arXiv:2601.05044v1 [cs.DS] 8 Jan 2026

诚邀阅读《NP完全问题的精细粒度复杂性》

Jesper Nederlof

2026年1月9日

摘要

假设 P 不等于 NP，那么任何用于求解 NP 完全问题的算法，其最坏情况运行时间都必然是超多项式的。但我们能达到的最快运行时间是多少？在着手探讨这个问题之前，还有一个更具挑衅性的问题摆在面前：既然对于许多问题，朴素的蛮力基准算法至今仍是最快的，那么它们的运行时间或许已经是最优的?

我们在本综述中称之为“NP 完全问题的细粒度复杂性”的领域，正是研究这个问题。我们诚邀读者在一场引人入胜的领域巡礼中，重温一些经典成果，并深入了解令人振奋的最新进展；巡礼将途经代数、复杂性理论、极值组合学与加法组合学、密码学，当然还有最重要的算法设计。

1 引言

算法设计的一个自然终极目标，是获得具有最优最坏情况运行时间的算法。更准确地说，目标是

1. 对于某个固定计算问题，算法能在时间 $T(s(x))$ 内求解每个实例 x；其中 T 表示所设计算法的最坏情况运行时间，它根据实例的选定规模度量 $s(x)$ 来表示 x，以及

2. 证明这一最坏情况运行时间构成了一道根本性的屏障，即任何函数 T' 的增长速度若显著低于 T，就不存在能实现运行时间 $T'(s(x))$ 的算法（例如 $T'(s(x)) = O(T(s(x))^{0.99})$）。

用现代术语来说，这一主题可称为细粒度复杂性。然而，在当前文献中，这一术语主要用于多项式时间算法。

假设 P 不等于 NP，那么对于每个 NP完全问题，最优最坏情况运行时间 $T(s(x))$ 关于位规模 $|x|$ 呈超多项式增长。遵循 Cobham–Edmonds论题，研究人员通常致力于避免此类超多项式运行时间。参数化复杂性 通过引入一种规模度量 $s(x)$ 来实现这一点，该度量可以远小于 $|x|$。在这种情况下，参数化复杂性的核心问题是，能否将超指数行为限制为仅依赖于 $s(x)$，以及能否设计出运行时间为$T(s(x))|x|^{O(1)}$的算法。

在本综述中，我们将 $s(x)$的选择限定为输入的规范参数，例如图的顶点数。由于这类规模函数 $s(x)$与

编码 x 所需的比特数 $|x|$ 呈多项式关系，因此参数化复杂性中的许多典型问题都变得平凡。不过，我们转而研究 NP 完全问题的细粒度复杂性，并力求找到最优的最坏情况运行时间 $T(s(x))$，不会因 T 可能是超多项式（事实上，很可能甚至是指数级）的前景而气馁！

人们很容易根据经验论据和奥卡姆剃刀，认定这类最优最坏情况运行时间是由非常自然且极其简单的算法所实现的：对于许多经典的 NP 完全问题，尽管数十年来一直有人研究，甚至早于 NP 完全性的定义，这些简单算法在本质上仍然拥有最快的最坏情况运行时间。例如，

- Karl Menger 早在 20 世纪 30 年代就提出过这样一个问题：[Sch05b] 对于旅行商问题，能否改进那个平凡的 $n!$时间算法——它只是尝试所有往返路线？20 世纪 60 年代，Bellman [Bel62], Held 和 Karp [HK62], 用一种简单的 $2^n n^2$ 时间动态规划方法给出了肯定的答案；但这本质上仍是该问题目前已知最快的最坏情况运行时间。

- 20 世纪 50 年代和 60 年代，几位俄罗斯科学家在一系列论文中研究了能否改进求解 NP 困难问题的朴素基准算法（俄语称为“perebor”，意为“暴力搜索”或“穷举搜索”），这些问题包括 k-合取范式可满足性问题（关于 k-合取范式可满足性问题的定义，见第 3 节）。Yablonski 也未能抵挡上述诱惑，甚至（错误地）声称自己证明了无法避免暴力搜索方法。详情请见综述 [Tra84]。

经过数十年的研究，研究人员认识到，认为简单的基线算法已达到最优最坏情况运行时间这一壁垒的天真假设与事实相去甚远：对于许多经典 NP 完全问题，例如哈密顿回路、 k-着色、装箱问题和子集和，研究已经取得了许多令人振奋的进展，动摇了（甚至在某些情况下推翻了）早先关于暴力搜索无法改进的看法。与此同时，对于某些计算问题，尤其是 k-合取范式可满足性问题，这个问题正迅速变得愈发重要：强指数时间假设 （SETH，见假设 3.1）大致指出，求解 k-合取范式可满足性问题时，暴力搜索无法避免；如今，这已成为广受认可的假设，并常被用作证据，表明算法应当具有最优的最坏情况运行时间。

精确指数时间算法。 上世纪，尽管上述问题已针对许多 NP 完全问题分别得到研究，但将 NP 完全问题作为一个整体，研究其精确最坏情况运行时间的领域，直到本世纪初才在几篇颇具影响力的综述推动下逐渐兴起。这些综述列出了一系列令人振奋的难题，由 Woeginger 撰写 [Woe01, Woe04, Woe08]。此后数年，该领域蓬勃发展，并以“（中等／精确）指数时间算法”为名；一系列专门探讨这一主题的达格斯图尔研讨会 [HPSW13, FIK08, HKPS11] 相继举行；教科书《精确指数算法》 [FK10]以及两篇综述文章 [FK13, KW16] 发表于《美国计算机协会通讯》。另请参阅以 k-合取范式可满足性问题为主题的综述 [Sch05a] 。

由于“精确指数时间”算法这一名称并不能真正将其与参数化复杂性区分开来（即，针对 NP 完全问题的参数化算法通常都是精确指数时间算法），而且我们认为所研究的核心问题也确实与“细粒度复杂性”这一较为年轻的领域相联系（尽管后者似乎仅限于多项式时间算法），因此在本综述中，我们使用了另一个术语来指代理论计算机科学中的这一子领域：“NP完全问题的精细粒度复杂性”。

针对 3-着色 的预备算法。 为了说明 NP 完全问题的精细粒度复杂性领域所研究的问题类型，我们考察以下问题：

k-着色**输入：** 一个无向图 $G = (V,\ E)$。**问题：** 是否存在一个函数 $c\colon V \to [k]$，使得对于每个 $\{v, w\} \in E$，都有 $c(v) \neq c(w)$?

固定 $k = 3$，并尝试为 3-着色设计一个快速算法。朴素基准算法会遍历 c 的所有 3^n 个候选项，因此运行时间为 $O^*(3^n)$；稍微聪明一些的算法则会遍历 $X \subseteq V$，并检查 X 是否为独立集，以及 $G[V \setminus X]$ 是否为二分图：很容易看出，当且仅当所求函数存在时，这样的 X 才存在，而检查二分性可以在多项式时间内轻松完成。事实上，采用这种策略时，我们甚至可以将枚举范围限制为大小至多为 $n/3$ 的集合 X，从而得到一个运行时间为 $O^*(\binom{n}{n/3}) = O^*(1.89^n)$ 的算法。

