩展。此外，如果存在这类较大的扩展，算法会计算一个集合$Y^+ \supseteq Y$，并保证所有剩余扩展也避开Y^+。这样便无需评估满足$Y \subseteq Z \subseteq Y^+$的节点$(X, Z)$，从而可以缩短搜索过程。我们还加入了若干合理性检查，以便在合适的输入上提前剪枝。算法的完整概述见算法 1。

直到第 9 行，过程处理的是$X = \emptyset$的情况。这部分由对所有主要步骤的简化组成；这些步骤也用于非空的X。我们将推迟介绍这部分，直到更详细地讨论一般情况。现在假设$X \neq \emptyset$。在第 10 至 15 行，预处理用于存储子超图$\mathcal{H}^*_x$（对所有$x \in X$）以及$(\mathcal{H}^V \backslash x)^*$的数据结构。前者存储候选私有边，而后者包含满足以下条件的那些$E \backslash Y$：$E \in \mathcal{H}$与X不相交。满足$|E \cap X| \geqslant 2$的边E可以丢弃。如果第 13 行触发了对$E \subseteq Y$的检查，那么$E \backslash Y = \emptyset$将包含在$(\mathcal{H}^V \backslash x)^*$中。这证明$X$无法在不含$Y$的情况下扩展。（参见推论 8 的第二项。）因此可以立即中止执行。填充好数据结构后，过程会在第 16 行验证每个x确实都有一条候选私有边，否则X就是冗余的。如果$(\mathcal{H}^V \backslash x)^* = \emptyset$，则不存在未击中边。此前对私有边的检查确保此时$X$必定是一个极小的击中集。因此，$X$将作为其自身的 0 阶扩展输出。

从第 20 行到第 22 行的以下部分就是前瞻。它判断在X中添加一个顶点是否足以使其成为极小解。根据引理 9，如果存在这样的顶点s，它必须包含在$S = \bigcap (\mathrm{H}^V \backslash x)^*$中。然而，这个条件并不充分，击中集$X \cup \{s\}$可能不是极小的。当且仅当每个$x \in X$都相对于$X$有一条（候选）私有边，且该边不包含$s$时，该集合才确实是极小的。相同的边也因此是相对于$X \cup \{s\}$的私有边。第 21 至 22 行检验了这一点。某个顶点$x \in X$有一条不包含s的私有边，这等价于$s \notin \bigcap \mathrm{H}^*_x$。因此，$s \notin U = \bigcup_{x \in X} \bigcap \mathrm{H}^*_x$会同时验证所有$x$。原则上，新元素$s$也需要一条私有边。不过，$(\mathrm{H}^V \backslash x)^*$中的任意一条边都可以满足此目的，因为它们都包含$s$，且与$X$不相交。第 17 行已经检查过$(\mathrm{H}^V \backslash x)^*$非空。总之，对于每个$s \in S \backslash U$，$X \cup \{s\}$都被认证为一个 1- 扩展。

剩下的几行将判断X是否存在高阶扩展，并从第 23 行的合理性检查开始。考虑只剩下一条尚未

算法 1 计算超图 $(V,\mathcal{H})$ 中避免 Y 的 X 0 阶和 1 阶扩展。

输入： 不相交集合 $X=\{x_1,\ldots,^{x_{|X|}}\}$ 和 $Y\subseteq V$。
输出： 所有极小扩展 $T\in\mathrm{Tr}(\mathcal{H})[X,Y]$，满足 $|T\backslash X|\leqslant 1$。
返回： 如果存在一个 $T'\in\mathrm{Tr}(\mathcal{H})[X,Y]$，满足 $|T\backslash X|>1$，则继续
其中 $Y\subseteq Y^+\subseteq V\backslash X$ 满足所有此类 T' 均属于 $\mathrm{Tr}(\mathcal{H})[X,Y^+]$。
如果不存在这样的 T'，则停止。

过程 `extendable`(X,Y):

`/* preprocessing empty sets` X `*/`
如果 $X=\emptyset$，**则**
  初始化 $\mathcal{H}^*=\emptyset$;
  对于每个 $E\in\mathcal{H}$，**执行**
    如果 $E\subseteq Y$，**则返回** 停止;
    如果 $E\cap X=\emptyset$，**则**将 $E\backslash Y$ 添加到 $\mathcal{H}^*$;
  $S\leftarrow\bigcap H^*$;
  对于每个 $s\in S$，**输出**$\{s\}$;
  如果 $|\mathcal{H}^*|=1$，**则返回** 停止;
  返回 继续执行 $Y\cup S$;

`/* preprocessing non-empty sets` X `*/`
初始化超图 $(\mathcal{H}^V\backslash X)^*=\emptyset$;
对 $x\in X$ **执行**初始化超图$\mathcal{H}_x^*=\emptyset$;
对 $E\in\mathcal{H}$ **执行**
  若 $E\subseteq Y$，**则返回** 停止;
  若 $E\cap X=\emptyset$，**则**添加 $E\backslash Y$至 $(\mathcal{H}^V\backslash X)^*$;
  若 $E\cap X=\{x\}$，**则**添加 $E\backslash Y$至$\mathcal{H}_x^*$;
如果 $\exists x\in X:\ \mathcal{H}_x^*=\emptyset$ **则返回** HALT;
若$(\mathcal{H}^V\backslash X)^*=\emptyset$ **则**
  输出 X;
  返回 停止;

`/* look ahead */`
$S\leftarrow\bigcap(\mathcal{H}^V\backslash X)^*$;
$U\leftarrow\bigcup_{x\in X}\bigcap H_x^*$;
对于每个 $s\in S\backslash U$，**输出** $X\cup\{s\}$;

`/* higher-order extensions */`
如果 $|(\mathcal{H}^V\backslash X)^*|=1$，**则返回** HALT;
对于每个$(E^*_{x_1}\ldots,E^*_{x_{|X|}})\in\mathcal{H}^*{}_{x_1}\times\cdots\times H^*{}_{x_{|X|}}$ **do**
  $W\leftarrow S\cup\bigcup_{i=1}^{|X|}E^*_{x_i}$;
  如果 $\forall E^*\in(\mathcal{H}^V\backslash X)^*:\ \ E^*\not\subseteq W$**则返回** 继续执行 $Y\cup S\cup U$;
返回停止;

算法2 有序顶点集（V, $\leqslant$）上超图 H 的横截超图问题的递归算法。初始调用为 `enumerate`$(\emptyset, \emptyset)$。输入：不相交集合 X、 $Y \subseteq V$。输出：极小击中集 $T \in$ Tr(H) $[X, Y]$。1 过程 `enumerate`(X, Y)：2 isExtendable $\leftarrow$ `extendable`(X, Y)；3 若 isExtendable $==$ 终止则返回；4 若 isExtendable $==$ 继续使用 Y^+ 则

5 $v_{\min} \leftarrow \min_{\leqslant} V \backslash (X \cup Y^+)$;6 `enumerate`$(X \cup \{v_{\min}\}\ Y^+)$;

7 `enumerate`$(X, Y^+ \cup \{v_{\min}\})$;

尚未被 X 击中，即 $|(\mathcal{H}^V \backslash x)^*| = 1$。因此我们有 $(\mathcal{H}^V \backslash x)^* = \{S\}$ 且 Tr(H)$[X, Y \cup S]$ 为空。换言之， X 没有高阶扩展。在其余部分，假设 $(\mathcal{H}^V \backslash x)^*$ 中有不止一条需要击中的超边。根据推论 10， X 具有避开 Y 的高阶扩展，当且仅当 X 可以在不使用 $Y \cup S$的任何顶点的情况下被扩展。推论 8 进而指出，这等价于存在边的组合 $E_{x_1} \in \mathcal{H}_{x_1}, \ldots, E_{x_{|X|}} \in \mathcal{H}^{x_{|X|}}$ 使得

$$E \backslash (Y \cup S) \nsubseteq \bigcup_{i=1}^{|X|} E_{x_i} \backslash (Y \cup S) \tag{1}$$

