中国科学: 信息科学 2024 年 第 54 卷 第 12 期: 2709–2726
SCIENTIA SINICA Informationis
论文

# 超图上最大独立集问题的精确算法

白天, 肖鸣宇*

电子科技大学计算机科学与工程学院, 成都 611731
* 通信作者. E-mail: myxiao@uestc.edu.cn

收稿日期: 2024–04–29; 修回日期: 2024–07–13; 接受日期: 2024–09–20; 网络出版日期: 2024–11–15

国家自然科学基金 (批准号: 62372095) 和四川省自然科学基金 (批准号: 2023NSFSC0059) 资助项目

**摘要** 最大独立集问题是计算机科学中最基础且重要的 NP 完全问题之一. 本文研究了超图上最大独立集问题 (MISH) 和超图上带惩罚的最大独立集问题 (PC-MISH) 的精确算法. 给定一个超图, MISH 问题旨在寻找图中最大的独立集, 其中, 超图中的独立集是一些两两不包含在同一超边的顶点构成的点子集. 而 PC-MISH 问题是 MISH 问题的松弛化变体. 在 PC-MISH 问题中, 仍然寻找一个点集 $X$, 但是它允许违背 "独立" 的性质, 也就是说, 允许超边包含 $X$ 中的多个顶点. 这个集合 $X$ 的价值定义为 $X$ 的大小减去包含至少两个 $X$ 中的顶点的超边数量. 而 PC-MISH 问题旨在找到一个价值最大的点集. 本文研究 MISH 和 PC-MISH 问题以 $n$ 以及 $\ell = n + m$ 为参数的精确算法, 其中 $n$ 是超图中顶点的个数, $m$ 是超图中超边的条数. 通过利用最大独立集问题在一般无向图上的精确算法可以直接得到 MISH 问题 $\mathcal{O}^*(1.1996^n)$ 时间的算法. 此外, 本文证明了 PC-MISH 问题可以在 $\mathcal{O}^*(1.9548^n)$ 时间内求解, 打破了穷举搜索的 $2^n$ 时间复杂度. 进一步, 本文重点给出 MISH 问题一个 $\mathcal{O}(1.1520^\ell)$ 时间的算法和 PC-MISH 问题一个 $\mathcal{O}(1.3982^\ell)$ 时间的算法, 分别改进之前时间为 $\mathcal{O}(1.1550^\ell)$ 和 $1.5^\ell 2^{o(\ell)}$ 的精确算法.

**关键词** 精确算法, 最大独立集问题, 带惩罚最大独立集问题, 超图, 最小子集反馈集问题

## 1 引言

NP 完全问题是计算理论中一个重要的概念, 自从 1971 年 Cook 提出以来, 它得到了广泛的研究和应用. 当 P $\neq$ NP 时, NP 完全问题不存在多项式时间算法, 这意味着我们无法找到传统的高效算法解决这些问题. 精确算法主要研究 NP 完全问题可以在多快的指数时间内精确求解, 在计算理论中产生了很多重要的理论和结果, 有着重大的影响和深远的意义 [1]. 事实上, 很多 NP 完全问题存在一种非常简单的指数时间复杂度的穷举搜索算法. 然而, 到目前为止, 一些 NP 完全问题的最佳算法仍是这种简单的穷举搜索算法, 即使尝试各种精巧的算法设计技术, 它们的运行时间仍未能得到改进. 例

**引用格式:** 白天, 肖鸣宇. 超图上最大独立集问题的精确算法. 中国科学: 信息科学, 2024, 54: 2709–2726, doi: 10.1360/SSI-2024-0129
Bai T, Xiao M Y. Exact algorithms for maximum independent set problem on hypergraphs (in Chinese). Sci Sin Inform, 2024, 54: 2709–2726, doi: 10.1360/SSI-2024-0129

 

如, 布尔可满足性问题 (Boolean satisfiability problem, SAT), 目前最好的算法仍是穷举搜索算法, 其运行时间为 $\mathcal{O}^*(2^n)=2^n\ell^{\mathcal{O}(1)}$, 其中 $n$ 为变量个数, $\ell$ 为输入大小, 此处符号 $\mathcal{O}^*(\cdot)$ 省略了多项式部分. 在强指数时间假设 (strong exponential time hypothesis, SETH) 假设下, 对任意常数 $\varepsilon>0$, SAT 问题不能在 $\mathcal{O}((2-\varepsilon)^n)$ 时间内求解 [2]. 另一个例子是旅行商问题 (travelling salesman problem, TSP), 在 20 世纪 60 年代已经提出了运行时间为 $\mathcal{O}(2^n n^2)$ 的动态规划算法 [3] (其中 $n$ 为图中的顶点数), 但过去 50 多年内的大量研究一直未能打破 $2^n$ 的时间复杂度瓶颈.

另外一些基础的 NP 完全问题精确算法的运行时间长期以来也不断地得到改进. 例如, 最小支配集问题 (dominating set problem) 可在 $\mathcal{O}(1.4689^n)$ 时间内解决 [4]; 最小多路割问题 (multiway cut problem) 可在 $\mathcal{O}(1.4661^n)$ 时间内解决 [5]; 最小反馈集问题 (feedback vertex set problem) 可在 $\mathcal{O}(1.7110^n)$ 时间内解决 [6]; 最小子集反馈集问题 (subset feedback vertex set problem, SFVS) 可在 $1.75^n 2^{o(n)}$ 时间内解决 [7,8]. 这些上界是否能继续改进的意义不仅在于解决该问题本身, 更在于更好地认识 NP 完全问题的计算复杂性. 在精确算法领域中, 最大独立集问题 (maximum independent set problem, MIS) 是最为重要的图算法问题之一, 其运行时间上界被大量学者深入研究, 在经过 30 多年的积累下该运行时间上界才打破 $\mathcal{O}(1.2^n)$, 目前最好的运行时间上界为 $\mathcal{O}(1.1996^n)$ [9]. 在这个时间复杂度下, 许多不太大的实例已经能够在短时间内进行求解. 本文考虑 MIS 问题推广至超图上的情况: 超图上最大独立集问题 (maximum independent set problem on hypergraphs, MISH) 和超图上带惩罚的最大独立集问题 (prize-collecting maximum independent set problem on hypergraphs, PC-MISH).

在图论中, 超图 (hypergraph) 是图的一种推广, 其中的超边可以包含任意数量的顶点. 相比之下, 一般的无向图中的一条边恰好包含两个顶点. 在超图上, 超边数量 $m$ 可以是顶点数 $n$ 的指数量级, 因此参数 $\ell=n+m$ 更多地被用于表征超图的规模. 在一个超图的点子集 $X$ 中, 如果任意两个顶点不同在一条超边中, 那么这个子集称为超图的一个大小为 $|S|$ 的独立集. 如果点子集 $X$ 在删除最少 $p$ 条超边之后是独立集, 这个子集称为超图的价值为 $|S|-p$ 的松弛独立集, 称 $p$ 为其惩罚值. MISH 问题旨在寻找超图中一个最大的独立集, 而 PC-MISH 问题旨在寻找超图中一个价值最大的松弛独立集.

MISH 问题是 MIS 问题的推广, 无论在理论还是实践中均为最为核心的组合优化 (combinatorial optimization) 问题之一. 此外, MISH 和 MIS 问题与最大团问题 (maximum clique problem) 和最小点覆盖问题 (minimum vertex cover problem) 有着非常紧密的联系. 它们在计算机视觉 (computer vision) [10]、社交网络 (social network) 覆盖率 [11]、选举中的合谋检测 (collusion detection) [12]、计算机图形学 (computer graphics) [13]、作业调度 (scheduling) [14] 等研究领域都有大量而至关重要的研究成果和应用. 而 PC-MISH 问题是 MISH 问题的松弛化版本, 更具有普适性和实用性. 在 MISH 问题中, 所求的集合必须为独立集 (即满足所有超边上独立性的约束), 这一要求在许多场合过于严格. 一个自然的松弛方式是: 可以通过花费一定的惩罚代价获得一个违背少量超边上独立性约束但是更大的点集. 因此, PC-MISH 问题应运而生. 这里强调, 在一般无向图上也可以定义带惩罚的最大独立集问题, 但是此问题与最大独立集问题是等价的. 在 PC-MISH 问题中, 违背任意一条超边上独立性约束的惩罚代价均为 1. 若需要将违背某些超边上独立性约束的惩罚代价定义为任意正整数, 可以通过添加该超边的多条重边多项式时间归约至 PC-MISH 问题. 事实上, 许多重要问题的带惩罚版本已经得到了深入的研究, 例如带惩罚的旅行商问题 (prize-collecting travelling salesman problem) [15]、带惩罚的斯坦纳树问题 (prize-collecting Steiner tree problem) [16] 和带惩罚的网络激活问题 (prize-collecting network activation problem) [17].

## 2 相关工作

MIS 问题的研究可以追溯至 1977 年 Tarjan 等 [18] 首次给出的非平凡精确算法, 该算法的时间复杂度为 $2^{n/3}n^{\mathcal{O}(1)}$, 其中 $n$ 为无向图顶点个数. 在 1986 年, Jian [19] 和 Robson [20] 分别将 MIS 问题的运算时间上界改进至 $\mathcal{O}(1.2346^n)$ 和 $\mathcal{O}(1.2109^n)$. 但是, MIS 问题是否能够在 $O(1.2^n)$ 时间内解决则受到了较大关注 [1]. 直到 2017 年, Xiao 等 [9] 才将运行时间改进至 $\mathcal{O}(1.1996^n)$.

MISH 问题是 MIS 问题在超图上的推广, 但是 MISH 问题可以多项式时间归约到 MIS 问题, 且保持解集大小和顶点数不变. 所以 MISH 问题也可以在 $\mathcal{O}^*(1.1996^n)$ 时间内求解. 事实上, 超图上相关研究通常以 $\ell = n + m$ 为参数设计精确算法, 其中 $n$ 和 $m$ 分别表示顶点数和超边数. 例如, Shi 等 [21] 研究了超图上的顶点覆盖问题 (vertex cover problem on hypergraphs), 给出了一个运行时间为 $\mathcal{O}(1.2381^\ell)$ 的精确算法. 此外, MISH 问题与另一个重要问题 —— 最小子集反馈集问题 (SFVS), 有着密切的联系. 借助分裂图 (split graph) 上 SFVS 问题的相关算法, MISH 问题可以在 $\mathcal{O}^*(1.1550^\ell)$ 时间以及和指数空间内求解 [22].