现在我们介绍一种更巧妙的算法（最初由 [BE05] 提出）：

定理 1.1。 *存在一种随机化算法，可在 1.5^n 时间内求解 3-着色，并在解存在时以至少 $1 - 1/e$ 的概率输出一个解。*

证明。 考虑 3-着色问题的一个基于列表的变体：对于每个顶点 $v \in V$，我们给定一个列表 $L(v) \subseteq [3]$，并要寻找一个赋值 $c\colon V \to [k]$，使得 $c(v) \in L(v)$ 且 $c(v) \neq c(w)$ 对每个 $\{v, w\} \in E$ 都成立。容易证明，如果对每个 $v \in V(G)$，都有 $|L(v)| \leq 2$，那么该问题可以在多项式时间内求解，例如使用简单的传播算法，或将其归约为 2-CNF 可满足性问题（已知该问题也可以在多项式时间内求解）。

现在考虑以下算法：对于每个顶点 v，均匀且独立地随机选取 $L(v) \in \binom{[3]}{2}$，并在多项式时间内求解由此得到的基于列表的变体。如果它检测到一个函数 c，那么显然该算法是正确的。对于另一个方向，注意到

$$\Pr_L[\forall_{v \in V} c(v) \in L(v)] \geq (2/3)^n,$$

由于所有列表都是独立重新采样的。此外，如果 $c(v) \in L(v)$对于所有 $v \in V$都成立，那么基于列表的变体就有一个满足所需性质的解。因此，如果我们运行该多项式时间算法 1.5^n 次，那么当解 c 存在时，我们未能输出**是**的概率至多为

$$(1 - (2/3)^n)^{1.5^n} \leq 1/e,$$

利用标准不等式 $1 + x \leq e^x$ □

本综述（不）讨论什么。本综述旨在引导理论计算机科学领域的研究人员了解 NP 完全问题的细粒度复杂性。我们希望传达的是，这是一个优美的领域，蕴含着精妙的思想，并与理论计算机科学和数学的其他领域有着诸多联系。为此，我们概述了一些精选成果的证明概要，其中既包括反映该领域令人振奋且仍在发展的特征的近期成果，也包括一些过于核心、优雅而不容略过的早期成果。不过，本综述绝不声称面面俱到。由于已有许多其他优秀的综述或教科书讨论相关内容，我们省略了一些非常重要的突破和研究方向。我们将在第7节列举其中一些。本综述以及整个领域的一个重要侧重点，是其定性特征：一般来说，相较于未能突破自然屏障的成果，我们认为那些能将运行时间缩短至突破自然屏障的成果更有趣。

2 记号

对于计算问题的实例 x 或算法的输入 x，我们使用 $O^*()$ 记号来省略关于 x 编码长度的多项式因子；整数采用二进制编码。

如果 b 是一个布尔值，我们令 $[b]$ 在 $b =$ **真** 时表示数字 1，否则表示数字 0。另一方面，如果 i 是一个整数，那么我们令 $[i]$ 表示集合 $\{1,\ldots,i\}$。

对于集合 S 和整数 i，我们令 2^S 表示 S 的幂集，并令 $\binom{S}{i}$ 表示 S 中基数恰好为 i 的子集族。利用斯特林近似，可以证明

$$\frac{n^n}{d^d(n-d)^{n-d}n^{O(1)}} \leq \binom{n}{d} \leq \frac{n^n}{d^d(n-d)^{n-d}}n^{O(1)}. \tag{1}$$

如果 $d = \alpha n$，那么除去关于 n 的多项式因子后，这等于

$$\frac{n^n}{d^d(n-d)^{n-d}} = \frac{n^n}{(\alpha n)^{\alpha n}((1-\alpha)n)^{(1-\alpha)n}} = \left(\alpha^{-\alpha}(1-\alpha)^{-(1-\alpha)}\right)^n = 2^{h(\alpha)n},$$

其中 $h = -\alpha \lg \alpha - (1-\alpha)\lg(1-\alpha)$ 是二元熵函数。通过初等微积分可以证明

$$h(p) \leq p\lg(4/p). \tag{2}$$

If f 是一个函数，其定义域为 S，并且我们令 $X \subseteq S$ 表示 $f(X) = \{f(x)\colon\ x \in X\}$.

矩阵和向量用粗体表示。若**a**、**b** 是两个维度相同的向量d，我们令 $\langle a、\ b\rangle := \sum_{i=1}^{d} \mathbf{a}[i]\mathbf{b}[i]$ 表示它们的内积。若**M** 是一个矩阵，其行由R 索引、列由 C索引，并且 $X \subseteq R$, $Y \subseteq C$ 我们令$\mathbf{M}[X,Y]$ 表示由来自 X 的行和来自 Y的列构成的**M** 的子矩阵。我们还使用 $\mathbf{M}[X,\cdot]$ 或$\mathbf{M}[\cdot,Y]$ 表示不限制行/列。

我们用 $i \equiv_p j$ 表示 i 等于 j 模 p；若上下文清楚，则省略 p。

3 合取范式的可满足性

研究最为广泛的 NP完全问题之一，是判定合取范式（CNF）中的布尔公式是否满足。回顾一下，这类公式是子句的合取，而子句是文字的析取，其中一个文字要么是变量，要么是其否定。若一个 CNF 中的布尔公式的所有子句都至多包含 k 个文字，则称其为 k-CNF。

k-合取范式可满足性问题**输入：** k-合取范式公式 φ，包含 n 个变量和 m个子句。**问题：** 是否存在一个赋值，使 n个变量满足 φ?

现在可以将关于 NP 完全问题的粗粒度/细粒度复杂性中可能最著名的假设表述如下:

假设 3.1 （指数时间假设，[IP01]）。存在一个 $\delta > 0$，使得没有算法能够在 O（$2^{\delta n}$）时间内解决 3-合取范式可满足性问题。

假设 3.2 (强指数时间假设，[IP01])。对于每个 $\varepsilon > 0$，都存在一个 k，使得没有任何算法能在 $O^*((2-\varepsilon)^n)$ 时间内解决 k-合取范式可满足性问题。

我们用“细粒度复杂性”指研究能否将 $t(n)$ 时间界改进为 $t(n)^{1-\Omega(1)}$ 时间界；类似地，也可以用“较粗粒度复杂性”指研究能否将 $t(n)$ 时间界改进为 $t(n)^{o(1)}$ 时间界。需要指出的是，假设假设 3.1 成立，那么对于许多问题（以标准规模度量为参数化），我们已经得到了在较粗粒度意义下最优的算法和下界。本综述研究的所有问题都属于此列。例如，已知本综述中任何问题的一个 $2^{o(n)}$ 时间算法（其中 n也按本综述中的定义）都足以推翻指数时间假设。这类推论通常源于标准的 NP 完全性归约和稀疏化引理，我们将在下文详细讨论（引理 3.2）。例如，参见 [CFK+15，第 14] 章。

3.1 k-合取范式可满足性问题算法

算法 Monien-Speckenmeyer k-SAT 算法(φ)　　φ 是一个包含 n 个变量的 k-合取范式

输出： φ 是否可满足

1: **如果**存在未满足的子句 $l_1 \vee l_2 \vee \ldots \vee l_{k'}$ **则**

2:　**对于** $i = 1, \ldots, k'$ **执行**

3:　　$\rho \leftarrow$ 通过设置 $\neg l_1, \neg l_2, \ldots, \neg l_{i-1}, l_i$ 得到的限制

4:　　**若** Monien-Speckenmeyer k-SAT 算法($\varphi|\rho$)**则返回真**

5:　**返回假**

6: **返回真**

算法 1： Monien 和 Speckenmeyer 算法，用于求解 k-合取范式可满足性问题。

关于 k-合取范式可满足性问题最坏情况复杂性的文献非常丰富。有几种不同的算法能够在 $O^*(2^{(1-1/O(k))n})$ 时间内求解 k-合取范式可满足性问题，但颇为奇怪的是，目前尚不清楚能否将其改进为运行时间为 $O^*(2^{(1-1/o(k))n})$ 的算法。人们投入了大量精力，试图减小运行时间 $O^*(2^{(1-1/O(k))n})$ 中大 O 项的常数（无论 k 是常数还是 k 非常数）。本文不关注此类改进，详情请参阅当前最先进的成果 [HKZZ19、Sch24] 。