对每个 $E \in \mathcal{H}^V \backslash x$ 都成立。我们避免显式计算集合 $E \backslash (Y \cup S)$ 或 $E_{x_i} \backslash (Y \cup S)$，而是重用在预处理中已准备好的 $E^* = E \backslash Y \in (\mathcal{H}^V \backslash x)^*$ 和 $E^*_{x_i} = E_{x_i} \backslash Y \in \mathcal{H}^*_{x_i}$。注意，(1) 等价于 $E \backslash Y \nsubseteq S \cup \bigcup_{i=1}^{|X|} E_{x_i} \backslash (Y \cup S)$，因而也等价于 $E^* \nsubseteq S \cup \bigcup_{i=1}^{|X|} E^*_{x_i}$。如果 X 确实存在避开 Y 的高阶扩展，那么先前的计算表明，它们不会使用来自 S 的顶点（否则它们将是 1- 扩展），也不会使用来自 U 的顶点（任何满足 $u \in U$ 的集合 $X \cup \{u\}$ 都是冗余的）。

现在回到 $X = \emptyset$ 的情形（第 1 至 9 行）。同样，会构造 $(\mathcal{H}^V \backslash x)^*$ 的数据结构，算法会计算集合 S。超图 $\mathcal{H}^*_x$ 显然不需要。在这个特殊情形下，包含关系 $s \in S$ 对于 $X \cup \{s\} = \{s\}$ 是 1-扩展这一点既是必要条件，也是充分条件。根据我们的假设，$\mathcal{H} \neq \emptyset$ 非空，因此唯一的真子集 $\emptyset \subsetneq \{s\}$ 不是击中集。

暴力循环（第 24 至 26 行）主导了运行时间。因此，我们的过程与 Bläsius 等人 [9]（另请参阅 Araújo 等人 [2]的分析）的方法具有相同的渐近时间 $O(\Delta^{|X|} mn)$。由于所有数据结构 $\mathcal{H}^*_x$ 和 $(\mathcal{H}^V \backslash x)^*$ 中都存放着互不相交的 $\mathcal{H}^*$的子超图，总空间需求为 $O(mn)$。计算所有 1-扩展X 以及延续集 Y^+ 不会增加运行时间。

4.3 延迟界

算法2将算法 1中的 `extendable` 过程与树搜索相结合。我们现在证明其正确性以及延迟改进。证明的技术部分包含在以下引理中。

14 横截秩、共形性与枚举

▶ **引理 11.** 设 X、$Y \subseteq V$ 为两个不相交的顶点集合。
1. 调用 $\texttt{enumerate}(X, Y)$ 会恰好枚举出 $\mathrm{Tr}(\mathcal{H})[X, Y]$中的极小击中集。**2.** 在算法*2*的执行过程中，每次调用 $\texttt{enumerate}(X, Y)$ 都满足 $|X| \leqslant k^* - 1$。

证明。 我们在假设第二个结论的一个较弱版本确实成立的前提下证明第一个结论，即 $|X| \leqslant k^*$。这样便可以对 $k^* - |X|$进行归纳证明。如果X 恰好等于横截秩，那么 X 要么是一个极小击中集（它自身的 0 阶扩展），要么没有极小扩展。因此，第 2 行对 $\texttt{extendable}(X, Y)$的调用会返回 HALT，并且可能会先输出解 X。过程 $\texttt{enumerate}(X, Y)$ 随后不再执行任何操作便返回。

现在假设 $|X| < k^*$。如果 $\mathsf{Tr}(\mathcal{H})[X, Y]$ 中的所有解都是 X 的 0 阶扩展或 1-扩展，那么如前所述，结论成立，因为 $\texttt{extendable}(X, Y)$ 会输出这些解，然后返回 halt。否则，它会对某个集合 $Y \subseteq Y^+ \subseteq V \backslash X$ 返回 continue 和 Y^+，其中 $\mathsf{Tr}(\mathcal{H})[X, Y]$ 中的所有高阶扩展实际上都属于 $\mathsf{Tr}(\mathcal{H})[X, Y^+]$。特别地，至少存在一个这样的扩展，因此 $V \backslash (X \cup Y^+)$ 非空。考虑下一个顶点 $v_{\min} = \mathsf{min}_{\leqslant} V \backslash (X \cup Y^+)$。根据归纳法，每个包含 $v_{\min}$ 的解都会由第 6 行对 $\texttt{enumerate}(X \cup \{v_{\min}\}\ Y^+)$ 的调用枚举出来，其余解则由第 7 行的 $\texttt{enumerate}(X, Y^+ \cup \{v_{\min}\})$ 找出。

第二个子句说明 $|X| \leqslant k^* - 1$ 始终成立。为导出矛盾，假设在某个时刻，针对一个至少含有 k^*个元素的 X 集合调用了 $\texttt{enumerate}(X, Y)$。由于$\mathcal{H}$ 非空，k^* 为正，因此（X，Y）$\neq$（$\emptyset$，$\emptyset$）不是搜索树的根节点。设（A，B）是（X，Y）的父节点。特别地，在执行 $\texttt{enumerate}(A, B)$期间，对 $\texttt{extendable}(A, B)$ 的调用（第2行）返回了 continue，并带有 B^+，这意味着我们有 $B^+ = Y$。因此，在 $T \in \mathsf{Tr}(\mathcal{H})[A, Y]$中存在一个极小击中集，其顶点数至少比 A多两个。集合 A 和 X 要么相等，要么 X 恰好多出一个顶点，即 $v_{\min} = \mathsf{min}_{\leqslant} V \backslash$（$A \cup Y$）。因此，该极小扩展的基数必定为

$$|T| \geqslant |A| + 2 \geqslant |X| + 1 \geqslant k^* + 1.$$

这与 k^* 是任意极小击中的最大基数相矛盾。◀

现在，我们已准备好证明延迟界。

▶ **引理12。**设(V, H) 是一个非空超图，具有 n 个顶点、m 条边、最大度 Δ和横截秩 k^*。算法*2*以 H 为输入，每条$\mathrm{Tr}(H)$中的边恰好输出一次，延迟为 $O(\Delta^{k^*-1} mn^2)$。执行过程占用 $O(mn)$ 空间。

证明。算法2的初始调用为 $\texttt{extendable}(\emptyset, \emptyset)$。因此，根据引理 11 的第一条，$\mathrm{Tr}(\mathcal{H}) = \mathrm{Tr}(\mathcal{H})[\emptyset, \emptyset]$ 会被完整枚举。(mn) 空间需求来自扩展子程序，其使用这么多空间；而树搜索只需维护当前的数对(X, Y)。

关于延迟，注意搜索树中父节点与子节点之间总会有至少一个新节点被添加到 X 或 Y。因此，树的深度至多为 n。树按前序遍历，因此在首次输出之前、任意两次连续输出之间，或最后一次输出之后直至终止，执行的 $\texttt{extendable}(X, Y)$调用次数至多为 $O(n)$。

最坏情况下，下一个输出对应于一个含有 $|X| = k^* - 1$ 的集合 X。因此，中间的任意 $O(n)$ 次调用都需要 $O(\Delta^{k^*-1} mn)$ 时间，总延迟为 $O(\Delta^{k^*-1} mn^2)$。请注意，从最后一次输出到算法终止之间的时间也满足这一点，因为在最坏情况下，对于从源节点 $(\emptyset,\ \emptyset)$ 到最后一个叶节点路径上的每个节点，算法都需要验证另一个子节点确实不会导向任何解。◀

4.4 计算复杂性

上一节提出了两个计算问题。最主要的是，我们需要一种算法来判定 X 是否具有高阶扩展。我们将其称为“问题(a)”。另一个问题则稍微隐晦一些。考虑算法2第6行和第7行中对 `enumerate` 的递归调用。第二个参数 $Y^+ = Y \cup S \cup U$ 表示，如果 X 存在一个避开 Y 的高阶扩展，那么其中没有任何一个包含 $S \cup U$ 的顶点。同理，如果我们能够找出一个集合 $X^+ = X \cup W$，使得只要 X 可扩展为一个极小击中集，W 就包含在每个这样的扩展中，那么树搜索就会更快（“问题 (b)”）。现在我们确定这两个问题的计算复杂性。结果表明，它们分别是 NP 和 W[3] 类的完全问题，以及 coNP 和 coW[3] 类的完全问题。我们的所有归约都使用极小击中集的参数化扩张问题。

Extension MinHS
输入： 一个超图 $(V, \mathcal{H})$ 以及一个由 $X \subseteq V$ 个顶点组成的集合。**参数：** 基数 $|X|$。**判定：** 是否 X 包含在$\mathcal{H}$的一个极小击中集中?