在 SFVS 问题中, 给定一个无向图和一个点子集 $T$ (称作关键点集合), 求解最小的点集 $S$, 使得删掉 $S$ 之后, 图中没有包括关键点的环. SFVS 问题有两种不同的版本, 限制版本 (restricted version) 和非限制版本 (unrestricted version). 在限制版本中, 解集 $S$ 不允许包含关键点; 而在非限制版本中, 解集 $S$ 可以包含任意顶点. 两个版本的 SFVS 问题是多个经典问题的推广. 例如, 当 $T$ 取为全体点集时, 非限制版的 SFVS 问题退化为最小反馈集问题; 当 $T$ 只包含一个顶点时, 限制版 SFVS 问题等价于最小多路割问题 [23]. 分裂图上两个版本的 SFVS 问题在分裂图上均是为 NP 难的 [24]. 如果一个无向图可以将点集划分为 $(I, K)$, 其中 $I$ 为独立集, $K$ 为团, 则称其为分裂图. 分裂图和超图的结构十分相似, 它们之间能够建立对应关系: 超图中的每个顶点与 $K$ 中的每个顶点一一对应; 超图中的每条超边和集合 $I$ 中的每个顶点一一对应, 使得超边包含的顶点恰与这条超边相应的顶点在 $K$ 中的邻居对应. 由此可见, 对于一个包含 $n$ 个顶点和 $m$ 条超边的超图, 可以用一个包含 $\ell = n + m$ 阶个顶点的分裂图表示, 且该分裂图的点集恰好可以划分为一个大小为 $m$ 的独立集和一个大小为 $n$ 的团. 特别地, 通过将分裂划分中的独立集设定为关键点集, MISH 问题和 PC-MISH 问题则分别可以利用分裂图上限制版和非限制版 SFVS 问题的算法进行求解.

在 $\ell$ 阶分裂图上, 限制版 SFVS 问题可以在 $\mathcal{O}(1.1550^\ell)$ 时间内求解 [22]; 而非限制版 SFVS 问题可以在 $1.5^\ell 2^{o(\ell)}$ 时间内求解 [25,26]. 所以在现有的文献中 MISH 问题和 PC-MISH 问题的精确算法运行时间上界为 $\mathcal{O}(1.1550^\ell)$ 和 $1.5^\ell 2^{o(\ell)}$. 值得强调的是, 目前已知最快的求解 PC-MISH 问题的算法仍是运行时间为 $\mathcal{O}^*(2^n)$ 的穷举搜索算法.

## 3 本文贡献

本文分别对 MISH 问题和 PC-MISH 问题的精确算法进行研究, 主要考虑以 $n$ 和 $\ell$ 为参数的精确算法, 其中 $\ell = n + m$ 为超图的点数与超边数之和. 事实上, 对于 MISH 问题, 以 $n$ 为参数的精确算法的研究较少, 原因之一是 MISH 问题的算法可以直接由 MIS 问题的算法得到. 此外, PC-MISH 问题尚未存在比穷举搜索要好的算法. 相较而言, 这两个问题的精确算法更多是以 $\ell$ 为参数开展研究. 在现有的文献中, MISH 问题可以在 $\mathcal{O}(1.1550^\ell)$ 的时间和指数空间进行求解 [22], 而 PC-MISH 问题可以在 $1.5^\ell 2^{o(\ell)}$ 时间内求解 [27]. 本文同时改进了这两个问题的 3 个精确算法结果, 具体结果如表 1 所示.

目前, MISH 问题的 $\mathcal{O}^*(1.1996^n)$ 时间的算法结果难以进一步改进. 而 MISH 问题的 $\mathcal{O}(1.1550^\ell)$ 时

**表 1　主要贡献**

**Table 1**　Main contributions

| Problem | Parameter $n$ | Parameter $\ell = n + m$ |
|---|---|---|
| MISH | $\mathcal{O}^*(1.1996^n)$ (Prop 1) | $\mathcal{O}(1.1520^\ell)$ (Thm 1) |
| PC-MISH | $\mathcal{O}^*(1.9489^n)$ (Thm 2) | $\mathcal{O}(1.3960^\ell)$ (Thm 2) |

间的算法需要消耗指数大小的空间 [22]. 本文提出的算法改进了求解此问题的时间复杂度上界, 同时将空间复杂度降为多项式大小. 特别地, 当在 $m < 0.28561n$ 时, 该算法同时优于运行时间为 $\mathcal{O}^*(1.1996^n)$ 和 $\mathcal{O}(1.1550^\ell)$ 的算法. 此外, 本文改进了 PC-MISH 问题以 $n$ 为参数的精确算法, 其运行时间为 $\mathcal{O}^*(1.9548^n)$, 成功突破了 $\mathcal{O}(2^n)$ 的时间复杂度上界.

事实上, 在许多场景的超图中, $m$ 远小于 $n$, 甚至满足 $m = o(n)$. 在此情况中以 $\ell$ 为参数和以 $n$ 为参数的精确算法的时间复杂度在指数时间意义下是等价的. 值得强调的是, 即使在 $m = o(n)$ 的限制下, MISH 问题依然是 NP 难的. 这是因为, 以 $\ell$ 为参数的 MISH 问题等价于以 $n + \mathrm{ecc}$ 为参数的 MIS 问题, 其中 ecc 表示边的团覆盖数 (edge-clique-cover number). 而最大度为 $\mathcal{O}(n^{1/2-\varepsilon})$ 的无向图的补图上, 有 $\mathrm{ecc} \leqslant \mathcal{O}(n^{1-2\varepsilon}\log n) = o(n)$ 成立 [28]. 通过在一个最大团问题的实例上添加冗余顶点并取补图进行归约, 能够说明 MISH 问题在 $m = o(n)$ 条件下的 NP 难性. 综上所述, 改进以 $\ell$ 为参数的算法有着显著的研究意义和价值.

本文指出, 分裂图上限制版 SFVS 问题和 MISH 问题可以在同样的指数时间内解决; 分裂图上非限制版 SFVS 问题与 PC-MISH 问题也可以在同样的指数时间内解决. 基于此结论, 可以得出推论: 分裂图上的限制版和非限制版子集反馈集问题分别能够在 $\mathcal{O}(1.1520^n)$ 和 $\mathcal{O}(1.3982^n)$ 时间内求解, 其中 $n$ 表示分裂图的顶点数.

本文所设计的算法为分支搜索算法 (branch-and-search algorithm). 核心思想是结合超图的结构性质进行深入分析, 得到一系列有效的约简规则 (reduction rule) 和分支规则 (branching rule), 并进行合理地组合. 此外, 本文使用了度量治之方法 (measure-and-conquer method) [29] 分析算法的时间复杂度, 并且采用了两种不同类型的度量治之方法分别分析了求解 MISH 问题和 PC-MISH 问题的算法时间复杂度.

## 4　基础内容

### 4.1　符号系统与问题定义

本文研究的对象为超图, 用符号 $G = (V, E)$ 表示, 其中集合 $V$ 表示点集, 多重集 $E$ 表示超边集. 每一条超边 $e \in E$ 是点集 $V$ 的一个子集. 超边 $e$ 的秩定义为 $e$ 所包含的顶点个数, 记作 $\mathrm{rank}(e)$. 本文考虑的超图允许存在秩为 0 或 1 的超边. 若超图 $G$ 的所有超边的秩至多为 $r$, 则称为这个超图的秩为 $r$, 记作 $\mathrm{rank}(G) = r$. 本文使用 $n = |V|$ 表示图 $G$ 的顶点数, 用 $m = |E|$ 表示图 $G$ 的超边数, 并且令参数 $\ell = n + m$. 本文中, $n$ 阶图是指顶点数为 $n$ 的图, 集合 $\{1, 2, \ldots, n\}$ 简记作 $[n]$.

在不引起歧义的情况下, 如果一个点子集仅仅包含一个元素, 如 $\{v\}$, 本文简写为 $v$. 点子集 $X$ 的边邻居, 记作 $N_E(X)$, 表示包含了 $X$ 中至少一个顶点的超边构成的集合. 点子集 $X$ 的点邻居, 记作 $N_V(X)$, 表示不在 $X$ 中但与 $X$ 中的顶点相连的顶点构成的集合; 并且令 $N_V[X] = N_V(X) \cup X$. 顶点 $v$ 的边度和点度分别记作 $\deg_E(v) = |N_E(v)|$ 和 $\deg_V(v) = |N_V(v)|$.

对于超图 $G = (V, E)$ 的点子集 $X \subseteq V$ 和超边子集 $Y \subseteq E$, 定义 $Y$ 限制在 $X$ 上的超边集

$Y|_X = \{e \cap X \mid e \in Y\}$. 由 $X \cup Y$ 导出的子超图定义为子图 $(X, Y|_X)$, 记作 $G[X,Y]$; 由点子集 $X$ 导出的子超图简记为 $G[X] = G[X,E]$. 同时, 对于子集 $S = X \cup Y$, 将 $G[V \setminus X, E \setminus Y]$ 简写为 $G - S$.

如果超图 $G$ 的一个点子集 $X \subseteq V$ 所导出子图的秩 $\mathrm{rank}(G[X]) \leqslant 1$, 则称 $X$ 为 (超图 $G$ 的) 独立集. 超图 $G$ 中点数最多的独立集称为最大独立集. 超图 $G$ 的点子集 $X$ 在删除最少 $p$ 条超边后, 是一个独立集, 则称其是价值为 $|X| - p$ 的松弛独立集. 超图 $G$ 中价值最大的松弛独立集称为最大松弛独立集, 其价值可表示为 $n - |S|$, 其中 $S \subseteq V \cup E$ 是满足 $\mathrm{rank}(G - S) \leqslant 1$ 的最小集合.

求解超图 $G$ 中最大独立集的问题称为超图上的最大独立集问题, 求解超图 $G$ 的最大松弛独立集的问题称为超图上的带惩罚最大独立集问题. 为方便叙述, 本文给出超图上最大独立集问题的一个广义形式作为研究对象, 并将该问题转化为对应的最小化问题进行描述. 这个广义形式主要体现在输入部分额外给定一个由顶点和超边构成的子集 $F$, 要求解集中不能包含 $F$ 中的元素. 该问题形式化定义如下.

**超图上广义最大独立集问题 (G-MISH)**

**输入:** 一个超图 $G = (V, E)$, 一个子集 $F \subseteq V \cup E$, 以及一个正整数 $k \in \mathbb{N}$.

**输出:** 判断是否存在一个大小不超过 $k$ 且与 $F$ 不相交的集合 $S \subseteq V \cup E$, 使得 $\mathrm{rank}(G - S) \leqslant 1$, 若存在则输出一个满足条件的集合 $S$.

一个 G-MISH 问题的实例 (instance) 记作 $\mathcal{I} = (G, F, k)$. 满足输出条件的集合 $S$ 称为实例 $\mathcal{I}$ 的解 (solution), 大小最小的解称为最优解. 一个实例可能不存在满足条件的解. 对于存在满足条件的解 $S$ 的实例又称为有解实例. 根据定义, 当 $F = E$ 时, G-MISH 等价于 MISH; 当 $F = \varnothing$ 时, G-MISH 等价于 PC-MISH. 因此, 当求解 MISH 或 PC-MISH 问题时, 只需令 $F = E$ 或 $F = \varnothing$, 然后遍历 $k$ 的值求解 G-MISH 问题即可得到最优解 $S$, 进而 $V \setminus S$ 便为最大独立集或最大松弛独立集. 此外, 本文研究 G-MISH 问题的另一个原因是该问题有助于描述分支搜索算法: 在算法中, 一些分支规则会将一个顶点直接加入 $F$ 中.