我们首先介绍几种求解 k-合取范式可满足性问题的简单算法。第一种算法比第二种和第三种慢（后两种算法在大 O 表示法中忽略常数因子后达到了当前最先进水平）。

3.1.1 Monien 和 Speckenmeyer 算法

Monien 和 Speckenmeyer 的算法 [MS85] 是最早提出一种算法的工作，它相较于用于 k-合取范式可满足性问题的平凡 $O^*(2^n)$ 时间算法，实现了（适度的）改进。该算法概述于算法 1 中。它使用限制这一概念，即一个函数 $\rho\colon [n] \to \{0, 1, *\}$，它将变量设为 0，1，或不对其赋值（对应于将其设为 $*$）。

关键步骤位于第 3 行：我们定义一个限制，使所有出现在前 i 个文字中的变量都被赋值，并以这种方式确保子句的前 $i-1$ 个文字不满足，而第 i 个文字得到满足。随后，算法继续确定公式$\varphi|\rho$ 是否可满足；该公式是通过移除所有被 ρ 满足的子句以及由 ρ 赋值的所有变量而得到的公式（后者可能导致出现空子句，进而得到一个不可满足的公式）。如果存在满足 φ 的赋值$\mathbf{x}$，那么它将在第 2 行循环的第 i次迭代中被检测到，其中 i 满足条件：$\mathbf{x}$ 满足 l_i，但不满足 $l_1, \ldots, l_{i-1}$ 如果 $T[n]$ 是该算法执行的递归调用次数，

则有 $T[1] = 1$和 $T[n] \leq \sum_{i=1}^{k} T[\max\{n-i, 0\}]$。因此，如果定义$T'[1] = 1$和 $T'[n] = \sum_{i=1}^{k} T'[\max\{n-i, 0\}]$，那么对于所有 n，都有 $T[n] \leq T'[n]$。

这些数 $T'[1]$、$T'[2]$、.....称为斐波那契 k步数。可以通过对 n 进行归纳，或借助组合方法[1] 证明 $T'[n] = 2^{(1-2^{-O(k)})n}$。

3.1.2 Schöning 算法

Schöning 提出了一种快得多的随机化[2] 算法，如算法 2 所示。一个关键部分是子程序 localSearch(φ, $\mathbf{x}$ d)，它能在 $O^*(k^d)$ 时间内判断 φ 是否存在一个与$\mathbf{x}$的汉明距离至多为 d 的满足赋值。这个子程序是在算法 1 的基础上稍作改动，并加入以下简单观察得到的：如果存在一个赋值$\mathbf{y}$ 满足 φ，但$\mathbf{x}$ 不满足 φ，原因是某个子句 $l_1 \vee l_2 \vee \ldots l_p$ 未被$\mathbf{x}$满足，那么 $\mathbf{x}$ 和$\mathbf{y}$ 在这些文字所涉及的 p 个变量中至少有一个取值不同。因此，我们可以针对 $i = 1$、.....、p 递归，并翻转$\mathbf{x}$ 中 l_i 所对应变量的取值；可以假设，在某次递归调用中，我们得到了一个更接近$\mathbf{y}$ 的赋值，从而减小距离参数。因此，localSearch(φ, $\mathbf{x}$ d) 至多调用 $p \leq k$ 次距离参数为 $d-1$的直接递归调用，因而触发的（间接）递归调用总数至多为 k^d。由于每次递归调用都需要多项式时间，因此可得出时间复杂度为 $O^*(k^d)$ 的结论。

算法 2 现在只需随机选取$\mathbf{x}$ ，并希望当存在解$\mathbf{y}$ 时， $d(\mathbf{x}, \mathbf{y})$（即$\mathbf{x}$ 与$\mathbf{y}$ 之间的汉明距离）至多为 d。这种情况发生的概率至少为 $\sum_{i=0}^{d} \binom{n}{i}/2^n \geq \binom{n}{d}/2^n$。因此，经过 $2^n/\binom{n}{d}$ 次迭代后，返回**真** 的概率至少为

$$1 - \left(1 - \binom{n}{d}/2^n\right)^{\lceil 2^n/\binom{n}{d}\rceil} \geq 1 - 1/e \geq 1/2,$$

其中，在第一个不等式中我们使用了 $1 + x \leq e^x$ 。

算法 SchoningkSAT(φ)　　φ 是一个含有 k 个文字、 n 个变量的合取范式
输出： **真** 的概率至少为 1 2 ；如果 φ 可满足，否则输出**假**
1: $d = n/(k+1)$ 2: **对于** $i = 1 \ldots \lceil 2^n/\binom{n}{d}\rceil$
执行3: 均匀随机选取$\mathbf{x} \in \{0,1\}^n$ 4: **如果**
localSearch(φ, $\mathbf{x}$ d) **则返回 真**5: **返回假**

算法 2: 用于 k-合取范式可满足性问题的 Schöning 算法。

[1]利用 $T'[n]$ 至多等于满足以下条件的子集数量这一事实：对每个 $i \in [n-k]$，都有 $X \cap \{i, \ldots, i+k\} \neq \emptyset$。将除至多 b 个元素外的所有 $[n]$ 划分为 n/b 个块，每块包含 $b = \Theta(2^k)$ 个连续元素 B_1，.....， $B_{n/b}$ 并说明，对于某个常数 $c > 1$， $B_i \cap X$ 的选项数量至多为 $2^b/c$。[2]DGH+02] 中介绍了一种基于覆盖码的去随机化方法。

现在我们利用 (1) 得到，运行时间 $O^*(k^d 2^n / \binom{n}{d})$ 变为

$$\begin{aligned} O^*\left(k^d 2^n d^d (n-d)^{n-d} n^{-n}\right) &= O^*\left(2^n (kd)^d (n-d)^{n-d} n^{-n}\right) \\ &= O^*\left(2^n \left(n\frac{k}{k+1}\right)^d \left(n\frac{k}{k+1}\right)^{n-d} n^{-n}\right) \\ &= O^*\left(2^n \left(\frac{k}{k+1}\right)^n\right) = O^*\left(2^n \left(1-\frac{1}{k+1}\right)^n\right) \\ &= O^*\left(2^{(1-1/O(k))n}\right), \end{aligned}$$

其中最后一行使用了 $1 + x \leq e^x$ 。

3.1.3 基于随机限制的算法

我们给出另一个运行时间为 $O^*(2^{(1-1/O(k))n})$ 的算法，该算法基于随机限制和切换引理（通常归因于 [Hås89], , 但该引理建立在许多以类似思想开展的先前工作之上）。这里介绍的算法还可以扩展为判定有界深度电路的可满足性问题，详见 [IMP11]。[3]

和之前一样，令 φ 为一个 k-合取范式。将出现在 φ 中的变量称为 v_1、……、v_n，并将子句称为 C_1、……、C_m。

切换引理界定了随机限制的决策树深度。对于我们的应用，需要精确表述，因此使用如下（稍显冗长的）定义：

定义 3.3。 合取范式 φ 的规范决策树 通过递归地将单个根顶点 r 扩展为一棵树来定义，具体如下：

- 如果 φ 没有子句，则该树只有一个顶点 r ，称为1-叶节点，
- 如果 φ 有一个空子句，则该树只有一个顶点 r ，称为0-叶 f,
- 否则，将 r 的层级 定义为0 ，其对应的限制 定义为将n 设为星号的限制。令 i 为使 C_i 尚未满足的最小整数，并令其变量为v'_1、……、v'_p。对于 $j = 0$、……、$p-1$，为每个层级为 j 且对应限制为 ρ 的顶点添加两个子节点，层级为 $j+1$ ，对应的限制则是将 ρ 中的 v'_i 设为**真** 或**假**后得到的限制。对层级为 p 、对应限制为 φ 的每个顶点 r' 递归继续构造，并将得到的规范决策树的根与 r'合并。