请注意，Extension MinHS 不要求所求的、扩张 X 的极小击中集具有特定大小。该问题是 NP 完全且 W[3]-完全的，参见 [9, 11]。

▶**定理 13**。 判定一个集合 X 是否具有高阶极小扩张（问题*(a)*）是NP-完全的；以基数 $|X|$ 为参数时，该问题是W[3]-完全的。

证明。 如前所述，我们双向归约于 Extension MinHS。其复杂性不受输入中包含一个额外集合的影响： S 所求极小解必须避开该集合。我们可以简单地移除 S，即将其从输入超图的每条超边中去掉，参见 [9]。我们证明，在保持参数的多项式时间归约下，高阶问题等价于（无限制的）扩展问题。

由于 $S = \bigcap \mathcal{H}^{V \setminus X}$ 可由 X 与 $\mathcal{H}$ 在多项式时间内计算得到，引理 9 已经给出了证明其属于 NP 与 W[3] 的归约。关于困难性，令 $(V,\mathcal{H})$ 为一个超图，并令 $X \subseteq V$ 为那个集合，我们需要判定是否存在一个没有任何基数限制的极小扩展。我们检查 X 本身是否为极小击中集，或能否通过添加来自 V的单个顶点而变为极小击中集。若是，则该归约为高阶变体产生一个平凡的 yes-实例。否则，我们判定原实例 $(V, \mathcal{H}\ X)$ 是否存在高阶扩展。◀

▶ **定理14.** 对于不相交集合 X, W，判定 X的每个极小扩展是否都包含 W （问题 *(b)*）是coNP-完全的；以$|X|$为参数时，该问题是coW[3]-完全的。改以 $|W|$ 为参数时，该问题是para-coNP-完全的。即使承诺 X 确实存在一个极小扩展，复杂度也不受影响。

证明。 我们通过将“包含-W”扩展问题的肯定实例归约为 否定实例Extension MinHS，反之亦然，来证明该定理。一个方向的归约可直接由以下观察得到：当且仅当 X 包含 W 时，X 不能在不包含 W的情况下扩展。正如定理 13 的证明中所述，为Extension MinHS 的输入添加一个扩展需要避开的集合，不会产生任何影响。

为了证明困难性，设 $(V,\mathcal{H}\ X)$ 是 Extension MinHS 的一个实例。如果 X 是一个极小击中集，我们就输出一个平凡的“否”实例作为问题 (b) 的实例。如果这不能解决该实例，我们便按如下方式继续。回顾一下，$\mathcal{H}(X)$ 是由与 X 相交的边构成的子超图。由于 X 不是一个击中集，$\mathcal{H}^{V \setminus X}$ 中至少存在一条未被击中的边。现在令

$W=\{w\}$ 为一个单元素集合，其唯一元素 w 不属于 V。我们通过如下方式定义一个顶点集为 $V\cup W$ 的超图 $\mathcal{G}$:

$$\mathcal{G}=\mathcal{H}(X)\cup\{E\cup W\mid E\in\mathcal{H}^{V\setminus X}\}.$$

每个 $x\in X$ 在$\mathcal{H}(X)\subseteq\mathcal{G}$中都有一条私有超边，因此 $X\cup W$ 是$\mathcal{G}$的极小击中集。特别地， X 在$\mathcal{G}$ 中可扩展这一承诺成立。集合 X 在原始超图$\mathcal{H}$ 中不可扩展，当且仅当$\mathcal{G}$ 中的每一个极小扩张都必须包含 W。该问题对于 $|W|=1$ 已经困难这一事实给出了参数化 co-NP 完全性。◀

注意到这两个困难性归约可在时间 $O(mn^2)$内计算，不会增加边数 m 或参数 $|X|$，并且至多添加一个新顶点。因此，我们从原始扩展问题 [9,定理 4]继承了所有细粒度下界。特别地，我们对高阶扩展的 $O(\Delta^{|X|}mn)$运行时间是紧的；若引入随机化，则至多可能相差一个因子 m 。

▶**推论 15.** 不存在任何算法能够求解问题 *(a)* 或 *(b)* 中的任何一个

1. 在时间 $f(|X|)\cdot(m+n)^{o(|X|)}$ 对于任何可计算函数 f，除非 W[2] =固定参数可处理;**2.** 在时间 $m^{|X|-\varepsilon}\cdot\mathrm{poly}(n)$ 对于任何常数 $|X|\geqslant 2$ 和 $\varepsilon>0$，除非强指数时间假设不成立。

此外，任何确定性归约都无法在时间界$m^{|X|+1-o(1)}\cdot\mathrm{poly}(n)$下证明 *(a)* 或 *(b)* 的 *SETH* 困难性，除非非确定性强指数时间假设不成立。

4.5 计算横截秩

我们最初着手寻找一种更快计算横截秩 $k^*=$rank(Tr($\mathcal{H}$)) 的方法。我们证明，同样的前瞻思想在这里也同样有用。

▶**定理 2.** 横截秩至少为 k 的超图可在时间$O(\Delta^{k-2}mn^{k-1})$内被识别。对于 $k<2$，运行时间为 $O(mn)$。

证明。我们首先讨论参数为$k=0$ 或 $k=1$时横截秩问题的平凡情形。任何满足 $\emptyset\notin$ H 的非空超图都具有正的横截秩。更具体地说，如果 $\emptyset\in$ H 是一条超边，则 Tr(H) $=\emptyset$ 为空，横截秩未定义。如果 H $=\emptyset$，则 Tr(H) ={$\emptyset$}，横截秩为 0。否则， $k^*\geqslant 1$成立。注意，对于 $k=0$ 和 $k=1$，任意极小击中集都见证了 rank(Tr(H)) $\geqslant k$。计算它需要 $O(mn)$ 时间。

在下文中，我们假设 $k\geqslant 2$。因此，不等式 $k^*\geqslant k$ 等价于存在一个由 $k-2$ 个顶点组成的集合，该集合具有更高阶的 极小扩展。（当 $k=2$时，这也包括空集。）考虑一个基数为 $|X|=k-2$ 的集合 $X\subseteq V$，并定义 $S=\bigcap\mathcal{H}^{V\setminus X}$。根据引理 9， X 包含于一个大小至少为$|X|+2$ 的极小击中集当且仅当 Tr($\mathcal{H}$)$[X,S]$ 非空。使用此前已用于证明算法 1（后半部分）正确性的相同论证，可以在 $O(\Delta^{|X|}mn)=O(\Delta^{k-2}mn)$ 时间内判定 Tr($\mathcal{H}$)$[X,S]\neq\emptyset$ 。对所有由 $(k-2)$个顶点组成的集合应用此过程，即可在总计 $O(\Delta^{k-2}mn^{k-1})$ 时间内判定横截秩。

如果横截秩确实高于给定阈值，我们就按如下方式找到一个见证。令 $X=\{x_1,\ldots,x_{k-2}\}$ 为算法验证的第一个集合，该集合包含于一个足够大的击中集中。因此，暴力循环（从算法 1 的第 24 行开始）计算出了一组适当选取的超边 $E_{x_1}\in\mathcal{H}x_1,\ldots,E_{x_{k-2}}\in\mathcal{H}${v19>，使得对于每条边

$E \in \mathcal{H}^{V\setminus X}$，我们都有 $E \nsubseteq S \cup \bigcup_{i=1}^{k-2} E_{x_i}$。因此，该集合

$$M = X \cup V \setminus \left(S \cup \bigcup_{i=1}^{k-2} E_{x_i} \right)$$

是$\mathcal{H}$的一个（不一定是极小的）击中集。我们断言，任何 极小击中集 $T \subseteq M$都是 X 的扩展，且至少包含 $|X| + 2 = k$ 个元素。根据附录A，可以在时间 $O(mn)$内由$\mathcal{H}$ 和 M计算得到。为验证这一断言，首先注意到，任何这样的 T 都必须包含所有 X。如果缺少某个 x_i ，超边 E_{x_i} 就不会被击中。此外，由于 T 与 S不相交，根据引理 9，它必须比 X 至少多包含两个元素。◀