### 4.2 分支搜索算法与度量治之技术

本文设计的算法采用了标准的分支搜索算法范式 (参见文献 [1, 2]). 分支搜索算法一般由多个约简规则和分支规则组成. 约简规则将输入实例 $\mathcal{I}$ 转化为 "较小" 的实例 $\mathcal{I}'$. 若输入实例 $\mathcal{I}$ 有解当且仅当输出实例 $\mathcal{I}'$ 有解, 则称此约简规则是正确的. 分支规则将当前实例 $\mathcal{I}$ 分成若干子实例 $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_r$. 若输入实例 $\mathcal{I}$ 有解当且仅当至少一个子实例有解例, 则称此分支规则是正确的.

为了量化每一条规则对实例规模的减小程度, 需要为实例选择一个度量函数 $\mu$. 这个度量函数需要保证在算法的每一步中, 度量单调非增, 并且度量小于等于零时, 此问题可以在多项式时间内解决. 形式化地, 若某一约简规则将输入实例 $\mathcal{I}$ 转化为实例 $\mathcal{I}'$, 则应满足 $\mu(\mathcal{I}') \leqslant \mu(\mathcal{I})$. 若某一分支规则将输入实例 $\mathcal{I}$ 分成 $r$ 个子实例 $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_r$, 则应满足 $\mu(\mathcal{I}_i) < \mu(\mathcal{I})$ $(i \in [r])$. 在具体的分支搜索算法分析中, 度量函数有许多种不同形式. 例如, 可以为每一个顶点设置不同权重, 或者为一些特殊的集合或参数 (如解集大小) 设置权重, 而度量为这些权重之和.

为了分析分支搜索算法的运行时间, 需要估计分支搜索算法所生成的搜索树的大小. 令 $R(\mu)$ 表示算法生成的关于度量函数 $\mu$ 的搜索树的大小. 对于将输入实例 $\mathcal{I}$ 分成 $r$ 个子实例 $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_r$ 的分支规则, 可以建立递归关系 $R(\mu) \leqslant \sum_{i \in [r]} R(\mu_i) + 1$, 其中 $\mu_i = \mu(\mathcal{I}_i)$ $(i \in [r])$. 此递归关系通常用向量 $(\mu - \mu_1, \mu - \mu_2, \ldots, \mu - \mu_r)$ 表示, 它称作该分支规则的分支向量 (branching vector). 分支向量的分支因子为 (branching factor) 方程 $x^{\mu} - \sum_{i=1}^{r} x^{\mu_i} = 0$ 的唯一正实数根.

若分支向量 $\boldsymbol{a} = (a_1, a_2, \ldots, a_r)$ 和 $\boldsymbol{b} = (b_1, b_2, \ldots, b_{r'})$ 满足 $r \leqslant r'$ 且 $a_i \geqslant b_i$ $(i \in [r'])$, 则称 $\boldsymbol{a}$ 不

劣于 $\boldsymbol{b}$. 事实上, 若 $\boldsymbol{a}$ 不劣于 $\boldsymbol{b}$, 则分支向量 $\boldsymbol{a}$ 的分支因子不大于 $\boldsymbol{b}$ 的分支因子. 如果分支搜索算法中各分支规则中最大的分支因子是 $\alpha$, 那么此时分支搜索算法中生成的搜索树大小为 $\mathcal{O}(\alpha^{\mu})$, 若算法中每一步都能在多项式时间内被执行, 则算法有时间复杂度上界 $\mathcal{O}(\alpha^{\mu})$.

### 4.3 超图上广义独立集问题的性质

解决 G-MISH 问题最简单的分支搜索思想是对某个顶点或超边 $x$ 分两种情况搜索: 要么 $x$ 在解中, 要么 $x$ 不在解中. 对于前者, 可以将 $x$ 从图中删去; 而对于后者, 则可以将 $x$ 加入 $F$ 中. 本小节首先给出这两种基本操作的一般性质.

**引理1** 设 $\mathcal{I}=(G,F,k)$ 为 G-MISH 问题的有解实例. 若存在 $\mathcal{I}$ 的解包含顶点或超边 $x$, 则删除 $x$ 后的实例 $\mathcal{I}'=(G-x,F,k-1)$ 的任意解与 $\{x\}$ 的并集是 $\mathcal{I}$ 的解. 若存在 $\mathcal{I}$ 的解不包含顶点或超边 $x$, 则将 $x$ 加入 $F$ 后的实例 $\mathcal{I}''=(G,F\cup\{x\},k)$ 的任意解是 $\mathcal{I}$ 的解.

**证明** 设 $S'$ 为实例 $\mathcal{I}'$ 的任一解, 令 $S=S'\cup\{x\}$. 一方面, $\mathrm{rank}(G'-S')\leqslant 1$ 且有 $G-S=G'-S'$ 成立; 另一方面, 有大小关系 $|S|=|S'|+1\leqslant k$. 所以, $S$ 为 $\mathcal{I}$ 的一个包含 $x$ 的解.

设 $S''$ 为实例 $\mathcal{I}''$ 的任一解, 令 $S=S''$. 一方面, $\mathrm{rank}(G''-S'')\leqslant 1$ 且有 $G-S=G''-S''$ 成立, 又根据 $S''$ 的定义, $S$ 不包含 $x$; 另一方面, 有大小关系 $|S|=|S''|\leqslant k$. 所以, $S$ 为 $\mathcal{I}$ 的一个不包含 $x$ 的解.

**引理2** 设 $\mathcal{I}=(G,F,k)$ 为 G-MISH 问题的有解实例. 令顶点 $v\in F$ 且包含在超边 $e\in N_E(v)$ 中. 对于 $\mathcal{I}$ 的任一解 $S$, 要么超边 $e\in S$ 成立, 要么点子集 $e\setminus\{v\}$ 中所有顶点都在 $S$ 中, 即 $e\setminus\{v\}\subseteq S$ 成立. 进一步, 若 $E\subseteq F$, 那么集合 $N_V(v)\subseteq S$ 必成立.

**证明** 考察集合 $e\setminus\{v\}$ 中的任意顶点 $u$, 可见顶点 $u$ 与 $v$ 均包含在超边 $e$ 中. 又由于 $v\in F$, 所以要么顶点 $u\in S$, 要么 $e\in S$. 由于 $u$ 的任意性, 可以得出, 要么超边 $e\in S$, 要么 $e\setminus\{v\}\subseteq S$.

进一步, 若 $E\subseteq F$, 考察任意包含了 $v$ 的超边 $e$, 由于超边 $e$ 不在解 $S$ 中, 说明 $e\setminus\{v\}$ 全体均在解 $S$ 中. 由于 $e$ 的任意性, 可以得出 $N_V(v)\subseteq S$ 成立.

接下来将介绍 G-MISH 问题中关于超边的 3 个简单而重要的结构性质. 这些性质将多次应用于 MISH 问题和 PC-MISH 问题的算法正确性论述中.

**引理3** 设 $\mathcal{I}=(G,F,k)$ 是 G-MISH 问题的有解实例, $e\in E$ 为一条超边. 若 $\mathrm{rank}(e)=2$, 则存在解 $S$ 不包含 $e$; 若 $\mathrm{rank}(e)=3$, 则存在一个解 $S$ 要么不包含超边 $e$, 要么 $e\in S$ 且 $e$ 中的 3 个顶点均不在解 $S$ 中.

**证明** 若 $\mathrm{rank}(e)=2$, 设 $S$ 为包含超边 $e$ 的解, 则将 $e$ 替换为其包含的任一顶点 $v\in e$ 而得到的集合 $S'=(S\setminus e)\cup\{v\}$ 也为一个解. 这是因为删掉 $v$ 之后, 超边 $e$ 的秩已经变为了 1.

若 $\mathrm{rank}(e)=3$ 且存在解 $S$ 包含了超边 $e$. 如果 $e$ 所包含的某一顶点 $v\in e$ 在解 $S$ 中, 那么在删除 $v$ 之后, 超边 $e$ 的秩变为 2. 此时, 将超边 $e$ 替换为 $e$ 所包含的另一个与 $v$ 不同的顶点 $u\in e\setminus\{v\}$ 而得到的集合 $S''=(S\setminus e)\cup\{u\}$ 也为一个解. 综上所述, 要么存在解不包含超边 $e$, 要么存在解仅包含超边 $e$ 而不包含 $e$ 中任何顶点.

**引理4** 设 $\mathcal{I}=(G,F,k)$ 是 G-MISH 问题的有解实例. 若某一顶点 $v$ 仅包含在一条超边 $e$ 中, 则存在解不包含 $v$.

**证明** 顶点 $v$ 只包含在超边 $e$ 中, 删除超边 $e$ 后, $v$ 总在最大独立集中. 因此, 若有解 $S$ 包含了 $v$, 则将 $v$ 替换为超边 $e$ 得到的集合 $S'=(S\setminus\{v\})\cup\{e\}$ 也为一个解. 所以总存在解不包含 $v$.

下文引理 5 是引理 2 的一个推广.

**引理5** 设 $\mathcal{I}=(G,F,k)$ 是 G-MISH 问题的有解实例, $e\in E$ 是任意一条超边, $S$ 是任意一个解. 令 $(N_1,N_2)$ 是 $e$ 包含的顶点的一个划分. 则要么超边 $e\in S$, 要么 $N_1\subseteq S$, 要么 $N_2\subseteq S$. 进一步, 若超边 $e\in F$, 要么 $N_1\subseteq S$, 要么 $N_2\subseteq S$.

**证明** 只需证明, 若超边 $e$ 不在解中, $S$ 至少包含 $N_1$ 和 $N_2$ 其中之一. 采用反证法. 若 $N_1$ 和 $N_2$ 均包含至少一个不属于 $S$ 的顶点中, 分别设为 $v_1\in N_1\setminus S$ 和 $v_2\in N_2\setminus S$. 那么 $v_1$ 和 $v_2$ 同时在点集 $e$ 中, 导出了矛盾. 所以, $S$ 至少包含 $N_1$ 和 $N_2$ 其中之一.

## 5 超图上最大独立集问题的精确算法

MISH 问题可以转换成一般无向图上的 MIS 问题. 将超图 $G$ 转换成一个具有相同顶点集合的 $n$ 阶无向图 $G'$: 无向图 $G'$ 的顶点集与 $G$ 的顶点集相同, 图 $G'$ 中两个顶点间有一条边当且仅当这两个顶点同时出现在 $G$ 中某条超边中. 不难发现一个顶点集是 $G$ 的独立集当且仅当它是 $G'$ 的独立集. 故可使用 MIS 的运行时间为 $\mathcal{O}(1.19951^n)$ 的精确算法求解 MISH 问题 [9].

**命题1** MISH 问题可以在 $\mathcal{O}^*(1.19951^n)$ 时间内求解.

命题 1 将在后续算法设计中被使用, 接下来将首先给出求解 MISH 问题的运行时间为 $\mathcal{O}(1.1520^\ell)$ 的算法. 本节内容将分为算法设计与算法分析两部分.

### 5.1 算法设计

G-MISH 问题在 $F=E$ 时退化为 MISH 问题. 设 $\mathcal{I}=(G,F,k)$ 为 G-MISH 问题的一个实例, 其中 $E\subseteq F$ 总是成立, 且 $F$ 的初始设置为 $E$. 本小节首先给出求解该实例的 7 条简单的约简规则. 注意到, 部分约简规则建立在 $E\subseteq F$ 这一事实之上, 因此这些约简规则不一定适用于一般的 G-MISH 问题.