我们用 $CDTD(\varphi)$表示 φ的规范决策树的深度。不难看出，该树可在线性于其大小、且乘以输入大小的多项式因子的时间内构造；由于它是二叉树，其大小至多为 $2^{CDTD(\varphi)}$。因此，我们可以在 $O^*(2^{CDTD(\varphi)})$时间内检查该树是否有一个1-叶节点，从而确定 φ 是否可满足。

现在我们陈述切换引理。文献中流传着许多种表述。我们的表述基于 [Bea94]。

[3]我们采用的是 Valentine Kabanets 在讲义中介绍的一个版本（链接）。

引理 3.4（切换引理，[Hås89]）。如果 φ 是一个 k-合取范式公式，且 ρ 是一个带有 pn 个星号的随机限制，[4] 那么

$$\Pr_{\rho}[CDTD(\varphi_{|\rho}) \geq d] \leq (7kp)^d.$$

算法 `SwitchkSAT`(φ) — φ 是一个含有 n 个变量的 k-合取范式

输出： φ 是否可满足，否则输出**假**

1: $p \leftarrow 1/(30k)$
2: 随机均匀地选取 $S \in \binom{[n]}{pn}$
3: **对于** 每个仅将星号赋给 $\{v_i$ 的限制 ρ： $i \in S\}$**执行**
4: 构造$\varphi_{|\rho}$的规范决策树
5: **如果** 遇到一个1-叶节点**则返回 真**
6: **返回假**

算法3： 基于切换引理的 k-合取范式可满足性问题算法（摘自 [IMP11]).

根据上述讨论以及对任意非负整数随机变量均成立的等式 $\mathbb{E}[X] = \sum_{i=0}^{\infty} \Pr[X \geq i]$，我们可知，第4行和第5行的期望运行时间为

$$\begin{aligned}
\mathbb{E}_{\rho}[2^{|CDTD(\varphi_{|\rho})|}] &= \sum_{i=0}^{\infty} \Pr_{\rho}[2^{|CDTD(\varphi_{|\rho})|} \geq i] \\
&= \sum_{i=0}^{\infty} \Pr_{\rho}[|CDTD(\varphi_{|\rho})| \geq \lg i] \\
&\quad \text{Using Lemma 3.4} \\
&\leq \sum_{i=0}^{\infty} (7kp)^{\lg i} \leq \sum_{i=0}^{\infty} \left(\tfrac{1}{4}\right)^{\lg i} = \sum_{i=0}^{\infty} 1/i^2 = O(1),
\end{aligned}$$

其中，最后一步使用了 p-级数的标准收敛性质（可通过用积分逼近该级数来证明）。现在，由于第3行考虑的限制数量至多为$2^{n-|S|}$，由期望的线性性可知，算法3的期望运行时间为
$O^*\left(2^{(1-p)n}\right) = O^*\left(2^{(1-\frac{1}{O(k)})n}\right)$.

3.2 稀疏化引理

关于 k-合取范式可满足性问题的细粒度难度，最具影响力的引理之一或许就是稀疏化引理。粗略地说，它表明，对于某个函数 f，一般的 k-合取范式可满足性问题几乎与 k-合取范式可满足性问题一样难；后者的子句数至多为 $f(k)n$，也就是说，如果将 k 视为常数，那么子句数与变量数呈线性关系。

陈述。与提出稀疏化引理的原始论文一样，为方便起见，我们用集合系统的击中集来表述它。若 $\mathcal{F} \subseteq 2^U$ 和 $X \subseteq U$，则称 X **击中** $\mathcal{F}$，如果 X与 $\mathcal{F}$中的每个集合都有交集。

[4] 具有恰好 t 个星号的版本以及每个变量具有独立星号概率 p 的版本，常常在文献中被提及。当前这个版本对我们来说更方便。

引理 3.5 ([IPZ98，CIP06])。存在一种算法，给定 $k \in \mathbb{N}$、$\varepsilon > 0$ 以及由大小至多为 k的集合构成的集合族 $\mathcal{F} \subseteq 2^U$，可在 $O^*(\ell)$时间内生成集合大小至多为 k 的集合系统 $\mathcal{F}_1$、……、$\mathcal{F}_\ell \subseteq 2^U$，使得

1. 每个子集 $X \subseteq U$ 命中 $\mathcal{F}$，当且仅当对于某个 i，X 命中 $\mathcal{F}_i$，
2. 对于每个 $i = 1$，……，ℓ，U 中的每个元素至多属于 $d = \left(\frac{4k^2}{\frac{\lg(1/\varepsilon)}{\varepsilon}}\right)^{k-1}$ 个 $\mathcal{F}_i$集合，
3. ℓ至多为 $2^{2\varepsilon n}$。

在引理 3.5 的原始版本中，第 2 项。中对 k 的依赖是双指数级的；在 [CIP06] 中，这一依赖被降为了关于 [的单指数级。尽管已知这方面存在一些下界 [SS12]，能否进一步改进这种依赖仍是一个有趣的问题。

图的情形（$k = 2$）。稀疏化引理的一个前身是 [JS99] 中的一个引理，该引理指出，稀疏图上的顶点覆盖大致上与一般图上的顶点覆盖同样困难（该论文实际上讨论的是独立集问题，但这与顶点覆盖等价；相比之下，顶点覆盖更自然地推广到超图）。

如果 $k = 2$，那么 $\mathcal{F}$ 可以看作一个图，而 X 恰好命中 $\mathcal{F}$，当且仅当它是该图的一个顶点覆盖；因此我们处于前面提到的 [JS99] 的情形。在这种情形下，证明思路很简单：如果引理3.5的条件2不成立，并且存在一个度至少为 $d = \lg(1/\varepsilon)/\varepsilon$ 的顶点，那么我们就对是否将该顶点 v 纳入顶点覆盖进行分支。我们递归处理两个子问题：在其中一个子问题中，将 v 纳入解中（顶点数减少1），并在递归调用中将其移除；在另一个子问题中，将 v 的所有邻居纳入解中，并在递归调用中移除 v 及其所有邻居（顶点数至少减少 d）。穷尽所有可能后，我们至多生成 $T[n]$ 个子问题，其中

$$T[n] \leq T[n-1] + T[n-d],$$

并且可以证明 $T[n] \leq \binom{n}{n/d} \leq 2^{h(\varepsilon/\lg(1/\varepsilon))n} \leq 2^{2\varepsilon n}$，其中最后一步用到了 (1)。

与 k-CNF-SAT的关系。引理 3.5 通常以合取范式公式的形式表述，但要约化到这里陈述的版本很简单。由于我们的引理版本并非直接针对合取范式公式，我们简要说明引理 3.5 最有用的推论之一：概述如何利用引理 3.5 证明，$2^{o(n+m)}$ 时间算法能够推翻假设 3.1。假设存在满足假设 3.1 条件的 $\delta > 0$，并令 φ为一个包含 n个变量的 3-CNF。应用引理 3.5，其中令 U 为所有 $2n$ 文字（每个变量对应一个正文字和一个负文字），令集合为 $\varepsilon = \delta/5$，并令$\mathcal{F}$ 包含针对 φ 每个子句的一个集合，该集合由该子句的文字组成（这些文字属于 U）。