5 共形度与超团枚举

我们已经看到，计算横截秩与枚举极小击中集之间存在联系。现在我们将证明，计算共形度与枚举极大超团之间存在定量等价关系。

▶**定理4.** 以下陈述等价。

- 横截秩至少为 k 的超图可在$\mathsf{poly}(m) \cdot n^{k+O(1)}$时间内识别。 k-共形的超图可在$\mathsf{poly}(m) \cdot n^{k+O(1)}$时间内识别。秩为 r 的超图的极小击中集可在增量延迟为$\mathsf{poly}(i) \cdot n^{r+O(1)}$的条件下枚举。秩为 r 的均匀超图的极大超团/独立集可在增量延迟为$\mathsf{poly}(i) \cdot n^{r+O(1)}$的条件下枚举。

定理4中的一些蕴含关系此前已有所知，或者可以很容易地从已有工作中推导出来。横截秩和共形度在以下意义上是姊妹问题。回顾一下，$\mathcal{H} = \{V\setminus E \mid E \in \mathcal{H}\}$ 是补边所构成的超图。下面给出引理1关于共形超图的一种改述。

▶ **引理 16** (Berge 和 Duchet [6]；见 [5])。 </style}H 的横截秩至少为 k ，当且仅当 H 的共形度至少为 k。

注意，共形度至少为 k 等价于说 $\mathcal{H}$ 是 非$(k-1)$-共形的。补 $\mathcal{H}$ 可以在 $O(mn)$时间内由 $\mathcal{H}$ 得到。因此，当且仅当参数为 $k-1$ 的共形度问题可以在相同的时间界内求解时，便可以在 $\mathsf{poly}(m) \cdot n^{k+O(1)}$ 时间内求解参数为 k 的横截秩问题。还可以直接验证，$\mathsf{rank}(\mathsf{Tr}(\mathcal{H})) \geqslant k$ 所需的见证与 $\mathcal{H}$ 的 $(k-1)$-共形性的反例是相同的。事实上，设 S 满足：其所有大小至多为 $k-1$ 的子集都包含在 $\mathcal{H}$的一条超边中。这等价于这些子集均不是 $\mathcal{H}$的击中集。反过来， S 本身不包含于 $\mathcal{H}$ 的任何超边中，意味着 S 是一个击中集。因此，任何满足 $T \subseteq S$ 的极小击中集 $T \in \mathsf{Tr}(\mathcal{H})$的大小都至少为 k。这样的一个子集 T 可以在 $O(m|S|)$ 时间内由 S 得到（见附录A）。

现在假设我们能够以增量多项式时间枚举任意 r-超图的极小击中集，使得第 i 次延迟有界于 $\mathsf{poly}(i) \cdot n^{r+O(1)}$。显然，这也包括 r-均匀超图。将图（2-均匀超图）的情形直接推广，即可得知如何利用这一点枚举极大超团。设 $(V,\mathcal{H})$ 是秩为 r 的均匀超图。补图 $\mathcal{H}^{\mathsf{C}} = \{E \in \binom{V}{r} \mid E \notin \mathcal{H}\}$ 包含所有不是 $\mathcal{H}$ 的边的 r-子集。设 $C \subseteq V$ 是 $\mathcal{H}$ 中的一个极大超团，

那么 C 就是 $\mathcal{H}^{\mathsf{C}}$ 中的一个极大独立集，也就是说， C 不包含任何补边。这等价于 $V\backslash C$ 是 $\mathcal{H}^{\mathsf{C}}$ 的一个极小击中集。反过来，对于任意 $T \in \mathsf{Tr}(\mathcal{H}^{\mathsf{C}})$， $V\backslash T$ 都是 $\mathcal{H}$ 的一个极大团。补图 $\mathcal{H}^{\mathsf{C}}$ 可在 $O(mn^r)$ 时间内计算，因此当且仅当 $\mathsf{Tr}(\mathcal{H}^{\mathsf{C}})$ 可枚举时，$\mathcal{C}(\mathcal{H})$ 可按 $\mathsf{poly}(i)\cdot n^{r+O(1)}$ 的延迟枚举。

5.1 通过更快的枚举加快判定

接下来要做的是，双向建立横截秩/共形度相关判定问题与极小击中集/极大超团相关枚举问题之间的联系。我们首先假设可以使用一种算法，在迭代多项式时间内枚举一个 r-均匀超图的超团，耗时为 $\mathsf{poly}(i)\cdot n^{r+O(1)}$。由此我们推导出一个耗时为 $\mathsf{poly}(m)\cdot n^{k+O(1)}$ 的算法，用于求解非均匀输入超图$\mathcal{H}$ 在任意参数 k 下的共形度。

令 $[\mathcal{H}]_k = \{S \in \binom{V}{k} \mid \exists E \in \mathcal{H}\colon S \subseteq E\}$ 为超图$\mathcal{H}$的 k-截面，其中包含的超边是那些在$\mathcal{H}$的某条超边中共同出现的 k 个顶点组成的集合。

▶ **引理17.** 超图 H 是 k-共形的，当且仅当 $\mathcal{C}([H]_k) \subseteq H$。

证明。设 $C \in \mathsf{C}([\mathrm{H}]_k)$ 是 k-section 的一个极大超团。这意味着，它的每个大小至多为 k 的子集都包含在某条超边中。如果 H 是 k-共形的，那么 C 本身包含在 H 的某条超边 $E \in$ 中。此外，注意到 E 也是 $[\mathrm{H}]_k$ 的一个超团。由 C 的极大性可得 $C = E \in \mathrm{H}$。因此，由于 C 是任意的，我们有 $\mathsf{C}([\mathrm{H}]_k) \subseteq \mathrm{H}$。

反过来，设 S 是一个集合，其大小至多为 k 的每个子集都包含在$\mathcal{H}$ 的某条超边中。特别地， S 是 $[\mathcal{H}]_k$ 中的一个超团。设 C 是一个满足 $C \supseteq S$ 的极大超团。如果 $\mathsf{C}([\mathcal{H}]_k) \subseteq \mathcal{H}$ 确实成立，那么 C 就是$\mathcal{H}$ 中包含 S 的一条超边。换言之，$\mathcal{H}$ 是 k-共形的。◀

我们指出，引理17很容易加强为： k-共形性等价于$\mathcal{C}([\mathcal{H}]_k)$恰好包含$\mathcal{H}$中包含意义下的极大边。不过，我们只需要较弱的版本。

该引理给出了以下算法，可用于验证$\mathcal{H}$ 是否为 k-共形，或找出一个反例 S。我们首先在 $O(km\cdot\binom{n}{k}) = O(mn^k)$时间内计算$\mathcal{H}$ 的 k-截面。（注意， $\binom{n}{k} = O(n^k/k!)$，因此即使 k 不是常数，上述估计依然成立。）显然，$[\mathcal{H}]_k$ 是 k-均匀的。假定的超团枚举算法可以在 $i^c\cdot n^{k+O(1)}$ 时间内列出$\mathcal{C}([\mathcal{H}]_k)$中的第 i个成员，其中 c是某个常数。对于每个超团 C，我们检查$C \in \mathcal{H}$ 是否为一条边。我们将如何在 $O(n^2)$时间内执行此检查的说明留到本节末尾。最多枚举 $m+1$ 个超团后即可中止，因为第$m+1$个超团已能证明$\mathcal{C}([\mathcal{H}]_k) \nsubseteq \mathcal{H}$。

如果在任何时候我们找到了一个超团 $C \in \mathcal{C}([\mathcal{H}]_k)\backslash\mathcal{H}$，那么显然，$C$ 中基数至多为 k 的每个子集都包含在$\mathcal{H}$的某条超边中。我们声称， C 本身不包含在任何超边中，因此是搜索问题「共形度」的有效输出。根据定义， C 不等于 $\mathcal{H}$中的任何超边。如果它被某个 $E \in \mathcal{H}$严格 包含，那么 E 就会是 $[\mathcal{H}]_k$ 中一个扩展 C 的超团，这与 C的极大性矛盾。算法的总运行时间为

$$O(mn^k) + \sum_{i=0}^{m+1} i^c \cdot n^{k+O(1)} = (m+1)^{c+1}\cdot n^{k+O(1)} = \mathsf{poly}(m)\cdot n^{k+O(1)}.$$