(1) 若 $k<0$ 或有两个 $F$ 中的顶点包含在同一条超边中, 则返回 “无解”.

(2) 若 $k\geqslant 0$ 且 $\mathrm{rank}(G)\leqslant 1$, 返回 $\varnothing$.

(3) 若存在超边 $e$ 和 $e'$ 满足 $e\subseteq e'$, 则直接移除 $e$.

(4) 若存在超边 $e$ 有 $\mathrm{rank}(e)\leqslant 1$, 则直接移除 $e$.

(5) 若存在不属于任何超边的顶点 $v$, 则直接移除 $v$.

(6) 若存在顶点 $v\in F$, 则删除 $N_V(v)$ 并放入解, $k$ 减小 $\deg_V(v)$.

(7) 若存在顶点 $v$ 仅包含在一条超边中, 则将 $v$ 加入 $F$.

上述 7 条基本约简规则的正确性可以直接由引理 2, 4, 5 得出. 若上述基本约简规则对于实例 $\mathcal{I}$ 均不可执行, 则称 $\mathcal{I}$ 为 I 类约简实例. 容易发现, 在 I 类约简实例中, $F=E$ 总成立, 且每个顶点至少包含在两条超边中.

接下来给出求解 MISH 问题的分支搜索算法 $\mathcal{A}_{\mathrm{MISH}}$ (如算法 1 所示), 其核心思想是对一个顶点 $v$ 进行分支: 要么 $v$ 在解中; 要么 $N_V(v)$ 在解中.

算法 $\mathcal{A}_{\mathrm{MISH}}$ 包括 5 个步骤, 采用递归算法的形式进行描述, 即每一步要么输出解, 要么调用算法自身.

步骤 (2) 的正确性可以直接由命题 1 得到. 在步骤 (3) 中, 由引理 5 可知, 要么 $A=e_1\cap e_2$ 在解中; 要么 $e_1\setminus A$ 以及 $e_2\setminus A$ 包含的顶点在解中, 即 $B$ 在解中. 此外, 在删除 $B$ 后, $e_1$ 和 $e_2$ 所包含的顶点完全相同, 故可再删去超边 $e_1$. 因此得证该步骤的正确性. 类似地, 步骤 (4) 和 (5) 的正确性可以由引理 2 和 5 直接得到. 综上所述, 算法 $\mathcal{A}_{\mathrm{MISH}}$ 是正确的.

进一步, 通过对算法 $\mathcal{A}_{\mathrm{MISH}}$ 的简单观察, 可以得出

**算法 1** MISH 问题的分支搜索算法 $\mathcal{A}_{\text{MISH}}$

**输入:** 超图 $G=(V,E)$, 一个子集 $F$(满足 $E\subseteq F\subseteq V\cup E$) 和一个正整数 $k$, 其中 $F$ 的初始值为 $E$.

**输出:** 判断是否有解, 若有解则输出一个大小不超过 $k$ 且与 $F$ 不相交的解集 $S\subseteq V\cup E$, 使得 $\text{rank}(G-S)\leqslant 1$.

1: 判断输入实例是否为 I 类约简实例, 若非 I 类约简实例, 则依次执行 7 条基本的约简操作.

2: 若 $\text{rank}(G)\leqslant 7$, 根据命题 1, 在 $\mathcal{O}^*(1.19951^n)$ 时间内直接求解.

3: 如果存在两个秩至少为 8 的超边 $e_1,e_2\in E$ 满足 $|e_1\cap e_2|\geqslant 2$, 则令 $A=e_1\cap e_2$ 且 $B=(e_1\cup e_2)\setminus A$, 计算

- $S_1=A\cup\mathcal{A}_{\text{MISH}}(G-A,F,k-|A|)$;
- $S_2=B\cup\mathcal{A}_{\text{MISH}}(G-B-\{e_1\},F\setminus\{e_1\},k-|B|)$,

若均无解, 则返回 "无解"; 否则返回 $S_1$ 和 $S_2$ 中的较小者.

4: 如果存在一个顶点 $v$ 包含在两个秩至少为 8 的超边 $e_1,e_2$ 中, 则计算

- $S_1=\{v\}\cup\mathcal{A}_{\text{MISH}}(G-v,F,k-1)$;
- $S_2=N_V(v)\cup\mathcal{A}_{\text{MISH}}(G-(N_V(v)\cup\{e_1,e_2\}),F\setminus\{e_1,e_2\}),k-\deg_V(v))$,

若均无解, 则返回 "无解"; 否则返回 $S_1$ 和 $S_2$ 中的较小者.

5: 找到一个秩至少为 8 的超边 $e$, 则将 $e$ 包含的顶点划分为两部分 $e=N_1\cup N_2$ 使得 $|N_1|=4$ 且 $|N_2|=\text{rank}(e)-4$, 计算

- $S_1=N_1\cup\mathcal{A}_{\text{MISH}}(G-N_1,F,k-|N_1|)$;
- $S_2=N_2\cup\mathcal{A}_{\text{MISH}}(G-N_2,F,k-|N_2|)$,

若均无解, 则返回 "无解"; 否则返回 $S_1$ 和 $S_2$ 中的较小者.

- 当算法执行完步骤 (2) 后, 图中至少存在一条秩不少于 8 的超边;
- 当算法执行完步骤 (3) 后, 任意两条秩不少于 8 的超边最多包含一个公共顶点;
- 当算法执行完步骤 (4) 后, 任意两条秩不少于 8 的超边均无公共顶点, 且步骤 (5) 总能够执行.

值得强调地是, 在求解 MISH 问题时, 初始的实例满足 $F=E$, 且算法 $\mathcal{A}_{\text{MISH}}$ 的任何一条步骤均未将超边从 $F$ 中移出 (除非直接删除), 所以 $E\subseteq F$ 总是成立.

### 5.2 基于度量治之的时间复杂度分析

本小节将使用度量治之技术分析算法 $\mathcal{A}_{\text{MISH}}$ 的时间复杂度. 首先定义各超边的权重: 秩为 $r$ 的超边权重为 $\omega_r=\min\{1,\lambda\cdot r\}$, 其中 $\lambda=1/7$. 实例 $\mathcal{I}$ 的度量函数定义为顶点数与所有超边权重之和, 即

$$\mu=\mu(\mathcal{I})=n+\sum_{r\in\mathbb{N}}\omega_r\cdot m_r,$$

其中 $m_r$ 表示秩为 $r$ 的超边数量.

下文将使用度量函数 $\mu$ 完成算法 $\mathcal{A}_{\text{MISH}}$ 的时间复杂度分析. 首先, 当 $\mu(\mathcal{I})\leqslant 0$ 时, 图中仅存在一些秩为 0 的超边, 所以可以在多项式时间内求解. 其次, 7 条基本约简规则至多执行多项式次, 并且均能够在多项式时间内完成; 此外这 7 条约简规则只会将一些顶点删除, 这不会使得度量 $\mu$ 增加. 因此, 以下分析总假设实例 $\mathcal{I}$ 为 I 类约简实例. 后续将分析算法中各分支规则的分支因子.

**引理6** 若 MISH 问题的 I 类约简实例 $\mathcal{I}$ 中超图的秩 $\text{rank}(G)\leqslant 7$, 则 $\mathcal{I}$ 可在 $\mathcal{O}^*(1.15199^{\mu})$ 时间内求解.

**证明** 若 I 类约简实例 $\mathcal{I}$ 中超图的秩 $\text{rank}(G)\leqslant 7$, 则步骤 (1) 将执行. 此时算法将直接调用 MIS 问题的精确算法在 $\mathcal{O}^*(1.19951^n)$. 设所有超边秩之和为 $R=\sum_{e\in E}\text{rank}(e)$. 根据条件, 每条超边的秩不超过 7. 所以

$$\mu=n+\sum_{e\in E}\lambda\cdot\text{rank}(e)=n+\lambda\cdot R.$$

**表 2 步骤 (3) 中各种情形下的分支向量及其分支因子**

**Table 2** Branching vectors and branching factors of step (3)

| $\|A\|$ | $\|A_1\|$ | $\|A_2\|$ | Branching vector (not worse than) | Branching factor (no more than) |
|---|---|---|---|---|
| 2 | $\geqslant 6$ | $\geqslant 6$ | $(2, 14-2\lambda)$ | 1.12216 |
| 3 | $\geqslant 5$ | $\geqslant 5$ | $(3, 12-3\lambda)$ | 1.11607 |
| 4 | $\geqslant 4$ | $\geqslant 4$ | $(4, 10-4\lambda)$ | 1.11600 |
| 5 | $\geqslant 3$ | $\geqslant 3$ | $(5, 8-5\lambda)$ | 1.12100 |
| 6 | $\geqslant 2$ | $\geqslant 2$ | $(6, 6-6\lambda)$ | 1.13278 |
| 7 | 1 | 1 | $(9-2\lambda, 3)$ | 1.13863 |
| 7 | 1 | $\geqslant 2$ | $(7, 4)$ | 1.13819 |

进一步, 每个顶点至少包含在两条超边中, 那么 $R \geqslant 2n$, 这意味着

$$\mu = n + \lambda \cdot R \geqslant n + 2\lambda n = \frac{9n}{7}.$$

所以, 当超图的秩不超过 7 时, 实例 $\mathcal{I}$ 可在 $\mathcal{O}(1.19951^n) \leqslant \mathcal{O}(1.19951^{7\mu/9}) = \mathcal{O}^*(1.15199^\mu)$ 时间内求解.

**引理7** 步骤 (3) 的分支因子至多为 1.13863.

**证明** 设步骤 (3) 的第 $i$ 个分支中 $\mu$ 减少了 $\delta_i$, 并且令 $A_i = e_i \setminus A$ $(i \in [2])$.

在第 1 个分支中, 集合 $A$ 被删除, 且超边 $e_i$ 的秩减少至 $|A_i|$, 其中 $i \in [2]$. 因此超边 $e_i$ 权重从 1 减少至 $\min\{1, \lambda \cdot |A_i|\}$, 所以 $\mu$ 减少了

$$\delta_1 = |A| + (1 - \min\{1, \lambda \cdot |A_1|\}) - (1 - \min\{1, \lambda \cdot |A_2|\}) \geqslant |A|.$$

在第 2 个分支中, 点子集 $B = A_1 \cup A_2$ 以及超边 $e_1$ 被删除, 并且超边 $e_2$ 的秩减少至 $|A|$. 由于 $e_1$ 和 $e_2$ 的秩至少为 8, 总权重减少了 $2 - \min\{1, \lambda \cdot |A|\}$, 所以 $\mu$ 减少了

$$\delta_2 = |A_1| + |A_2| + 2 - \min\{1, \lambda \cdot |A|\} \geqslant 18 - 2|A| - \min\{1, \lambda \cdot |A|\}.$$

在步骤 (3) 中, $A$ 的大小至少为 2, 因此根据 $A$ 的大小考虑以下两种情形. 总共 7 种分支向量及其分支因子如表 2 所示.

情形 1: $2 \leqslant |A| \leqslant 6$. 此时分支向量不劣于 $(2, 14-2\lambda)$, $(3, 12-3\lambda)$, $(4, 10-4\lambda)$, $(5, 8-5\lambda)$ 或 $(6, 6-6\lambda)$, 它们的分支因子至多为 1.13278.