我们得到 $2^{4\delta n/5}$ 个集合系统 $\mathcal{F}_1, \ldots, \mathcal{F}_\ell$，使得每个子集 $X \subseteq U$ 都命中 $\mathcal{F}$，当且仅当对于某个 i，X命中 $\mathcal{F}_i$。接着，我们将每个集合族 $\mathcal{F}_i$ 转换为一个 k-合取范式，其中每个集合对应一个子句，子句中的文字就是该集合的所有元素。若一个子集 $X \subseteq U$ 有效，则称其恰好包含 v_i 和 $\neg v_i$ 中每个 i所对应的 1 个文字。对所有有效子集应用引理 3.5 的条件 1，可知 φ 可满足，当且仅当某个 φ_i 可满足。由于每个 φ_i 中的子句数为 $O(n)$，利用假设中的次指数时间算法，我们可以在总计 $2^{4\delta n/5}2^{o(n+m)} \leq O(2^{5\delta/6})$ 时间内确定每个 φ_i（进而也确定 φ）的可满足性，这与假设 3.1 矛盾。

虽然这解决了较粗粒度归约的问题（以假设 3.1 为条件），但引理 3.5 也经常被用作更细粒度归约（以假设 3.2 为条件）的第一步，例如来自 [ABHS22、CDL+16] 的归约。

3.2.1 算法

在介绍算法之前，我们需要以下定义：

定义 3.6 (花与花瓣)。 为简洁起见，我们将大小为 s 的集合称为 s-集合。 s-花 是由 s-集合 $S_1, \dots, S_z$ 组成的集合，满足花心 $H := \cap_{i=1}^{z} S_i$ 非空。集合 $S_1 \setminus H, \dots, S_z \setminus H$ 称为花瓣；数量 $|S_i \setminus H|$ （它在 s-花中与 i 无关）称为花瓣大小。

引理 3.5 中的算法见算法 4。常数 $\theta_1, \dots, \theta_k$ 将在后文定义。若花 $S_1, \dots, S_z$ 的花瓣大小为 p，则称其为好 花，当且仅当 $z \geq \theta_p$。我们用 $\pi(\mathcal{F})$ 表示 $\mathcal{F}$ 中按包含关系极小的集合所构成的集合。

算法 约简$(\mathcal{F})$ 假设 $\mathcal{F}$ 中的集合彼此不包含（$\mathcal{F} = \pi(\mathcal{F})$）
输出： 引理 3.5 所述的一组集合系统。
1: **对于** $s = 2, \dots, k$ **do**
2: **对于** $p = 1, \dots, s-1$ **执行**
检查是否存在一个好的 s-花，并在存在时以其为分支
3: **如果** 存在一个 s-花 $S_1, \dots, S_z$，其花瓣大小为 p 且 $z \geq \theta_p$ **那么**
4: $H \leftarrow \cap_{i=1}^{z} S_i$
5: $\mathcal{F}_{heart} \leftarrow \pi(\mathcal{F} \cup \{H\})$
6: $\mathcal{F}_{petals} \leftarrow \pi(\mathcal{F} \cup \{S_i \setminus H : i = 1, \dots, z\})$.
根据我们命中花心还是全部 花瓣进行分支
7: **返回** 约简$(\mathcal{F}_{heart})$ ∪约简$(\mathcal{F}_{petals})$。
8: **返回** $\{\mathcal{F}\}$。

算法 4： 实现引理 3.5 的算法。

注意， π 保留了击中集族，因此每个 $X \subseteq U$ 命中 $\mathcal{F}$，当且仅当它命中 $\mathcal{F}_{heart}$ （若 $H \cap X \neq \emptyset$）或 $\mathcal{F}_{petals}$ （若 $H \cap X = \emptyset$）；由此，引理 3.5 的第 1 项确实为真。

观察 3.7. 如果 $\mathcal{F}$ 没有好 j-花，则每个 h-集合至多包含在 $\theta_{j-h} - 1$ 个大小为 j 的集合中。

要看出这一点为真，注意，如果某个 h-集合 H 包含在至少 θ_{j-h} 个大小为 j 的集合中，那么这些 θ_{j-h} 集合将构成一个以 H 为花心的 j-花；由于花瓣大小为 $j-h$，它将是好花。

利用 $h = 1$，我们可知，如果 $\mathcal{F}$ 是由 reduce 输出的，那么 U 的每个元素至多属于 θ_{k-1} 个 $\mathcal{F}$ 集合，因为在这种情况下，对每个 $j \leq k$，它都没有好 j-花。因此，根据观察 3.7，只要 $\theta_{k-1} \leq d$，引理 3.5 的第 2 项就成立。

3.2.2 界定运行时间和输出大小。

原始证明（以及教科书 [FG06] 中的证明）都包含若干关于子句在算法 4 的递归路径上多次被添加和移除的论断。我们采用一种（可以说）更直接的方法。[5]

我们通过为集合系统 $\mathcal{F}$赋予势函数 φ 来界定 reduce$(\mathcal{F})$ 的运行时间，该势函数表示向稀疏化推进的进度。势函数 φ 定义如下：

[5]将这篇综述送审后，作者得知，在 SOSA'26 [LSX26] 会议上发表了另一份关于稀疏化引理的更直接证明。

定义 3.8。令 α 为稍后设定的某个参数，$\beta_j = (4\alpha k)^{j-1}$ 和 $\theta_j = \alpha\beta_j$。定义 $\sigma(\mathcal{F})$为满足以下条件的最大 $s < k$：对于每个 $j \le s$，每个 h-集合至多属于 $2\theta_{j-h}$ 个大小为 j 的集合。定义$\varphi(\mathcal{F}) = \sum_{j=1}^{\sigma(\mathcal{F})} \beta_{k-j+1}|\mathcal{F}_j|$，其中 $\mathcal{F}_j$表示 $\mathcal{F}$的所有 j-集合。

显然，$\varphi(\mathcal{F})$非负。接下来，我们将证明每次递归调用中 φ 都会大幅增加，并在约简终止时为 φ 给出上界。对于上界部分，注意到

$$\begin{aligned}
\varphi(\mathcal{F}) &= \sum_{j=1}^{\sigma(\mathcal{F})} \beta_{k-j+1}|\mathcal{F}_j| \\
&\le \sum_{j=1}^{\sigma(\mathcal{F})} \beta_{k-j+1}(n2\theta_{j-1}) && \text{each element is in at most } 2\theta_{j-1} \text{ sets of size } j \\
&\le k\beta_{k-j+1}\alpha 2\beta_{j-1}n && \theta_j = \alpha\beta_j \\
&\le k\alpha 2\beta_{k-1}n && \beta_j = (4\alpha k)^{j-1} \\
&\le \beta_k n.
\end{aligned} \tag{3}$$

现在我们考虑一次特定的约简递归调用，并为势函数的增加量给出下界。由于 $\varphi(\mathcal{F})$涉及求和至 $\sigma(\mathcal{F})$，因此首先证明 $\sigma(\mathcal{F})$永不减小会很有帮助。直观而言，可以将 σ 视为算法的阶段：算法已经处理了大小至多为$\sigma(\mathcal{F})$的集合，并保证这些集合具有良好性质，即它们不会过度集中。

引理 3.9. $\sigma(\mathcal{F}) \le \min\{\sigma(\mathcal{F}_{heart})\ \sigma(\mathcal{F}_{petals})\}$。