为了支持检查团 C 是否属于 $\mathcal{H}$，我们花费 $O(mn)$ 的时间对超图进行预处理，将其存入一个字典，从而能够在 $O(|C|\cdot\log|\mathcal{H}|)$ 的时间内进行成员查询 $=O(n\log m)$。由于任意超图至多有 $m = O(2^n)$ 条边，因此查询时间为 $O(n^2)$。

5.2 通过更快判定实现更快枚举

现在我们证明这一等价关系的逆向结论。假设存在一个常数$c \geqslant 1$，使得横截秩可以在 $m^c \cdot n^{k+O(1)}$ 的时间内判定。我们声称，在这一假设下，可以对秩为 r 的任意超图 $\mathcal{H}$ 的极小击中集进行枚举，使得第 i 次延迟有界于 $i^c \cdot n^{r+O(1)}$。Eiter 和 Gottlob [25], 最早证明了该问题允许使用增量多项式算法（尽管其延迟要差得多）。他们反复调用判定子程序，以检查对于某个超图 $\mathcal{G}$，是否成立$\mathcal{G} = \mathrm{Tr}(\mathcal{H})$。利用下面的引理，该子程序的运行时间为 $|\mathcal{G}|^{r+2} \cdot \mathrm{poly}(n, |\mathcal{H}|)$。[4] 主要问题在于，该算法对于我们的目的而言过于缓慢。为了改善延迟，我们需要将对 $\mathcal{G}$ 大小的依赖降低到 $|\mathcal{G}|^c$ （即，使指数与 r无关）。相应地，我们愿意接受另一种 $n^{r+O(1)}$ 依赖。

下文中，我们令 $\mathrm{Tr}(\mathcal{G})|_r$ 表示$\mathcal{G}$ 的所有基数至多为 r 的极小击中集的集合。显然，$\mathrm{rank}(\mathrm{Tr}(\mathcal{G})) \leqslant r$ 成立当且仅当 $\mathrm{Tr}(\mathcal{G})|_r = \mathrm{Tr}(\mathcal{G})$。

▶ **引理18.** （Eiter 和 Gottlob [25]） 设 G、H 为两个超图，且满足 $r = \mathrm{rank}(H)$。则 $G = \mathrm{Tr}(H)$ 为真，当且仅当以下三个条件成立。
1. $\mathcal{G} \subseteq \mathrm{Tr}(\mathcal{H})$ - 所有边（属于 $\mathcal{G}$ ）都是极小击中集（针对 $\mathcal{H}$）；**2.** $\mathrm{Tr}(\mathcal{G})|_r \subseteq \mathcal{H}$ - 所有极小击中集，其针对 $\mathcal{G}$ 且至多有 r 个顶点者，都是边，属于 $\mathcal{H}$；**3.** $\mathrm{rank}(\mathrm{Tr}(\mathcal{G})) \leqslant r$ - 横截秩（ $\mathcal{G}$ 的）至多为 r。

检查顶点集合 $T \subseteq V$ 是否为 H 的极小击中集，可在时间 $O(|\mathrm{H}||T|)$ 内完成。因此，第一个条件可在时间 $|\mathrm{G}| \cdot O(|\mathrm{H}|n)$ 内检验。关于条件2，只需暴力枚举 V 的所有至多含 r 个顶点的子集，检查它们是否为 G 的极小击中集，若是，再检查它们是否也是 H 的一条超边。此外，后一项检验至多触发 $|\mathrm{H}|$ 次，因为 $|\mathrm{Tr}(\mathrm{G})|_r| > |\mathrm{H}|$ 已足以推出 $\mathrm{Tr}(\mathrm{G})|_r \not\subseteq \mathrm{H}$。整个验证所需时间为 $O(|\mathrm{G}|r\binom{n}{r} + |\mathrm{H}|r)) = |\mathrm{G}| \cdot O(n^r)$。后一估计利用了 H 是一个 r-超图，从而蕴含 $|\mathrm{H}| \leqslant \binom{n}{\leqslant r} = O(\binom{n}{r}) = O(n^r/r!)$。

问题出在条件 3。Eiter 和 Gottlob [25] 利用 Berge 和 Duchet 的结果 [6]，在时间 $O(|\mathcal{G}|^{r+2}n)$ 内判定该问题；我们则改用改进后的算法。显然，$\mathcal{G}$ 的横截秩至多为 r，当且仅当它是参数为 $r+1$ 的横截秩问题的一个否实例。我们在时间 $|\mathcal{G}|^c \cdot n^{r+O(1)}$ 内检查这一点。总而言之，这就得到了所需的方法，可在时间 $|\mathcal{G}|^c \cdot n^{r+O(1)}$ 内判定 $\mathcal{G} = \mathrm{Tr}(\mathcal{H})$。（另见定理 5。）

我们将其转化为一个枚举算法，用于枚举 $\mathrm{Tr}(\mathcal{H})$。该算法维护一个解的集合$\mathcal{G} \subseteq \mathrm{Tr}(\mathcal{H})$，并按阶段运行，阶段编号从 0 到 $|\mathrm{Tr}(\mathcal{H})|$。第 i 个阶段对应于目前已找到的极小击中集数量 $|\mathcal{G}|$。在每个阶段，算法都会检查$\mathcal{G} = \mathrm{Tr}(\mathcal{H})$ 是否已成立。如果成立，则终止。否则，它必须在 $\mathrm{Tr}(\mathcal{H})\backslash\mathcal{G}$ 中找到一个新的极小解。根据引理18，必有 $\mathrm{Tr}(\mathcal{G})|_r \not\subseteq \mathcal{H}$，或者 $\mathrm{rank}(\mathrm{Tr}(\mathcal{G})) > r$。

首先，假设 $\mathrm{Tr}(\mathcal{G})|_r \not\subseteq \mathcal{H}$，因此存在一个 $\mathcal{G}$ 的极小击中集 S，其中有 $|S| \leqslant r$ 个顶点，且它不是 $\mathcal{H}$ 的超边。$\mathcal{G} \subseteq \mathrm{Tr}(\mathcal{H})$ 的不变性表明，$\mathcal{G}$ 的任意超边都与 $\mathcal{H}$ 中的每个超边相交。反过来说，这也意味着 $\mathcal{H}$ 中的每条超边[5] H 都是 $\mathcal{G}$ 的击中集。由于 S 是极小的，如果存在某个 $H \in \mathcal{H}$ 满足 $H \subseteq S$，那么我们就有$H = S$，进而得到 $S \in \mathcal{H}$，矛盾。因此，对每个 $H \in \mathcal{H}$，都有 $H \not\subseteq S$。这等价于补集 $V\backslash S$ 是 $\mathcal{H}$ 的一个（不一定是极小的）击中集。由于$S \in \mathrm{Tr}(\mathcal{G})|_r$ 是 $\mathcal{G}$ 的击中集，因此 $\mathcal{G}$ 的任何超边 $G \in$ 都不完全包含于 $V\backslash S$。因此，

[4] 严格来说，Eiter 和 Gottlob [25] 只证明了当 $\mathcal{G}$ 和 $\mathcal{H}$ 均为斯佩纳时的引理18。为求完整起见，我们在附录 B 中给出一般情形的简短证明。[5] 在此证明中，我们用符号 H 表示 $\mathcal{H}$ 的边，以便更好地区分它们与$\mathcal{G}$的边。

任意 极小击中集 $T \subseteq V\backslash S$ 都属于 Tr($\mathcal{H}$)\$\mathcal{G}$。这种新的解 T 可以在时间 $O(|\mathcal{H}|n)$ $= n^{r+O(1)}$ 内由 S 计算得出。