情形 2: $|A| \geqslant 7$, 在此情形下, 有 $\delta_1 \geqslant 7$ 并且 $\delta_2 = |A_1| + |A_2| + 1$. 若 $|A_1| = |A_2| = 1$, 那么分支向量为 $(9-2\lambda, 3)$, 对应的分支因子为 1.13863. 否则有 $|A_1| + |A_2| \geqslant 3$, 进而分支向量不劣于 $(7, 4)$, 对应的分支因子为 1.13819.

综上所述, 步骤 (3) 的分支因子至多为 1.13863.

**引理8** 步骤 (4) 的分支因子至多为 1.13391.

**证明** 由于 $N_E(v)$ 中至少存在两条超边, 设 $e_1, e_2 \in N_E(v)$. 由于执行完步骤 (3) 之后, 任意两条秩不超于 8 的超边至多包含一个公共邻居. 因此在步骤 (4) 中, 必有 $e_1 \cap e_2 = \{v\}$. 在步骤 (4) 执行完成以后, 将执行基本约简操作使得实例进一步简化为 I 类约简实例, 此时超边 $e_1$ 和 $e_2$ 以及集合 $N_V[v]$ 将被删除. 根据步骤 (4) 的执行条件, $|N_V[v]| \geqslant |e_1 \cup e_2| = 15$. 此外, 由于 $e_1$ 和 $e_2$ 秩不少于 8,

**表 3　步骤 (5) 中各种情形下的分支向量及其分支因子**

**Table 3**　Branching vectors and branching factors of step (5)

| $\lvert N_1\rvert$ | $\lvert N_2\rvert$ | Branching vector (not worse than) | Branching factor (no more than) |
|---|---|---|---|
| 4 | 4 | $(5,5)$ | 1.14870 |
| 4 | 5 | $(5-\lambda, 6+\lambda)$ | 1.13500 |
| 4 | 6 | $(5-2\lambda, 7+2\lambda)$ | 1.12459 |
| 4 | $\geqslant 7$ | $(4+4\lambda, 7+10\lambda)$ | 1.11637 |

他们的权重均为 1, 所以在步骤 (4) 第 2 个分支中, $\mu$ 至少减少了 17. 显然, 在步骤 (4) 第 1 个分支中, $\mu$ 至少减少了 1. 所以, 步骤 (4) 的分支向量不劣于 $(1,17)$, 其对应的分支因子为 1.13391.

**引理9**　步骤 (5) 的分支因子至多为 1.14870.

**证明**　在步骤 (5) 中, 超边的秩 $\mathrm{rank}(e)=|N_1|+|N_2|\geqslant 8$, 并且满足 $|N_1|=4$ 和 $|N_2|\geqslant 4$. 而在步骤 (4) 后, 任意两条秩不少于 8 的超边均无公共顶点. 所以, 对于任意顶点 $v\in e$, 存在与 $e$ 不同的超边 $e'$ 包含了 $v$, 并且有 $\mathrm{rank}(e')\leqslant 7$ 成立. 所以当 $v$ 被删除时, 超边 $e'$ 的秩减少 1, 从而 $\mu$ 至少减少了 $\lambda$.

令 $\delta_1$ 和 $\delta_2$ 表示第 1 个和第 2 个分支中度量 $\mu$ 减少的数值. 在第 1 个分支中, 点集 $N_1$ 中的 4 个顶点被删除, 超边 $e'$ 的秩减少了 $|N_1|$ 且超边 $e$ 的秩减少至 $|N_2|$. 所以 $\mu$ 减少了

$$\delta_1=|N_1|+\lambda\cdot|N_1|+1-\min\{1,\lambda\cdot|N_2|\}=5+4\lambda-\min\{1,\lambda\cdot|N_2|\}.$$

类似地, 在第 2 个分支中, 点集 $N_2$ 中的顶点被删除, 超边 $e$ 的秩减少至 $|N_1|$. 所以 $\mu$ 减少了

$$\delta_2=|N_2|+\lambda\cdot|N_2|+1-\min\{1,\lambda\cdot|N_1|\}=(1+\lambda)\cdot|N_2|+3\lambda.$$

集合 $N_2$ 的大小至少为 4, 总共 4 种分支向量及其分支因子如表 3 所示.

情形 1: $4\leqslant|N_2|\leqslant 6$. 在此情形下, 分支向量分别为 $(5,5)$, $(5-\lambda,6+\lambda)$ 以及 $(5-2\lambda,7+2\lambda)$, 它们的分支因子至多为 1.14870.

情形 2: $|N_2|\geqslant 7$. 适当放缩易得 $\delta_1\geqslant 4+4\lambda$ 和 $\delta_2\geqslant 7+10\lambda$, 此时的分支因子至多为 1.11637.

综上所述, 步骤 (5) 的分支因子至多为 1.14870.

基于上述对算法 $\mathcal{A}_{\mathrm{MISH}}$ 的时间复杂度分析, 步骤 (3)∼(5) 的分支因子均不超过 1.15199, 结合引理 6 可以得出以下结论.

**引理10**　MISH 问题可在 $\mathcal{O}^*(1.15199^{\mu})$ 时间内求解.

**定理1**　MISH 问题可在 $\mathcal{O}(1.1520^{\ell})$ 时间以及多项式空间内求解.

**证明**　根据度量 $\mu$ 的定义, 有

$$\mu=n+\sum_{r\in\mathbb{N}}\omega_r\cdot m_r\leqslant n+\sum_{r\in\mathbb{N}}m_r=n+m=\ell.$$

结合引理 10, 可知 MISH 问题可以在 $\mathcal{O}^*(1.15199^{\mu})\leqslant 1.15199^{\ell}\ell^{\mathcal{O}(1)}\leqslant\mathcal{O}(1.1520^{\ell})$ 时间内求解.

此外, 算法 $\mathcal{A}_{\mathrm{MISH}}$ 采用递归调用的形式进行计算, 递归层数不超过 $n$ 且图的存储空间为多项式, 因此算法空间复杂度为多项式.

## 6 超图上带惩罚最大独立集问题的精确算法

本节将给出 PC-MISH 问题的运行时间为 $\mathcal{O}(1.3982^{\ell})$ 和 $\mathcal{O}^*(1.9548^n)$ 的算法.

### 6.1 算法设计

G-MISH 问题在 $F=\varnothing$ 时退化为 PC-MISH 问题. 设 $\mathcal{I}=(G,F,k)$ 为 G-MISH 问题的一个实例, 其中 $F\subseteq V$ 总是成立. 本小节将给出算法求解实例 $\mathcal{I}$. 注意到, $F$ 的初始设置为 $\varnothing$. 在本算法中, 不会添加任何超边到 $F$ 中, 因此在算法递归调用中将始终保持 $F\subseteq V$ 这一性质成立, 且此性质将应用于算法中. 接下来介绍算法中的 6 条简单的约简规则.

(1) 若 $k<0$, 则直接返回 "无解".

(2) 若 $k\geqslant 0$ 且 $\mathrm{rank}(G)\leqslant 1$, 返回 $\varnothing$.

(3) 若存在超边 $e$ 有 $\mathrm{rank}(e)\leqslant 1$, 则直接移除 $e$.

(4) 若存在不属于任何超边的顶点 $v$, 则直接移除 $v$.

(5) 若顶点 $v\in V\setminus F$ 仅包含在 1 条超边 $e$ 中, 则将 $v$ 加入 $F$ 中.

(6) 若有两条秩为 2 的超边 $e_1,e_2\in E$, 它们包含的顶点完全相同, 则直接删除 $e_1$.

上述约简规则的正确性可以直接由引理 1∼4 直接得出. 若上述约简规则对于实例 $\mathcal{I}$ 均不可执行, 则称 $\mathcal{I}$ 为 II 类约简实例. 容易发现, 在 II 类约简实例中, 每个顶点至少包含在两条超边中, 并且秩为 2 的超边所包含的顶点两两不同.

接下来给出求解 PC-MISH 问题的分支搜索算法 $\mathcal{A}_{\text{PC-MISH}}$ (如算法 2 所示), 其核心思想是对某个顶点 $v$ 进行分支: 要么最优解包含 $v$; 要么将 $v$ 加入 $F$ 中, 然后依据引理 2 进行进一步分支搜索.

算法 $\mathcal{A}_{\text{PC-MISH}}$ 包括 8 个步骤, 采用递归算法的形式进行描述, 即每一步要么输出解, 要么调用算法自身.

算法共分为两部分, 第 I 部分包括步骤 (2)∼(5), 用于解决 $F$ 非空的情况. 第 II 部分包括步骤 (6)∼(8), 用于解决 $F$ 为空的情况.

算法 $\mathcal{A}_{\text{PC-MISH}}$ 第 I 部分中, 步骤 (2) 和 (5) 为约简规则, 而步骤 (3) 和 (4) 为分支规则. 它们的正确性均可由引理 1∼3 直接得到. 在算法 $\mathcal{A}_{\text{PC-MISH}}$ 第 II 部分中, 步骤 (6) 是正确的, 因为当 $n\leqslant k$ 时 $V$ 是解, 而 $m\leqslant k$ 时 $E$ 是解. 步骤 (7) 和 (8) 为分支规则. 它们的正确性可直接由引理 1, 2, 5 得到. 综上所述, 算法 $\mathcal{A}_{\text{PC-MISH}}$ 是正确的.

进一步, 通过对算法 $\mathcal{A}_{\text{PC-MISH}}$ 的简单观察, 可以得出

- 当算法执行完步骤 (2) 后, 每条超边至多包含一个 $F$ 中顶点, 且步骤 (3)∼(5) 恰有一条可执行.
- 步骤 (3)∼(5) 的执行顺序是优先处理包含了 $F$ 中顶点的超边中秩较大者.
- 当算法执行完步骤 (5) 后, $F$ 必定为空.
- 当算法执行完步骤 (7) 后, 任意超边 $e$ 的秩必然满足 $2\leqslant\mathrm{rank}(e)\leqslant 4$, 且步骤 (8) 总能执行.

### 6.2 基于度量治之的时间复杂度分析

本小节将使用度量治之技术分析算法 $\mathcal{A}_{\text{PC-MISH}}$ 的时间复杂度. 首先定义实例 $\mathcal{I}$ 的度量函数

$$\mu=\mu(\mathcal{I})=\kappa\cdot|V\setminus F|+(2-\kappa)k,$$

其中 $\kappa=0.45706$.

**算法 2** PC-MISH 问题的分支搜索算法 $\mathcal{A}_{\text{PC-MISH}}$

**输入:** 超图 $G=(V,E)$, 一个子集 $F$(满足 $F\subseteq V$) 和一个整数 $k$, 其中 $F$ 的初始值为 $\varnothing$.

**输出:** 判断是否有解, 若有解则输出一个大小不超过 $k$ 且与 $F$ 不相交的解集 $S\subseteq V\cup E$, 使得 $\text{rank}(G-S)\leqslant 1$.

判断输入实例是否为 II 类约简实例, 若非 II 类约简实例, 则依次执行 6 条基本的约简操作.