证明。为导出矛盾，假设 $j = \min\{\sigma(\mathcal{F}_{heart})\ \sigma(\mathcal{F}_{petals})\} + 1 \le \sigma(\mathcal{F})$。根据 σ 的定义，这意味着必须加入一个大小为 j 的集合，即所选的 s-花的花心大小为 j 或花瓣大小为 j，因此在 $\mathcal{F}_{heart}$ 或 $\mathcal{F}_{petals}$ 中，某个 h-集合属于至少 $2\theta_{j-h}$个大小为 j的集合。由于 $j < s$，$\mathcal{F}$ 不包含良好的 j-花，因此 h-集合在 $\mathcal{F}$中至多包含于 $\theta_{j-h} - 1$ 个大小为 j 的集合中。因此，在 $\mathcal{F}_{heart}$ 或 $\mathcal{F}_{petals}$中，某个固定的 h-集合 X 新增了超过 θ_{j-h} 个大小为j 的超集。对于 $\mathcal{F}_{heart}$，只新增了 H，所以这是不可能的。对于 $\mathcal{F}_{petals}$，这意味着 X 包含于花瓣 $S_1 \setminus H$，……，$S_z \setminus H$中的至少 $l = \theta_{j-h}$ 个中；不失一般性，这些花瓣为 $S_1 \setminus H$，……，$S_l \setminus H$。但这样一来，集合 S_1，……，S_l也是一个 s-花，其花心为$H \cup X$。这与 S_1，……，S_z被选为 S_1矛盾；S_l的花瓣大小为 $j - h$，因此由于 $l = \theta_{j-h}$，它是良好的。而且由于它的花瓣大小更小，算法本应优先选择它。

□

根据引理 3.9，分析 φ的增加时无需担心 σ，因为 σ 只会增加（这只会使 φ 增加）。接下来，我们通过比较 φ（$\mathcal{F}_{heart}$）与 φ（$\mathcal{F}$）的增加量，给出其下界。注意，在 $\mathcal{F}_{heart}$ 中会添加一个 h-集合；由于花心 H 的加入，最多会移除 $2\theta_{j-h}$ 个大小为 $j \le \sigma$（$\mathcal{F}$）的集合，这是因为 H 至多属于 $2\theta_{j-h}$ 个大小为 $j \le \sigma$（$\mathcal{F}$）的集合（注意，我们无需考虑满足 $j > \sigma$（$\mathcal{F}$）的 j-集合，因为它们并不

会对 φ（$\mathcal{F}$）作出贡献）。因此，我们有

$$\begin{aligned}\varphi(\mathcal{F}_{heart}) &\geq \varphi(\mathcal{F}) + \beta_{k-h+1} - \sum_{j=h+1}^{\sigma(\mathcal{F})} 2\theta_{j-h}\beta_{k-j+1} \\ &\geq \varphi(\mathcal{F}) + \beta_{k-h+1} - 2\alpha k\beta_{k-h} \qquad \theta_j = \alpha\beta_j \text{ and } \beta_j = (4\alpha k)^{j-1} \\ &> \varphi(\mathcal{F}).\end{aligned} \tag{4}$$

类似地，比较 $\mathcal{F}_{petals}$ 与 $\mathcal{F}$，注意到大小为 p 的 z 个花瓣已被添加，并且每个这样的集合至多包含于 $2\theta_{j-p}$ 个大小为 j的集合中。因此，我们得到

$$\begin{aligned}\varphi(\mathcal{F}_{petals}) &\geq \varphi(\mathcal{F}) + z\left(\beta_{k-p+1} - \sum_{j=p+1}^{\sigma(\mathcal{F})} 2\theta_{j-p}\beta_{k-j+1}\right) \\ &\qquad \theta_j = \alpha\beta_j \text{ and } \beta_j = (4\alpha k)^{j-1} \\ &\geq \varphi(\mathcal{F}) + z(\beta_{k-p+1} - 2\alpha k\beta_{k-p}) \\ &\qquad S_1, \ldots, S_z \text{ is a good } s\text{-flower with petal size } p \\ &\geq \varphi(\mathcal{F}) + \theta_p\beta_{k-p+1}/2 \\ &\geq \varphi(\mathcal{F}) + \alpha\beta_p\beta_{k-p+1}/2 \\ &\geq \varphi(\mathcal{F}) + \tfrac{1}{2}\alpha\beta_k.\end{aligned} \tag{5}$$

定义 $T(m)$ 为当 $m = \varphi(\mathcal{F})$ 时，reduce($\mathcal{F}$) 的递归调用次数。由 (3) 可知，$T(m) = 0$ （当 $m \geq \beta_k n$ 时），而在其他情况下，由 (4) 和 (5) 可知

$$T(m) \leq T(m+1) + T(m + \tfrac{1}{2}\alpha\beta_k).$$

由于 $\beta_k n/(\frac{1}{2}\alpha\beta_k) = 2n/\alpha$，这意味着 $T(0) \leq \sum_{i=0}^{2n/\alpha} \binom{\beta_k n}{i} \leq \binom{2n}{\alpha}\binom{\beta_k n}{2n/\alpha}$。利用不等式 $\binom{n}{k} \leq 2^{h(k/n)n}$，将其用二元熵函数 $h(p) = p\lg\frac{1}{p} + (1-p)\log\frac{1}{1-p}$ 表示，我们得到 $T(0) \leq \binom{2n}{\alpha}2^{\lambda n}$，其中

$$\begin{aligned}\lambda &\leq h\left(\frac{2}{\alpha\beta_k}\right)\beta_k \\ &\leq \frac{2}{\alpha}\lg(2\alpha\beta_k) \qquad \text{Using } h(p) \leq p\lg(4/p) \text{ (1)} \\ &\leq \frac{4(k-1)\lg(8\alpha k)}{\alpha} \qquad \text{Using } \beta_k = (4\alpha k)^{k-1}\end{aligned} \tag{6}$$

利用 $\alpha \approx k\lg(1/\varepsilon)/\varepsilon$ 可得 $\lambda \leq 2\varepsilon$ （当 ε 足够小时），因此我们可以选取 d 为

$$\theta_{k-1} = \alpha(4\alpha k^2)^{k-2} \leq (4\alpha k^2)^{k-1} = \left(\frac{4k^2\lg(1/\varepsilon)}{\varepsilon}\right)^{k-1}.$$

引理 3.5 的证明至此完成。

4 集合覆盖及其特殊情形

近年来，人们在以下著名 NP 完全问题的变体上取得了许多进展：

集合覆盖
输入： 一个全集 U、一个包含 $|U| = n$ 的集合系统 $\mathcal{S} \subseteq 2^U$，以及一个整数 k。**问题：** 是否存在集合 S_1、.....、$S_k \in \mathcal{S}$，使得 $\cup_{i=1}^{k} S_i = U$?

集合划分
输入： 一个全集 U、一个包含 $|U| = n$ 的集合系统 $\mathcal{S} \subseteq 2^U$，以及一个整数 k。**问题：** 是否存在两两不相交的集合 S_1、.....、$S_k \in \mathcal{S}$，使得 $\cup_{i=1}^{k} S_i = U$?