请注意，我们使用的关于 S 的仅有两个性质： $S \in$Tr($\mathcal{G}$) 是 $\mathcal{G}$ 的一个极小击中集，并且它不是 $\mathcal{H}$ 的超边。在另一种情况下，即 rank(Tr($\mathcal{G}$)) $> r$ 时，参数为 $r+1$ 的横截秩算法会为 $\mathcal{G}$ 提供一个极小击中集S，且满足 $|S| > r$。$\mathcal{H}$ 的秩为 r，因此 $S \notin \mathcal{H}$。与上文相同的论证再次表明，任何满足 $T \subseteq V\backslash S$ 的 $\mathcal{H}$ 的极小击中集 T 都属于 Tr($\mathcal{H}$)\$\mathcal{G}$。总之，计算一个新解的主要开销是验证 $\mathcal{G} \neq$ Tr($\mathcal{H}$) 所需的 $i^c \cdot n^{r+O(1)}$ 时间。

5.3 定理 5 和6 的证明

上述结果还意味着，更优的枚举算法与更快地判定 $\mathcal{G}$ =Tr($\mathcal{H}$) 之间存在等价关系。

▶**定理 5**。 设 $\mathcal{H}$ 是一个秩为 r 的超图。以下命题等价。可以用增量延迟 $\mathcal{H}$ 的极小击中集，其增量延迟为 poly$(i)\cdot n^{r+O(1)}$。对于任意超图 $\mathcal{G}$，判定它是否恰好包含 $\mathcal{H}$ 的所有极小击中集，可以在 poly$(|\mathcal{G}|)\cdot n^{r+O(1)}$ 的时间内完成。

证明。其中一个方向已在第5.2小节的第一部分中给出。另一个方向可以用与上文第二部分类似的方式得到。假设可以访问一个黑盒过程，它判定 G =Tr(H) 是否成立，否则返回 (G\ Tr(H)) ∪ (Tr(H)\G) 中的某条超边。然后，从 G $= \emptyset \subseteq$ Tr(H) 出发并保持 G $\subseteq$ Tr(H)，可确保我们总能从 Tr(H)\G 中得到一条新超边，直到达到 G =Tr(H)。若该黑盒算法对 r-超图 H 的运行时间为 $|\mathrm{G}|^c \cdot n^{r+O(1)}$，则枚举的第 i 次延迟为 $i^c \cdot n^{r+O(1)}$。◀

定理4表明，为了得到横截秩问题乃至共形度的改进算法，我们需要r-均匀超图中极大超团的更好枚举算法。然而，对于最简单的情形 $r = 2$，即普通图中的极大团，已经有非常快的枚举算法。这给出了一种改进的共形性测试。

▶**定理 6.** 共形超图可在时间 $O(mn^3)$内被识别。

证明。图 $G = (V, E)$ 的极大团可以以延迟 $O(|E|n)$ =$O(n^3)$ 进行枚举，且与迄今已见的解的数量无关 [34, 41]。我们将此代入第5.1小节中的方法，以判定超图的共形性，即带有 $k = 2$ 的共形度。更具体地说，我们构造 2-截面 $[\mathrm{H}]_2$ （又名共现图或团扩张），它在 V 中任意一对共同出现于 H 的一条超边中的顶点之间有一条超边。枚举其极大团中的前 $m = |\mathrm{H}|$ 个所需时间为$O(mn^3)$。若枚举算法在第 m 个团之后仍未终止，或者其中某一个不是 H 的超边，则 H 不共形；否则它是共形的。◀

参考文献

1 Amir Abboud、Karl Bringmann、Holger Dell 和 Jesper Nederlof。SETH 与正交向量猜想被证伪所带来的更多推论。载于 计算理论研讨会第 *50* 届会议论文集（*STOC*），第 253–266 页，2018年。doi:10.1145/3188745.3188938。

2 Júlio Araújo、Marin Bougeret、Victor A. Campos 和 Ignasi Sau。计算最大极小阻塞集与击中集的参数化复杂性。Algorithmica，85：444–491，2023年。doi:10.1007/s00453-022-01036-5。3 Cristina Bazgan、Ljiljana Brankovic、Katrin Casel、Henning Fernau、Klaus Jansen、Kim- Manuel Klein、Michael Lampis、Mathieu Liedloff、Jérôme Monnot 和 Vangelis Th. Paschos。上支配的诸多方面。Theoretical Computer Science，717：2–25，2018年。doi:10.1016/j.tcs.2017.05.042。4 Austin R. Benson、Rediet Abebe、Michael T. Schaub、Ali Jadbabaie 和 Jon Kleinberg。单纯复形闭包与高阶链接预测。美国国家科学院院刊，115：E11221–E11230，2018年。 doi:10.1073/pnas.1800683115。5 Claude Berge。《超图——有限集组合学》，North-Holland 数学文库第 45 卷。North-Holland 出版公司，荷兰阿姆斯特丹，1989年。6 Claude Berge 和 Pierre Duchet。Gilmore 定理的一种推广。载于《第二届捷克斯洛伐克图论最新进展研讨会论文集》，第 49–55 页，1975年。7 Jan C. Bioch 和 Toshihide Ibaraki。正布尔函数的识别与对偶化复杂性。Information and Computation，123：50–63，1995年。 doi:10.1006/inco.1995.1157。8 Johann Birnick、Thomas Bläsius、Tobias Friedrich、Felix Naumann、Thorsten Papenbrock 和 Martin Schirneck。利用部分信息枚举击中集以发现唯一列组合。Proceedings of the VLDB Endowment，13：2270–2283，2020年。 doi:10.14778/3407790.3407824。9 Thomas Bläsius、Tobias Friedrich、Julius Lischeid、Kitty Meeks 和 Martin Schirneck。高效枚举数据剖析中产生的超图击中集。Journal of Computer a nd System Sciences，124：192–213，2022年。 doi:10.1016/j.jcss.2021.10.002。10 Nicolas Boria、Federico Della Croce 和 Vangelis Th. Paschos。关于最大最小顶点覆盖问题。Discrete Applied Mathematics，196：62–71，2015年。 doi:10.1016/j.dam.2014.06.001。11 Endre Boros、Vladimir Gurvich 和 Peter L. Hammer。正布尔函数的对偶子蕴含项。Optimization Methods and Software，10：147–156，1998年。 doi:10.1080/10556789808805708。12 Florent Capelli 和 Yann Strozecki。增量延迟枚举：空间与时间。Discrete Applied Mathematics，268：179–190，2019年。 doi:10.1016/j.dam.2018.06.038。13 Marco L. Carmosino、Jiawei Gao、Russell Impagliazzo、Ivan Mihajlin、Ramamohan Paturi 和 Stefan Schneider。强指数时间假设的非确定性扩展及其对不可归约性的推论。载于《第七届理论计算机科学创新会议（ITCS）论文集》，第 261–270 页，2016年。doi:10.1145/2840728.2840746。14 Jianer Chen、Xiuzhen Huang、Iyad A. Kanj 和 Ge Xia。通过参数化复杂性获得强计算下界。Journal of Computer and System Sciences，72：1346–1367，2006年。doi:10.1016/j.jcss.2006.04.007。15 Grant A. Cheston、Gerd H. Fricke、Stephen T. Hedetniemi 和 David P. Jacobs。关于上分数支配的计算复杂性。Discrete Applied Mathematics，27：195–207，1990年。 doi:10.1016/0166-218X(90)90065-K。16 Pierre Colomb 和 Lhouari Nourine。形式背景的键与共形超图。载于《第六届形式概念分析国际会议（ICFCA）论文集》，第 140–149 页，2008年。 doi:10.1007/978-3-540-78137-0_10。17 Colin Cooper、Alan M. Frieze 和 Wesley Pegden。随机二元矩阵的秩。Electronic Journal of Combinatorics，26，2019年。文章编号 P4.12。 doi:10.37236/8092。18 Marek Cygan、Fedor V. Fomin、Łukasz Kowalik、Daniel Lokshtanov、Dániel Marx、Marcin Pilipczuk、Michał Pilipczuk 和 Saket Saurabh。《参数化算法》。Springer，瑞士 Cham，2015年。doi:10.1007/978-3-319-21275-3。19 Peter Damaschke。用于双重超图对偶化（带秩限制）和最大极小顶点覆盖的参数化算法。Discrete Optimization，8：18–24，2011年。doi:10.1016/j.disopt.2010.02.006。