**if** $F\neq\varnothing$ **then**

若一条超边 $e$ 包含了至少两个在 $F$ 中的顶点, 则删除超边 $e$ 并加入解中, 即返回 $S=\{e\}\cup\mathcal{A}_{\text{PC-MISH}}(G-\{e\},F,k-1)$.

若存在秩至少为 4 的超边 $e$ 包含 $F$ 中顶点 $v$, 设 $e=\{v\}\cup U$, 计算

- $S_1=\{e\}\cup\mathcal{A}_{\text{PC-MISH}}(G-\{e\},F,k-1)$;
- $S_2=U\cup\mathcal{A}_{\text{PC-MISH}}(G-U,F,k-3)$,

若均无解, 则返回 "无解"; 否则返回 $S_1$ 和 $S_2$ 中的较小者.

若存在秩为 3 的超边 $e$ 包含 $F$ 中顶点 $v$, 设 $e=\{v\}\cup U$, 计算

- $S_1=\{e\}\cup\mathcal{A}_{\text{PC-MISH}}(G-\{e\},F\cup U,k-1)$;
- $S_2=U\cup\mathcal{A}_{\text{PC-MISH}}(G-U,F,k-2)$,

若均无解, 则返回 "无解"; 否则返回 $S_1$ 和 $S_2$ 中的较小者.

若存在秩为 2 的超边 $e$ 包含 $F$ 中顶点 $v$, 设 $e=\{v,u\}$, 删除 $u$ 并放入解中, 即返回 $S=\{u\}\cup\mathcal{A}_{\text{PC-MISH}}(G-u,F,k-1)$.

**else if** $F=\varnothing$ **then**

若 $m\leqslant k$, 则返回 $E$; 若 $n\leqslant k$, 则返回 $V$.

若顶点 $v$ 包含在一条秩至少为 5 的超边 $e$ 中, 将 $e$ 包含的顶点划分为 $(N_1,N_2)$ 满足 $|N_1|=2$ 且 $|N_2|\geqslant 3$, 则计算

- $S_1=\{e\}\cup\mathcal{A}_{\text{PC-MISH}}(G-\{e\},\varnothing,k-1)$;
- $S_2=N_1\cup\mathcal{A}_{\text{PC-MISH}}(G-N_1,\varnothing,k-2)$;
- $S_3=N_2\cup\mathcal{A}_{\text{PC-MISH}}(G-N_2,\varnothing,k-|N_2|)$,

若均无解, 则返回 "无解"; 否则返回 $S_1$, $S_2$, $S_3$ 中的较小者.

任选一条超边 $e$ 及其所包含的一个顶点 $v\in e$. 计算

- $S_1=\{v\}\cup\mathcal{A}_{\text{PC-MISH}}(G-v,\varnothing,k-1)$;
- $S_2=\mathcal{A}_{\text{PC-MISH}}(G,\{v\},k)$,

若均无解, 则返回 "无解"; 否则返回 $S_1$ 和 $S_2$ 中的较小者.

**end if**

下文将使用度量函数 $\mu$ 完成算法 $\mathcal{A}_{\text{MISH}}$ 的时间复杂度分析. 首先, 当 $\mu(\mathcal{I})\leqslant 0$ 时, $\mathcal{I}$ 可在多项式时间内求解. 其次, 6 条基本约简规则中, 要么删除一些顶点或超边放入解中, 要么将一些顶点放入 $F$ 中, 这两种操作都不会使得度量 $\mu$ 增加. 因此, 以下分析总假设实例 $\mathcal{I}$ 为 II 类约简实例. 此外, 算法 $\mathcal{A}_{\text{PC-MISH}}$ 的约简规则 (步骤 (2) 和 (5)) 也仅执行顶点或超边的删除操作, 亦不会使得度量 $\mu$ 增加. 后续将分析算法中各分支规则的分支因子.

**引理11** 步骤 (3) 的分支向量不劣于 $(2-\kappa,6)$, 对应的分支因子为 1.23672.

**证明** 第 1 个分支中, 超边 $e$ 放进解中, $k$ 减少 1, 集合 $V$ 和 $F$ 均无变化, 所以 $\mu$ 减少了 $2-\kappa$. 第 2 个分支中, $e\setminus\{v\}$ 包含的顶点全部放进解中. 根据执行条件, 有 $\text{rank}(e)=4$, 所以 $k$ 减少了 3, 集合 $V$ 中删去了 3 个顶点而 $F$ 无变化, 所以 $\mu$ 减少了 6. 故步骤 (3) 的分支向量不劣于 $(2-\kappa,6)$, 对应的分支因子为 1.23672.

**引理12** 步骤 (4) 的分支向量不劣于 $(2+\kappa,4)$, 对应的分支因子为 1.24496.

**证明** 第 1 个分支中, 超边 $e$ 放进解中, 点子集 $e\setminus\{v\}$ 中的两个顶点全部加入 $F$ 中. 因此, $k$ 减少 1, 集合 $V$ 无变化而 $F$ 增加 2, 所以 $\mu$ 减少了 $2+\kappa$. 第 2 个分支中, 点子集 $e\setminus\{v\}$ 包含的顶点全部放进解中. 根据执行条件, 有 $\text{rank}(e)=3$, 所以 $k$ 减少了 2, 集合 $V$ 中删去了 2 个顶点而 $F$ 无变化, 所以 $\mu$ 减少了 4. 故步骤 (3) 的分支向量不劣于 $(2+\kappa,4)$, 对应的分支因子为 1.24496.

表 4 步骤 (8) 中各种情形下的分支向量及其分支因子 a)

**Table 4** Branching vectors and branching factors of step (8) a)

| Order of the steps | Branching vector (not worse than) | Branching factor (no more than) |
|---|---|---|
| (8), (3), (3) | $(2, 6+\kappa, 8, 4-\kappa)$ | **1.39811** |
| (8), (3), (4) | $(2, 6+\kappa, 6, 4+\kappa)$ | 1.39227 |
| (8), (3), (5) | $(2, 6+\kappa, 4)$ | 1.34742 |
| (8), (4), (2) | $(2, 4+\kappa, 4+2\kappa)$ | 1.36640 |
| (8), (4), (4) | $(2, 4+\kappa, 6+2\kappa, 4+3\kappa)$ | **1.39811** |
| (8), (4), (5) | $(2, 4+\kappa, 4+2\kappa)$ | 1.36640 |
| (8), (5), (5) | $(2, 4+\kappa)$ | 1.25426 |

a) The largest branching factors are highlighted in bold.

**引理13** 步骤 (7) 的分支向量不劣于 $(2-\kappa, 4, 6)$, 对应的分支因子为 1.39404.

**证明** 根据执行条件, 可知 $|N_1| \geqslant 2$ 且 $|N_2| \geqslant 3$. 第 1 个分支中, 超边 $e$ 放进解中, $k$ 减少了 1, $V$ 和 $F$ 均无变化, 所以 $\mu$ 减少了 $2-\kappa$. 第 2 个分支中, 点集 $N_1$ 放进解中, $k$ 减少了 2, 集合 $V$ 的大小减少了 2 而 $F$ 仍为空, 所以 $\mu$ 减少了 4. 第 3 个分支中, 点集 $N_2$ 放进解中, $k$ 至少减少了 3, 集合 $V$ 中至少删去了 3 个顶点而 $F$ 仍为空, 所以 $\mu$ 减少了 6. 故步骤 (7) 的分支向量不劣于 $(2-\kappa, 4, 6)$, 对应的分支因子为 1.39404.

步骤 (8) 是整个算法的瓶颈, 需要研究在执行完此步骤后可能的步骤执行序列, 然后整体分析所有情形下的分支向量及其分支因子.

**引理14** 步骤 (8) 执行完成后, 有 $|F| \leqslant 1$. 进一步, 步骤 (8) 后, 存在一个步骤执行序列, 直至 $F$ 为空或算法停止, 算法第 I 部分的步骤将至少出现两次, 整体的分支因子至多为 1.39811.

**证明** 步骤 (8) 的第 1 个分支中, 顶点 $v$ 被删除, $k$ 减少了 1, 所以 $|F| = 0$ 且度量 $\mu$ 减少了 2. 步骤 (8) 的第 2 个分支中, 顶点 $v$ 放入 $F$ 中, 所以 $|F| = 1$ 且度量 $\mu$ 减少了 $\kappa$. 所以步骤 (8) 自身的分支向量为 $(2, \kappa)$.

接下来讨论第 2 个分支执行后算法的后续执行过程, 并分析整个子分支过程的分支向量及其分支因子. 由于 $|F| = 1$, 算法将进入第 I 部分. 算法第 I 部分包含 4 个步骤, 其中步骤 (2) 和 (5) 为约简规则, 执行后度量 $\mu$ 减少 2; 步骤 (3) 为分支规则, 其分支向量不劣于 $(6, 2-\kappa)$; 步骤 (4) 为分支规则, 其分支向量不劣于 $(4, 2+\kappa)$. 根据步骤 (3)$\sim$(5) 的执行顺序, 算法会优先处理秩更大的超边. 由于顶点 $v$ 至少包含于两条超边中, 假设算法首先处理超边 $e$, 其次处理超边 $e'$, 那么有 $2 \leqslant \mathrm{rank}(e') \leqslant \mathrm{rank}(e) \leqslant 4$. 根据超边 $\mathrm{rank}(e)$ 的值, 考虑以下 3 种情形, 总共 7 种分支向量及其分支因子如表 4 所示.

情形 1: $\mathrm{rank}(e) = 4$. 步骤 (3) 将被执行, 要么 $e \setminus \{v\}$ 包含的顶点被删除, 要么超边 $e$ 被删除. 此时, 整体的分支向量为 $(2, 6+\kappa, 2)$. 若超边 $e$ 被删除, $|F| = 1$ 依然成立. 接下来的步骤将处理 $e'$, 步骤 (3)$\sim$(5) 之一将被执行. 分支向量分别为 $(2, 6+\kappa, 8, 4-\kappa)$, $(2, 6+\kappa, 6, 4+\kappa)$, $(2, 6+\kappa, 4)$, 分支因子至多为 1.39811.

情形 2: $\mathrm{rank}(e) = 3$. 在此情形中, 步骤 (3) 不可执行, 步骤 (4) 将被执行. 此时, 要么 $e \setminus \{v\}$ 包含的顶点被删除, 要么超边 $e$ 被删除且 $e \setminus \{v\}$ 包含的顶点全部加入 $F$ 中. 所以, 整体的分支向量为 $(2, 4+\kappa, 2+2\kappa)$. 若超边 $e$ 被删除且 $e \setminus \{v\}$ 包含的顶点全部加入 $F$ 中. 一方面, $F$ 的大小可能超过 1, 步骤 (2) 可能执行. 另一方面, $F$ 非空, 那么 $2 \leqslant \mathrm{rank}(e') \leqslant \mathrm{rank}(e) \leqslant 3$, 所以步骤 (4) 或 (5) 将被执行. 分支向量分别为 $(2, 4+\kappa, 4+2\kappa)$, $(2, 4+\kappa, 6+2\kappa, 4+3\kappa)$, $(2, 4+\kappa, 4+2\kappa)$, 分支因子至多为 1.39811.