4.1 使用 Yates 算法和容斥原理在 $2^n n^{O(1)}$ 时间内求解集合覆盖

我们将用粗体表示矩阵或向量变量。对于整数 s，我们用 $\mathbf{I}_s$表示维度为 s 的单位矩阵。固定一个域 $\mathbb{F}$。给定矩阵 $\mathbf{A} \in \mathbb{F}^{R \times C}, \mathbf{B} \in \mathbb{F}^{R_2 \times C_2}$，我们将 克罗内克积 $\mathbf{A} \otimes \mathbf{B}$ 定义为一个矩阵，其行和列分别由 $R \times R_2$ 和 $C \times C_2$ 索引，元素为 $(\mathbf{A} \otimes \mathbf{B})[(r, r_2), (c, c_2)] = \mathbf{A}[r, c] \cdot \mathbf{B}[r_2, c_2]$。克罗内克积与普通矩阵乘积满足分配律，这一事实通常称为混合乘积性质：

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}.$$

第 n个克罗内克幂 $\mathbf{A}^{\otimes n}$ 是 n个$\mathbf{A}$相乘的乘积，因此该矩阵的行由 R^n 索引，列由 C^n 索引，其元素定义为

$$\mathbf{A}^{\otimes n}[r_1, \ldots, r_n, c_1, \ldots, c_n] = \prod_{i=1}^{n} \mathbf{A}[r_i, c_i].$$

以下引理说明了克罗内克幂对我们很有用：

引理 4.1 （Yates 算法 [Yat37]）。 设 $\mathbf{A} \in \mathbb{F}^{R \times C}$, n 是整数，且$\mathbf{v} \in \mathbb{F}^{C^n}$ 作为输入给定。那么，可以在 $O(\max\{|R|^{n+2}, |C|^{n+2}\})$ 时间内计算$\mathbf{A}^{\otimes n}\mathbf{v}$ 。

证明。设 $r = |R|$和 $c = |C|$。通过一个简单的快速傅里叶变换风格的过程即可证明该引理。我们采用 [Kas18，第 3 节1]中的讲述。注意，根据混合乘积性质可得$\mathbf{A}^{\otimes n} = \mathbf{A}^{[n-1]}\mathbf{A}^{[n-2]} \cdots \mathbf{A}^{[0]}$, 其中

$$\mathbf{A}^{[\ell]} = \underbrace{\mathbf{I}_r \otimes \mathbf{I}_r \otimes \ldots \otimes \mathbf{I}_r}_{n-\ell-1} \otimes \mathbf{A} \otimes \underbrace{\mathbf{I}_c \otimes \mathbf{I}_c \otimes \ldots \otimes \mathbf{I}_c}_{\ell}. \tag{7}$$

现在，朴素地计算$\mathbf{A}^{\otimes n}\mathbf{v}$ 的算法会对以下表达式进行求值：

$$\mathbf{A}^{[n-1]} \left(\mathbf{A}^{[n-1]} \left(\ldots \mathbf{A}^{[1]} \left(\mathbf{A}^{[0]} \mathbf{v} \right) \right) \right).$$

运行时间界可由以下观察得出：乘以$\mathbf{A}^{[\ell]}$ withavectoris 所需的算术运算次数至多为 $O(r^{n-\ell}c^{\ell+1})$。 □

我们考虑由幂集 2^U （按字典序排列）索引的向量，以及矩阵

$$\mathbf{Z} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \qquad \text{and} \qquad \mathbf{M} = \begin{pmatrix} 1 & 0 \\ \text{-1} & 1 \end{pmatrix}$$

将发挥关键作用。特别是，若$\mathbf{v}$ 以 2^U 为索引，则$\mathbf{Z}^{\otimes n}\mathbf{v}$ 就是所谓的*zeta* 变换：对于任意 Y，它满足$(\mathbf{Z}^{\otimes n}\mathbf{v})[Y] = \sum_{X\subseteq Y}\mathbf{v}[X]$ 。类似地，*Möbius* 变换$(\mathbf{M}^{\otimes n}\mathbf{v})[Y] = \sum_{X\subseteq Y}(-1)^{|Y\setminus X|}\mathbf{v}[X]$，对于任意 Y均成立。由于$\mathbf{MZ} = \mathbf{I}_2$，根据混合乘积性质，$\mathbf{M}^{\otimes k}$ 和$\mathbf{Z}^{\otimes k}$ 互为逆矩阵，对每个 k均如此。[6]

定理 4.2 ([BHK09])。 集合覆盖可以在时间 $2^n n^{O(1)}$内求解。

证明。令$\mathbf{v}$ 为 $\mathcal{S}$ 的指示向量（即，若满足条件则$\mathbf{v}[X] = 1$，否则$\mathbf{v}[X] = 0$）　　.

1. 计算$\mathbf{z} = \mathbf{Z}^{\otimes n}\mathbf{v}$

2. 计算向量 $\mathbf{z}'$，其定义为 $\mathbf{z}'[X] = (\mathbf{z}[X])^k$

3. 计算 $\mathbf{M}^{\otimes n}\mathbf{z}'$，当且仅当 $(\mathbf{M}^{\otimes n}\mathbf{z}')[U] > 0$ 时输出 **真** 。

在步骤 1 和步骤 3 中使用引理 4.1，可得到一个 $2^n n^{O(1)}$ 时间算法。为说明其正确性，我们定义另一个向量 $\mathbf{s}$，其索引由 2^U 给出如下。对于每个 $X \subseteq U$，令

$$\mathbf{s}[X] = \left|\left\{(S_1,\ldots,S_k) \in \mathcal{S}^k : \bigcup_{i=1}^{k} S_i = X\right\}\right|,$$

并注意到，集合覆盖的实例是是实例，当且仅当 $\mathbf{s}[U] > 0$。观察到，对于任意 $X \subseteq U$，都有

$$\begin{aligned}\mathbf{z}'[X] &= \left|\left\{(S_1,\ldots,S_k) \in \mathcal{S}^k : S_i \subseteq X \text{ for all } i \in \{1,\ldots,k\}\right\}\right| \\ &= \left|\left\{(S_1,\ldots,S_k) \in \mathcal{S}^k : \bigcup_{i=1}^{k} S_i \subseteq X\right\}\right| = (\mathbf{Z}^{\otimes n}\mathbf{s})[X].\end{aligned}$$

因此，$(\mathbf{M}^{\otimes n}\mathbf{z}')[U] = (\mathbf{M}^{\otimes n}\mathbf{Z}^{\otimes n}\mathbf{z}')[U] = \mathbf{s}[U]$，正确性得证。 □

4.2 避免在整个子集格上进行计算

对于集合族 $\mathcal{F}$，我们记作 $\downarrow\mathcal{F} := \{X\colon X \subseteq F, F \in \mathcal{F}\}$ 以及 $\uparrow\mathcal{F} := \{X\colon F \subseteq X, F \in \mathcal{F}\}$。通常将 $\downarrow\mathcal{F}$ 称为 $\mathcal{F}$ 的向下闭包，将 $\uparrow\mathcal{F}$ 称为 $\mathcal{F}$的向上闭包。对于向量 $\mathbf{v}$，我们令 $\mathsf{supp}(\mathbf{v})$ 表示 $\mathbf{v}$ 的指标集合，其中 $\mathbf{v}$ 在这些位置上的值非零。重新审视定理 4.2 的证明，尤其是引理 4.1，可以观察到，由于 $\mathbf{Z}$ 和 $\mathbf{M}$ 都是下三角矩阵，因此在 $\mathbf{A} = \mathbf{Z}$ 或 $\mathbf{A} = \mathbf{M}$ 的特殊情形下，式（7）中的 $\uparrow\mathsf{supp}(\mathbf{A}^{[\ell]}\mathbf{v}) \subseteq \uparrow\mathsf{supp}(\mathbf{v})$。因此，将引理 4.1 证明中的算法限制为仅计算由 $\uparrow\mathsf{supp}(\mathbf{v})$ 索引的条目，便可直接得到一个算法，在时间 $|\uparrow\mathsf{supp}(\mathbf{v})|n^{O(1)}$内计算 $\mathbf{M}^{\otimes n}\mathbf{v}$ 和 $\mathbf{Z}^{\otimes n}\mathbf{v}$。