20 János Demetrovics 和 Vu Duc Thi。《键、反键与素属性》。Annales Uni- versitatis Scientiarum Budapestinensis de Rolando Eötvös Nominatae Sectio Computatorica，8:35–52，1987年。21 Andrzej Dorobisz 和 Jakub Kozik。《超图着色的局部计算算法——遵循 Beck 方法》。载于《自动机、语言与编程国际研讨会（ICALP）第50届会议论文集》，第48:1–48:20页，2023年。 doi:10.4230/LIPICS.ICALP.2023.48。22 Rodney G. Downey 和 Michael R. Fellows。《参数化复杂性的基础》。计算机科学丛书。Springer，英国伦敦，2013年。 doi:10.1007/978-1-4471-5559-1。23 Louis Dublois、Tesshu Hanaka、Mehdi K. Ghadikolaei、Michael Lampis 和 Nikolaos Melissinos。《最大极小反馈顶点集的（不可）近似性》。Journal of Computer and System Sciences，124:26–40，2022年。 doi:10.1016/J.JCSS.2021.09.001。24 Louis Dublois、Michael Lampis 和 Vangelis Th. Paschos。《上支配集：针对路径宽度的紧致算法与亚指数近似》。Theoretical Computer Science，923:271–291，2022年。 doi:10.1016/J.TCS.2022.05.013。25 Thomas Eiter 和 Georg Gottlob。《识别超图的极小横截集及相关问题》。SIAM计算杂志，24:1278–1304，1995年。 doi:10.1137/S0097539793250299。26 Thomas Eiter、Georg Gottlob 和 Kazuhisa Makino。《单调对偶化与超图横截集生成的新结果》。SIAM计算杂志，32:514–537，2003年。 doi:10.1137/S009753970240639X。27 Thomas Eiter、Kazuhisa Makino 和 Georg Gottlob。《单调对偶化的计算方面：简要综述》。Discrete Applied Mathematics，156:2035–2049，2008年。 doi:10.1016/j.dam.2007.04.017。28 Jörg Flum 和 Martin Grohe。《参数化复杂性理论》。理论计算机科学丛书，EATCS系列。Springer，德国柏林和海德堡，2006年。 doi:10.1007/3-540-29953-X。29 Fedor V. Fomin、Serge Gaspers、Dieter Kratsch、Mathieu Liedloff 和 Saket Saurabh。《迭代压缩与精确算法》。Theoretical Computer Science，411:1045–1053，2010年。doi:10.1016/J.TCS.2009.11.012。30 Alan M. Frieze 和 Dhruv Mubayi。《简单超图着色》。组合理论杂志B辑，103:767–794，2013年。 doi:10.1016/j.jctb.2013.09.003。31 Eva-Maria C. Hols、Stefan Kratsch 和 Astrid Pieterse。《顶点覆盖的消除距离、阻塞集与核》。SIAM离散数学杂志，36:1955–1990，2022年。 doi:10.1137/20M1335285。32 Russell Impagliazzo 和 Ramamohan Paturi。《关于 k-SAT的复杂性》。Journal of Computer and System Sciences，62:367–375，2001年。 doi:10.1006/jcss.2000.1727。33 Russell Impagliazzo、Ramamohan Paturi 和 Francis Zane。《哪些问题具有强指数复杂性？》。Journal of Computer and System Sciences，63:512–530，2001年。doi:10.1006/jcss.2001.1774。34 David S. Johnson、Christos H. Papadimitriou 和 Mihalis Yannakakis。《关于生成所有极大独立集》。Information Processing Letters，27:119–123，1988年。 doi:10.1016/0020-0190(88)90065-8。35 Richard M. Karp。《组合问题之间的可约性》。载于《计算机计算复杂性研讨会论文集》，第85–103页，1972年。doi:10.1007/978-1-4684-2001-2_9。36 Leonid Khachiyan、Endre Boros、Khaled Elbassioni 和 Vladimir Gurvich。《关于具有有界超边交的超图对偶化及其他相关超图类》。Theoretical Computer Science，382:139–150，2007年。 doi:10.1016/j.tcs.2007.03.005。37 Heikki Mannila 和 Kari-Jouko Räihä。《依赖推断》。载于《超大型数据库第13届国际会议（VLDB）论文集》，第155–158页，1987年。URL:https://www.vldb.org/conf/1987/P155.PDF。

38 Thorsten Papenbrock、Jens Ehrlich、Jannik Marten、Tommy Neubert、Jan-Peer Rudolph、Martin Schönberg、Jakob Zwiener 和 Felix Naumann。《函数依赖发现：七种算法的实验评估》。*Proceedings of the VLDB Endowment*，8:1082–1093，2015年。doi:10.14778/2794367.2794377。39 Raymond Reiter。《基于第一性原理的诊断理论》。人工智能，32:57–95，1987年。doi:10.1016/0004-3702(87)90062-2。40 Elias C. Stavropoulos、Vassilios S. Verykios 和 Vasileios Kagklis。《频繁项集隐藏问题的横截超图方法》。知识与信息系统，47:625–645，2016年。doi:10.1007/S10115-015-0862-3。41 Shuji Tsukiyama、Mikio Ide、Hiromu Ariyoshi 和 Isao Shirakawa。《一种生成所有极大独立集的新算法》。*SIAM*计算杂志，6:505–517，1977年。doi:10.1137/0206036。42 Meirav Zehavi。《以顶点覆盖为参数的最大极小顶点覆盖》。*SIAM*离散数学杂志，31:2440–2456，2017年。doi:10.1137/16M109017X。

A 引言中省略的证明

横截秩在时间 $O(m^{k+1} + m^k n)$。对于超图 $\mathcal{H}$，令 min($\mathcal{H}$) 为 $\mathcal{H}$ 的最小化，即由其包含关系下极小的边构成的子超图。注意，对任意 $\mathcal{H}$，min($\mathcal{H}$) 都是斯佩纳的。对于击中集而言，考虑最小化还是完整超图并无区别，我们有 Tr($\mathcal{H}$) = Tr(min($\mathcal{H}$))。此外，极小边恰好是击中集的极小击中集，min($\mathcal{H}$) =Tr(Tr($\mathcal{H}$))，参见 [5]。

不失一般性，我们可以假设 $k \geqslant 3$。这意味着，与横截秩的运行时间相比，计算 min(H) 所需的 $O(m^2 n)$ 时间可以忽略不计。因此，我们假设输入超图 H 是斯佩纳的。对于某个具有 k 条边的子超图 G $= \{E_1, E_2, \ldots, E_k\} \subseteq$ H，我们令 D(G) $= \bigcup_{1 \leqslant i < j \leqslant k} (E_i \cap E_j)$ 表示至少出现在 G 的两条边中的所有顶点的集合。采用这一记号，引理1指出，rank(Tr(H)) $\geqslant k$ 等价于存在一个 k-子超图，使得 $S = V \backslash D$(G) 是 H 的一个击中集。后者又等价于 D(G) 不包含 H 的任何超边。在较高层面上，我们的算法搜索 F$_i \subseteq$ H 的 k 子超图，每个含有 $k-1$条边，它们可以组合成某个 G，使得对所有 $E \in$H，$E \nsubseteq D$(G) 都成立。

我们首先给出一种计算集合 $D(\mathcal{G})$ 的替代方法，即

$$D(\mathcal{G}) = \bigcap_{i=1}^{k} (E_1 \cup \cdots \cup E_{i-1} \cup E_{i+1} \cup \cdots \cup E_k)\,. \tag{2}$$

要证明这一等式，注意到，任何至少属于 $\mathcal{G}$ 中两条不同边的顶点 $v \in V$，也包含在 $\mathcal{G}$ 中除一条边以外所有边的并集中。反过来，假设 $\mathcal{G}$ 中 v 的度至多为 1，并令 $E_i \in \mathcal{G}$ 为包含 v 的唯一边；如果不存在这样的边，则令 $E_i = E_1$。那么，$v \notin E_1 \cup \cdots \cup E_{i-1} \cup E_{i+1} \cup \cdots \cup E_k$。

下文中，我们假设 $\mathcal{H}$ 的边具有一个（任意的）顺序。固定一个具有 $k-1$ 条边的子超图 $\mathcal{F} \subseteq \mathcal{H}$。有序列表 $L(\mathcal{F})$ 包含 $\mathcal{H}$ 中所有满足 $E \in$ 且 $E \subseteq \bigcup \mathcal{F}$ 的边，并按照 $\mathcal{H}$ 所诱导的顺序排列。所有可能的 $k-1$-超图的有序列表，可以在总计 $\binom{m}{k-1} \cdot \big(O(kn) + O(mn)\big) = O(m^k n)$ 时间内计算出来。