情形 3: $\mathrm{rank}(e)=2$. 在此情形中, 步骤 (3) 和 (4) 不可执行, 步骤 (5) 将被执行. 此时, $e\setminus\{v\}$ 包含的顶点将被删除. 所以, 整体的分支向量为 $(2,2+\kappa)$. 接下来的步骤将处理 $e'$, 由于 $2\leqslant\mathrm{rank}(e')\leqslant\mathrm{rank}(e)\leqslant 2$, 那么 $\mathrm{rank}(e')=2$ 成立, 所以步骤 (5) 将再次被执行. 整体的分支向量为 $(2,4+\kappa)$, 对应的分支因子不超过 1.25426.

综上所述, 存在一个步骤执行序列, 直至 $F$ 为空或算法停止, 算法第 I 部分的步骤将至少出现两次, 整体的分支因子至多为 1.39811.

在上述对算法 $\mathcal{A}_{\text{PC-MISH}}$ 的时间复杂度分析中, 分支因子至多为 1.39811, 故可得出引理 15.

**引理15** PC-MISH 问题可以在 $\mathcal{O}^*(1.39811^{\mu})$ 时间内求解.

**定理2** PC-MISH 问题可在 $\mathcal{O}(1.3982^{\ell})$ 和 $\mathcal{O}^*(1.9548^{n})$ 时间以及多项式空间内求解.

**证明** 由于 PC-MISH 问题的初始实例满足 $F=\varnothing$, 所以要么步骤 (6) 直接返回解, 要么一定满足关系 $m>k$ 且 $n>k$. 所以可以推出

$$\mu=\kappa\cdot|V\setminus F|+(2-\kappa)k\leqslant\kappa n+(2-\kappa)n=2n,$$

并且

$$\mu=\kappa\cdot|V\setminus F|+(2-\kappa)k=\kappa n+2(1-\kappa)k+\kappa k\leqslant\kappa n+(1-\kappa)(n+m)+\kappa m=\ell.$$

结合引理 15, PC-MISH 问题可在 $\mathcal{O}^*(1.39811^{\mu})\leqslant\mathcal{O}^*(1.39811^{2n})\leqslant\mathcal{O}^*(1.9548^{n})$ 和 $\mathcal{O}^*(1.39811^{\mu})\leqslant 1.39811^{\ell}\ell^{\mathcal{O}(1)}\leqslant\mathcal{O}(1.3982^{\ell})$ 的时间内求解.

此外, 算法 $\mathcal{A}_{\text{PC-MISH}}$ 采用递归调用的形式进行计算, 递归层数不超过 $n$ 且图的存储空间为多项式, 因此算法的空间复杂度为多项式.

## 7 超图上最大独立集问题与分裂图上最小子集反馈集问题的关系

若一个无向图可以将点集划分为 $(I,K)$, 其中 $I$ 为独立集, $K$ 为团, 则称其为分裂图. 在分裂图上 SFVS 问题中, 给定一个分裂图 $G=(V,E)$ 以及关键点集合 $T\subseteq V$, 找到最小的点子集 $S$ (称为子集反馈集), 使得删掉 $S$ 之后任何关键点都不在环中. 该问题存在两个不同版本, 限制版和非限制版. 在限制版问题中, 解集 $S$ 不允许包含关键点; 而在非限制版问题中, 解集 $S$ 可以包含任何顶点. 本节将给出分裂图上两个版本的 SFVS 问题与 MISH 和 PC-MISH 问题之间的联系. 本节的论证将使用如下性质 [27]: 一个分裂图没有经过关键点的环当且仅当没有经过关键点的三角形 (长度为 3 的环).

**定理3** 对于任意实数 $\alpha>1$, MISH 问题能在 $\mathcal{O}^*(\alpha^{\ell})$ 时间内求解当且仅当 $n$ 阶分裂图上限制版 SFVS 问题能在 $\mathcal{O}^*(\alpha^{n})$ 时间内求解.

**证明** 首先证明, 若分裂图上的限制版 SFVS 问题能在 $\mathcal{O}^*(\alpha^{n})$ 时间内求解, 则 MISH 问题也能在 $\mathcal{O}^*(\alpha^{\ell})$ 时间内求解.

设 $\mathcal{I}=(G=(V,E),F=E,k)$ 为 MISH 问题的一个实例, 其中 $|V|=n$ 且 $|E|=m$. 现构造分裂图上限制版 SFVS 问题的实例 $\mathcal{I}'=(G'=(V',E'),T',k')$ 如下: 对于点集 $V$ 中的每个顶点 $v$, 均构造一个顶点 $v'$, 得到集合 $K'=\{v'\mid v\in V\}$; 对于超边集 $E$ 中每一条超边 $e$, 均构造一个顶点 $t'_e$, 得到集合 $I'=\{t'_e\mid e\in E\}$. 进而得到点集

$$V'=K'\cup I'=\{v'\mid v\in V\}\cup\{t'_e\mid e\in E\}.$$

令 $K'$ 为团, $I'$ 为独立集, 且对于任意顶点 $v' \in K'$ 和 $t'_e \in I'$, 满足 $v'$ 与 $t'_e$ 相连当且仅当 $v \in e$, 即

$$E' = \{v'u' \mid v', u' \in K'\} \cup \{v't'_e \mid v \in e\}.$$

根据构造, $G'$ 是 $\ell = n + m$ 阶分裂图, 并且 $(I', K')$ 是一个分裂划分. 最后令 $T' = I'$ 且 $k' = k$.

现调用分裂图上的限制版 SFVS 问题的算法求解, 得到子集反馈集 $S' \subseteq V'$, 运行时间为 $\mathcal{O}^*(\alpha^\ell)$. 由于 $S'$ 不包含关键点, 令 $S = \{v \in V \mid v' \in S'\}$. 根据子集反馈集的定义, 在子图 $G' - S'$ 中, 每个关键点至多连接一个顶点, 否则此关键点仍在环中. 这说明了在 $G - S$ 中每一条超边的秩至多为 1, 又由于 $|S| = |S'| \leqslant k$ 所以 $S$ 是 $\mathcal{I}$ 的解. 所以, 可以在 $\mathcal{O}^*(\alpha^\ell)$ 时间内求解 MISH 问题.

然后证明, 若 MISH 问题也能在 $\mathcal{O}^*(\alpha^\ell)$ 时间内求解, 则 $n$ 阶分裂图上的限制版 SFVS 问题也能在 $\mathcal{O}^*(\alpha^n)$ 时间内求解.

设 $\mathcal{I} = (G = (V, E), T, k)$ 为分裂图上限制版 SFVS 问题的实例, 其中 $|V| = n$. 不妨设 $T$ 是独立集且 $V \setminus T$ 是团. 事实上, 若 $T$ 中有 3 个两两相连的关键点, 则 $\mathcal{I}$ 一定无解; 而若 $T$ 中有 2 个相连的关键点, 与它们相连的非关键点必然在子集反馈集中, 可以直接删去并令 $k$ 减少 1, 此后这两个关键点之一必不会出现在任何三角形中, 可以直接从图中移除. 进而分裂图 $G$ 存在分裂划分 $(I, K)$, 其中 $I$ 为独立集且 $T \subseteq I$, 而 $K$ 为团. 又由于 $I$ 中的非关键点不在任何经过关键点的三角形中, 可以直接从图中移除. 所以令 $I = T$ 且 $K = V \setminus T$.

构造 MISH 问题的实例 $\mathcal{I}' = (G' = (V', E'), F' = E', k')$ 的实例如下: 超图 $G' = (V', E')$ 满足 $V' = \{v' \mid v \in V\}$ 以及 $E' = \{e'_w \mid w \in I\}$, 并且超边 $e'_v$ 包含顶点 $v'$ 当且仅当在 $w$ 和 $v$ 在 $G$ 中构成一条边. 这里有 $\ell = |V'| + |E'| = |V| = n$, 令 $k' = k$. 此时, $\mathcal{I}$ 有大小不超过 $k$ 的子集反馈集当且仅当 $\mathcal{I}'$ 有大小不超过 $k'$ 的解, 也即 $\mathcal{I}'$ 有大小至少为 $\ell - k'$ 的独立集. 所以, 可以直接调用 MISH 问题的算法求解, 运行时间为 $\mathcal{O}^*(\alpha^\ell) = \mathcal{O}^*(\alpha^n)$.

结合定理 1 和 3, 可得以下推论.

**推论1** $n$ 阶分裂图上限制版 SFVS 问题可在 $\mathcal{O}(1.1520^n)$ 时间以及多项式空间内求解.

**定理4** 对于任意实数 $\alpha > 1$, PC-MISH 问题能在 $\mathcal{O}^*(\alpha^\ell)$ 时间内求解当且仅当 $n$ 阶分裂图上非限制版 SFVS 问题能在 $\mathcal{O}^*(\alpha^n)$ 时间内求解.

**证明** 首先证明, 若分裂图上的非限制版 SFVS 问题能在 $\mathcal{O}^*(\alpha^n)$ 时间内求解, 则 PC-MISH 也能在 $\mathcal{O}^*(\alpha^\ell)$ 时间内求解.

设 $\mathcal{I} = (G = (V, E), F = E, k)$ 为 PC-MISH 问题的实例, 其中 $|V| = n$ 且 $|E| = m$. 现构造分裂图上非限制版 SFVS 问题的实例 $\mathcal{I}' = (G' = (V', E'), T', k')$ 如下: 对于点集 $V$ 中的每个顶点 $v$, 均构造一个顶点 $v'$, 得到集合 $K' = \{v' \mid v \in V\}$; 对于超边集 $E$ 中任意超边 $e$, 也构造一个顶点 $w'_e$, 得到集合 $I' = \{w'_e \mid e \in E\}$. 进而得到点集

$$V' = K' \cup I' = \{v' \mid v \in V\} \cup \{w'_e \mid e \in E\}.$$

令 $K'$ 为团, $I'$ 为独立集, 且对于任意顶点 $v' \in K'$ 和 $w'_e \in I'$, 满足 $v'$ 与 $w'_e$ 相连当且仅当 $v \in e$, 即

$$E' = \{v'u' \mid v', u' \in K'\} \cup \{v'w'_e \mid v \in e\}.$$

根据构造, $G'$ 是 $\ell = n + m$ 阶分裂图, 并且 $(I', K')$ 是 $G'$ 的一个分裂划分. 最后令 $T' = I'$ 且 $k' = k$.

现调用分裂图上的非限制版 SFVS 问题的算法求解得到子集反馈集 $S' \subseteq V'$, 运行时间为 $\mathcal{O}^*(\alpha^\ell)$. 令 $S = \{v \in V \mid v' \in S'\} \cup \{e \in E \mid w'_e \in S'\}$. 根据子集反馈集的定义, 在子图 $G' - S'$ 中, 每个关键点

至多连接一个顶点, 否则此关键点仍在环中. 这说明了在 $G-S$ 中每一条超边至多包含一个顶点, 又由于 $|S|=|S'|\leqslant k$ 所以 $S$ 是 $\mathcal{I}$ 的解. 所以, 可以在 $\mathcal{O}^*(\alpha^\ell)$ 时间内求解 PC-MISH 问题.