对于某个含有 k 条边的$\mathcal{G} \subseteq \mathcal{H}$，设$\mathcal{F}_1, \ldots, \mathcal{F}_k \subseteq \mathcal{G}$是它所有恰好缺少一条边的子超图。根据公式 2，对于某个 $E \in \mathcal{H}$，存在 $E \subseteq D(\mathcal{G})$ 当且仅当 E 出现在每个列表 $L(\mathcal{F}_1) \ldots, L(\mathcal{F}_k)$ 中。等价地，若 $L(\mathcal{F}_1) \cap \cdots \cap L(\mathcal{F}_k) = \emptyset$，则$\mathcal{G}$见证 rank(Tr($\mathcal{H}$)) $\geqslant k$。

由于这些列表已排序且每个列表至多有 m 个条目，我们可以在 $O(km)$ 时间内求它们的交集。对输入$\mathcal{H}$ 的所有 k-边子超图执行此操作，总耗时为 $\binom{m}{k} \cdot O(km) = O(m^{k+1})$。请注意，最后这个上界并不假设 k 为常数。若搜索找到一个合适的子超图$\mathcal{G}$，则包含于 $V \setminus D(\mathcal{G})$ 中的任意极小击中集，其大小至少为 k。下面我们将说明如何在 $O(mn)$ 时间内计算出其中一个。

最小化击中集。有时我们会遇到这样的情况：给定某个超图的击中集 $S\ \mathcal{H}$，需要计算任意一个极小击中集$T \subseteq S$。标准贪心算法会尝试所有顶点 $s \in S$，并测试移除 $S\setminus\{s\}$ 后其是否仍为击中集；如果是，s 就会被移除。所得结果必然是一个极小击中集。瓶颈在于反复测试$\mathcal{H}$ 的所有超边是否都被击中，这导致总运行时间为 $O(m|S|^2)$。下面我们将展示如何将其降至 $O(m|S|)$。

算法首先用 $O(m|S|)$ 的时间构造 S的二部关联图。更具体地说，该图为 I，其顶点为 $S \cup \mathcal{H}$，满足当且仅当 $s \in E$时，$s \sim E$ 是一条边。由于 S 是一个击中集，任意超边 $E \in \mathcal{H}$ 的度 $\deg_I(E)$ 至少为 1。算法按如下方式构造集合 T。它首先遍历一遍超边 $E \in \mathcal{H}$，并且如果 $\deg_I(E) = 1$，就将唯一的邻居 $s \in N_I(E)$ 放入 T，并从关联图中移除其闭邻域$N_I[s]$，即 s 以及所有包含 s的超边。

如果这尚未耗尽划分 $\mathcal{H}$，那么所有剩余超边的度都至少为 2。在第二阶段，算法任意选择一条[6] 剩余的 $E \in \mathcal{H}$和它的一个邻居 s，并将 s 从图中移除（不将其加入 T，也不移除任何超边）。这不会使 $\mathcal{H}$ 中的任何元素成为孤立点，但可能导致其中一些元素的度变为 1。若如此，则按上述相同方式将唯一的邻居加入 T，从而减少剩余超边的数量。此过程会持续进行，直到所有超边都以这种方式被移除。

集合 T 是 $\mathcal{H}$ 的一个击中集，因为只有当某条超边的一个顶点被加入 T时，该超边才会被移除。度为1规则还确保每个 $s \in T$ 都有一条私有超边，因此 T 是极小的。关于运行时间的关键观察是，无论某个顶点$s \in S$ 是否被加入 T，处理它所需的时间都是 $O(\deg_I(s))$。更准确地说，存在一个通用常数 $c > 0$，使得该过程的运行时间为

$$O(m|S|) + \sum_{s \in S} c \cdot \deg_I(s) = O(m|S|).$$

B 引理18的证明

我们考虑超图上的一种二元关系。如果$\mathcal{G} \preccurlyeq \mathcal{H}$，那么$\mathcal{G}$ 的每条超边都包含$\mathcal{H}$ 的一条超边作为子集。注意，$\preccurlyeq$ 是一个预序，因为它具有自反性和传递性。此外，$\mathcal{G} \subseteq \mathcal{H}$ 意味着$\mathcal{G} \preccurlyeq \mathcal{H}$。当该关系限定于斯佩纳超图时，它甚至是反对称的（即构成一个偏序）。我们所用到的关键事实是，取击中集时该关系会反转，正如 Birnick、Bläsius、Friedrich、Naumann、Papenbrock 和 Schirneck 的以下结果所述 [8]

▶**引理19** (Birnick 等人 [8])。设 G 和 H 为超图。那么，$G \preccurlyeq H$ 成立，当且仅当$\mathrm{Tr}(G) \succcurlyeq \mathrm{Tr}(H)$。

[6] 对超边及其邻居采用不同的打破平局规则，可能会得到不同的极小集 $T \subseteq S$。我们忽略这些实现细节，因为我们只需要找到某个极小击中集。

▶ **引理18**。 (Eiter 和 Gottlob [25]) 设 G、H 为两个超图，且满足 $r = \mathrm{rank}(H)$。那么，$G = \mathrm{Tr}(H)$ 成立，当且仅当满足以下三个条件。
1. $\mathcal{G} \subseteq \mathrm{Tr}(\mathcal{H})$ - 所有属于 $\mathcal{G}$ 的边均为 $\mathcal{H}$ 的极小击中集；**2.** $\mathrm{Tr}(\mathcal{G})|_r \subseteq \mathcal{H}$ ——$\mathcal{G}$ 的所有至多含 r 个顶点的极小击中集都是 $\mathcal{H}$ 的边；**3.** $\mathrm{rank}(\mathrm{Tr}(\mathcal{G})) \leqslant r$ - $\mathcal{G}$ 的横截秩至多为 r。

证明。回顾可知，对 H 的最小化有 Tr(Tr(H) =min(H) 成立（见附录A）。假设我们确实有 G =Tr(H)。那么，条件1便不成问题。该假设进一步蕴含 $\mathrm{Tr(G)}|_r = \mathrm{Tr(Tr(H))}|_r = \mathrm{min(H)}|_r \subseteq$ min(H) $\subseteq$ H，以及 rank(Tr(G)) =rank(min(H)) $\leqslant$ rank(H) $= r$，此即后两个条件。

对于相反方向，我们断言：若三个条件均成立，则有 $\mathrm{min}(\mathcal{G}) = \mathrm{Tr}(\mathcal{H})$。包含关系 $\mathcal{G} \subseteq \mathrm{Tr}(\mathcal{H})$ 蕴含超图 $\mathcal{G}$ 是斯佩纳的，也就是说，$\mathrm{min}(\mathcal{G}) = \mathcal{G}$。因此，确立该断言便足以得到所求的 $\mathcal{G} = \mathrm{Tr}(\mathcal{H})$。

关于该论断，我们有 $\mathrm{min}(\mathcal{G}) \subseteq \mathcal{G} \subseteq \mathrm{Tr}(\mathcal{H})$，因此 $\mathrm{min}(\mathcal{G}) \preccurlyeq \mathrm{Tr}(\mathcal{H})$。条件2和3共同推出 $\mathrm{Tr}(\mathcal{G}) \subseteq \mathcal{H}$，因此 $\mathrm{Tr}(\mathcal{G}) \preccurlyeq \mathcal{H}$。将引理19应用于 $\mathrm{Tr}(\mathcal{G})$ 和 $\mathcal{H}$，得到 $\mathrm{min}(\mathcal{G}) = \mathrm{Tr}(\mathrm{Tr}(\mathcal{G})) \succcurlyeq \mathrm{Tr}(\mathcal{H})$。由于 $\mathrm{min}(\mathcal{G})$ 和 $\mathrm{Tr}(\mathcal{H})$ 都是斯佩纳的，关系 $\preccurlyeq$ 的反对称性给出 $\mathrm{min}(\mathcal{G}) = \mathrm{Tr}(\mathcal{H})$ ◀