然后证明, 若 PC-MISH 也能在 $\mathcal{O}^*(\alpha^\ell)$ 时间内求解, 则分裂图上的限制版 SFVS 问题能在 $\mathcal{O}^*(\alpha^n)$ 时间内求解.

设 $\mathcal{I}=(G,T,k)$ 为分裂图上非限制版 SFVS 问题的一个实例, 其中 $|V|=n$ 且 $|E|=m$. 注意到 $\alpha$ 是常数, 存在充分大的正整数 $C$, 使得分支向量 $(1,C,C)$ 的分支因子不超过 $\alpha$. 首先在多项式时间内找到 $G$ 的分裂划分 $(I,K)$ [30], 其中 $I$ 为独立集且而 $K$ 为团. 若 $K$ 的大小不超过 $2C+1$, 则穷举 $K$ 中与子集反馈集的交集, 然后可在多项式时间内求得实例 $\mathcal{I}$ 的子集反馈集. 否则, 假设 $|K|\geqslant 2C+1$. 接下来考虑两种情形.

情形 1: $K$ 中存在关键点 $t$. 此时将 $K\setminus\{t\}$ 划分为两部分 $(K_1,K_2)$, 使得 $K_1$ 和 $K_2$ 都至少包含 $C$ 个顶点. 注意到, 对于任意解 $S$, 要么 $t\in S$, 要么 $K_1\subseteq S$, 要么 $K_2\subseteq S$, 否则存在顶点 $x_1\in K_1\setminus S$ 和 $x_2\in K_2\setminus S$ 使得 $tx_1x_2$ 构成三角形. 所以执行如下分支规则:

- 若 $K_1\cap F$ 非空, 则令 $S_1$ 无解, 否则计算 $(G-K_1,T\setminus K_1,F,k-|K_1|)$ 的解 $S_1$;
- 若 $K_2\cap F$ 非空, 则令 $S_2$ 无解, 否则计算 $(G-K_2,T\setminus K_2,F,k-|K_2|)$ 的解 $S_2$;
- 若 $t\in F$, 则令 $S_3$ 无解, 否则计算 $(G-t,T\setminus\{t\},F,k-1)$ 的解 $S_3$.

若 $S_1$, $S_2$, $S_3$ 均无解, 则返回无解, 否则返回较小者. 此分支规则的正确性可由引理 5 直接得到, 其分支向量子不劣于 $(1,C,C)$, 所以其分支因子不高于 $\alpha$.

情形 2: $K$ 中无关键点. 在此情形中, $I$ 中的非关键点 $v$ 不在任何经过关键点的三角形中, 因此可以直接将 $v$ 移除, 得到 $(G-v,T,F\setminus\{v\},k)$.

现在, 假设 $T$ 是独立集且 $V\setminus T$ 是团. 构造 PC-MISH 问题的实例 $\mathcal{I}'=(G'=(V',E'),F'=\varnothing,k')$ 的实例如下: 超图 $G'=(V',E')$ 满足 $V'=\{v'\mid v\in V\}$ 以及 $E'=\{e'_w\mid w\in I\}$, 并且超边 $e'_v$ 包含顶点 $v'$ 当且仅当在 $w$ 和 $v$ 与 $G$ 中构成一条边. 这里有 $\ell=|V'|+|E'|=|V|=n$, 令 $k'=k$. 若 $\mathcal{I}$ 有大小不超过 $k$ 的子集反馈集当且仅当 $\mathcal{I}'$ 有大小不超过 $k'$ 的解, 也即 $\mathcal{I}'$ 有价值至少为 $\ell-k'$ 的松弛独立集. 所以, 可以直接调用 MISH 问题的算法求解, 运行时间为 $\mathcal{O}^*(\alpha^\ell)=\mathcal{O}^*(\alpha^n)$.

结合定理 2 和 4, 可得以下推论.

**推论2** $n$ 阶分裂图上非限制版 SFVS 问题可在 $\mathcal{O}(1.3982^n)$ 时间和多项式空间内求解.

## 8 总结

一般无向图上的最大独立集问题是精确算法中研究最为火热也最为重要的问题之一, 然而超图上的最大独立集问题目前研究相对较少. 本文系统地研究超图上的最大独立集问题, 同时考虑超图中带惩罚这一概念下的变体问题, 给出了这两个问题在 $n$ 和 $\ell$ 两个参数下的多个改进精确算法. 特别地, 本文给出的 PC-MISH 问题以 $n$ 为参数的精确算法突破了 $\mathcal{O}(2^n)$ 的时间复杂度瓶颈.

## 参考文献


1 Fomin F V, Kratsch D. Exact Exponential Algorithms. Berlin: Springer, 2010

2 Cygan M, Fomin F V, Kowalik L, et al. Parameterized Algorithms. Berlin: Springer, 2015

3 Bellman R. Dynamic programming treatment of the travelling salesman problem. J ACM, 1962, 9: 61–63

4 Iwata Y. A faster algorithm for dominating set analyzed by the potential method. In: Proceedings of International Symposium on Parameterized and Exact Computation, 2011. 41–54

5 Chitnis R, Fomin F V, Lokshtanov D, et al. Faster exact algorithms for some terminal set problems. J Comput Syst Sci, 2017, 88: 195–207

6 Iwata Y, Kobayashi Y. Improved analysis of highest-degree branching for feedback vertex set. Algorithmica, 2021, 83: 2503–2520

7 Iwata Y, Wahlström M, Yoshida Y. Half-integrality, LP-branching, and FPT algorithms. SIAM J Comput, 2016, 45: 1377–1411

8 Iwata Y, Yamaguchi Y, Yoshida Y. 0/1/all CSPs, half-integral $A$-path packing, and linear-time FPT algorithms. In: Proceedings of the 59th IEEE Annual Symposium on Foundations of Computer Science (FOCS), 2018. 462–473

9 Xiao M, Nagamochi H. Exact algorithms for maximum independent set. Inf Comput, 2017, 255: 126–146

10 Feo T A, Resende M G C, Smith S H. A greedy randomized adaptive search procedure for maximum independent set. Oper Res, 1994, 42: 860–878

11 Puthal D, Nepal S, Paris C, et al. Efficient algorithms for social network coverage and reach. In: Proceedings of IEEE International Congress on Big Data, 2015. 467–474

12 Araujo F, Farinha J, Domingues P, et al. A maximum independent set approach for collusion detection in voting pools. J Parallel Distrib Comput, 2011, 71: 1356–1366

13 Sander P V, Nehab D, Chlamtac E, et al. Efficient traversal of mesh edges using adjacency primitives. ACM Trans Graph, 2008, 27: 1–9

14 van Bevern R, Mnich M, Niedermeier R, et al. Interval scheduling and colorful independent sets. J Sched, 2015, 18: 449–469

15 Blauth J, Nägele M. An improved approximation guarantee for prize-collecting TSP. In: Proceedings of the 55th Annual ACM Symposium on Theory of Computing, 2023. 1848–1861

16 Pedrosa L L C, Rosado H K K. A 2-approximation for the k-prize-collecting steiner tree problem. Algorithmica, 2022, 84: 3522–3558

17 Fukunaga T. Spider covers for prize-collecting network activation problem. In: Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms, 2015. 9–24

18 Tarjan R E, Trojanowski A E. Finding a maximum independent set. SIAM J Comput, 1977, 6: 537–546

19 Jian T. An $\mathcal{O}(2^{0.304n})$ algorithm for solving maximum independent set problem. IEEE Trans Comput, 1986, C-35: 847–851

20 Robson J M. Algorithms for maximum independent sets. J Algorithms, 1986, 7: 425–440

21 Shi L, Cai X. An exact fast algorithm for minimum hitting set. In: Proceedings of the 3rd International Joint Conference on Computational Science and Optimization, 2010. 64–67

22 Bai T, Xiao M Y. Exact and parameterized algorithms for restricted subset feedback vertex set in chordal graphs. In: Proceedings of International Conference on Theory and Applications of Models of Computation, 2022. 249–261

23 Zhou X Q, Xiao M Y. Exact algorithms for the subset feedback vertex set problem in undirected graphs. Chinese J Comput, 2018, 41: 493–505 [周晓清, 肖鸣宇. 无向图中子集反馈顶点集问题的精确算法. 计算机学报, 2018, 41: 493–505]

24 Fomin F V, Heggernes P, Kratsch D, et al. Enumerating minimal subset feedback vertex sets. Algorithmica, 2014, 69: 216–231

25 Fomin F V, Gaspers S, Lokshtanov D, et al. Exact algorithms via monotone local search. J ACM, 2019, 66: 1–23

26 Fomin F V, Le T N, Lokshtanov D, et al. Subquadratic kernels for implicit 3-hitting set and 3-set packing problems. ACM Trans Algorithms, 2019, 15: 1–44

27 Philip G, Rajan V, Saurabh S, et al. Subset feedback vertex set in chordal and split graphs. Algorithmica, 2019, 81: 3586–3629

28 Alon N. Covering graphs by the minimum number of equivalence relations. Combinatorica, 1986, 6: 201–206

29 Fomin F V, Grandoni F, Kratsch D. A measure & conquer approach for the analysis of exact algorithms. J ACM, 2009, 56: 1–32

30 Stephane F, Hammer P. Split graphs. In: Proceedings of the 8th South-east Combinatorics, Graph Theory, and Computing, 1977. 311–315

# Exact algorithms for maximum independent set problem on hypergraphs

Tian BAI & Mingyu XIAO*

*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu* 611731*, China*

* Corresponding author. E-mail: myxiao@uestc.edu.cn

**Abstract** The maximum independent set problem is one of the most fundamental and significant NP-complete problems in computer science. This paper studies exact algorithms for the maximum independent set problem on hypergraphs (MISH) and the prize-collecting maximum independent set problem on hypergraphs (PC-MISH). Given a hypergraph, MISH aims to find a maximum independent set, where an independent set in the hypergraph is a subset of vertices such that no two vertices are contained in the same hyperedge. The PC-MISH problem is a relaxation of the MISH problem. In this problem, it is allowed to find a non-independent set $X$ that violates the independence constraints on some hyperedges, that is, these hyperedges contain more than one vertices. The prize of the subset $X$ is defined as the number of vertices minus the number of hyperedges on which $X$ violates the independence constraint. In PC-MISH, we are asked to find a subset of vertices with the maximum prize. This paper studies the exact algorithms for both MISH and PC-MISH parameterized by two parameters $n$ and $\ell = n + m$, where $n$ is the number of vertices and $m$ is the number of hyperedges. Using the exact algorithm for the maximum independent set problem in undirected graphs, an $\mathcal{O}^*(1.1996^n)$-time for MISH can be directly obtained. In this paper, we show that PC-MISH can be solved in $\mathcal{O}^*(1.9548^n)$ time, breaking the $2^n$-barrier. Furthermore, this paper proposes an $\mathcal{O}(1.1520^\ell)$-time algorithm for MISH and an $\mathcal{O}(1.3982^\ell)$-time algorithm for PC-MISH. These two results improve the previous time bound $\mathcal{O}(1.1550^\ell)$ and $1.5^\ell 2^{o(\ell)}$ for the MISH and PC-MISH problems, respectively.

**Keywords** exact algorithms, maximum independent set problem, prize-collecting maximum independent set problem, hypergraphs, minimum subset feedback vertex set problem