引理 4.3 ([BHKK10])。 集合覆盖可以在 $|\uparrow\mathcal{S}|n^{O(1)}$ 时间内求解。

[6] ζ 变换和莫比乌斯变换通常是在任意偏序集的背景下更一般地定义的；我们将讨论范围限定在子集格这一特殊情形下的这些变换。

集合覆盖的一个经过深入研究的特殊情况（尽管集合数量呈指数级）是第1节中定义的 k-染色问题。 k-染色问题可以看作集合覆盖的一个实例，其中 $\mathcal{S}$ 是 G的所有（按包含关系）极大独立集构成的族。因此，定理 4.2 给出了首个用于 k-染色的 $O^*(2^n)$ 时间算法，在经过长期研究后揭示了最坏情况复杂性的一个新自然屏障。[7]

当 $\mathcal{S}$ 定义为极大独立集的集合时， $\uparrow\mathcal{S}$ 中的所有集合都是 G 的支配集。利用Shearer熵引理， [BHKK10] 证明，最大度数为 d 的图至多有 $(2^{d+1}-1)^{n/(d+1)}$ 个支配集。因此，引理 4.3 得出以下结果：

定理 4.4([BHKK10])。 对于最大度数为 d 的图，任意 d, k-着色问题都可以在时间 $O^*((2^{d+1}-1)^{n/(d+1)})$ 内求解。

我们将展示引理 4.3 中思想的更多应用，并采用以下扩展（这是对 [BHKK10] 中观察结果的轻微扩展）：

引理 4.5 (跨越中间层)。 设 $\mathcal{S}$ 向下封闭（即，如果 $S\in\mathcal{S}$ 且 $S'\subseteq S$，则 $S'\in\mathcal{S}$），并设 $\mathtt{A}$。是一个预言机算法，能在 $\mathtt{T}$。时间内判定给定集合 $S\subseteq U$ 是否属于 $\mathcal{S}$。那么存在一个算法，以集合族 $\mathcal{F}\subseteq\binom{U}{\geq|U|/2}$ 为输入，并判定是否存在 $S_1,\ldots,S_k\in\mathcal{S}, i\in[k]$ 和 $F\in\mathcal{F}$，使得 $\cup_{j=1}^{i-1}S_j\supseteq F$ 且 $\cup_{j=i}^{k}S_j\supseteq U\setminus F$。

此外，对于每个 $\varepsilon>0$，都存在一个 $\varepsilon'>0$，使得如果 $|{\downarrow}\mathcal{F}|\leq(2-\varepsilon)^n$，那么算法的运行时间为 $O((2-\varepsilon')^n\cdot\mathtt{T}_\mathtt{o})$ 时间。

证明概要。 将引理 4.1 证明中的算法限制为只计算以 ${\downarrow}\mathcal{F}$ 为索引的表项，即可得到以下结果：

断言4.6。 在 $\mathtt{T}_\mathtt{o}|{\downarrow}\mathcal{F}|n^{O(1)}$ 时间内，我们可以针对每个 $F\in{\downarrow}\mathcal{F}$ 和每个 $i\in[k]$，计算是否存在集合 S_1、……、 $S_i\in\mathcal{S}$，使得 $\cup_{j=1}^{i}S_j\supseteq F$。

首先，使用断言4.6，针对每个 i 和 $F\in\mathcal{F}$，计算是否存在 S_1、……、 $S_i\in\mathcal{S}$，使得 $\cup_{j=1}^{i}S_j\supseteq F$。其次，我们使用集合族 $\mathcal{F}'=\{U\setminus F\colon F\in\mathcal{F}\}$，通过断言4.6，针对每个 i 和 $F\in\mathcal{F}$，计算是否存在 S_{k-i}、……、 $S_k\in\mathcal{S}$，使得 $\cup_{j=i}^{k}S_j\supseteq U\setminus F$。现在，当且仅当存在某个 F 和 i 满足这些条件时，我们输出 YES。

该算法的运行时间为 $(|{\downarrow}\mathcal{F}|+|{\downarrow}\mathcal{F}'|)\mathtt{T}_\mathtt{o}\cdot n^{O(1)}$。由于根据假设， $|{\downarrow}\mathcal{F}|\leq(2-\varepsilon)^n$，因此只需给 $|{\downarrow}\mathcal{F}'|$ 一个上界。令 a_σ 为 ${\downarrow}\mathcal{F}'$ 中大小为 σn 的集合数量。由于 ${\downarrow}\mathcal{F}'$ 中的集合必须是某个 $F\in\mathcal{F}$ 的 $U\setminus F$ 的子集，因此我们有

$$a_\sigma\leq\sum_{F\in\mathcal{F}}\binom{|U\setminus F|}{\sigma n}\leq(2-\varepsilon)^n\binom{n/2}{\sigma n}.$$

另一方面，显然有 $a_\sigma\leq\binom{n}{\sigma n}$。因此，根据 (1) 及其后续说明，我们有

$$a_{\sigma n}\leq\left(\min\{2^{h(\sigma)},(2-\varepsilon)2^{h(2\sigma)/2}\}\right)^n. \tag{8}$$

现在，引理即可得证，因为当 $\sigma\neq\frac{1}{2}$ 时， $h(\sigma)<1$；而如果 $\sigma=\frac{1}{2}$，那么 $h(2\sigma)=0$。更形式化且定量地说，可以使用不等式 $h(\frac{1}{2}-x)=h(\frac{1}{2}+x)\leq 1-x^2$，该不等式对所有 $x\in(0,\frac{1}{2})$ 均成立。 □

[7]实际上，与 k-合取范式可满足性问题和集合大小至多为 k 的集合覆盖不同，我们目前甚至不知道如何在时间 $O((2-\varepsilon)^n)$ 内求解 $k=7$ 的 k-着色问题，其中 $\varepsilon>0$ 为某个常数。对于 $k=5$，6，这类改进算法直到最近才被发现（参见 [Zam21]）。

如下文将看到，在集合覆盖的若干特殊情形中，可以构造一个族 $\mathcal{F}$，它见证了集合覆盖实例的一个解，并且该族足够小，因此可以通过引理 4.5 得到非平凡的快速算法。第一个例子是通过随机方式构造 $\mathcal{F}$：以概率 $1/2^{\Omega(n)}$ 将每个大小约为 $n/2$ 的集合加入其中：

定理 4.7 ([Ned16])。对于每个 $\sigma > 0$，都存在 $\varepsilon > 0$，使得任何满足 $k \geq \sigma n$ 的集合覆盖实例，都可以通过蒙特卡洛算法在 $|\mathcal{S}|^{O(1)}(2-\varepsilon)^n$ 时间内求解。

证明概要。设 ε_1 是一个仅取决于 σ 的参数，其值将在后文确定。如果解中包含一个大小至少为 $\varepsilon_1 n$ 的集合，我们可以通过猜测该集合，并利用引理 4.2 求解由猜测集合之外的所有元素所诱导的集合覆盖实例，在 $|\mathcal{S}|^{O(1)}(2-\varepsilon_1)^n$ 时间内检测到这个解。因此，从现在起，我们可以假设解中不会使用任何大小至少为 $\varepsilon_1 n$ 的集合。

现在，我们的算法构造 $\mathcal{F} \subseteq 2^U$：对 U 中每个基数至少为 $n/2$ 且至多为 $(1/2+\varepsilon_2)n$ 的子集，独立地以 $p = 2^{-\sigma n}$ 的概率将其纳入其中，其中 ε_2 是一个仅取决于 σ 且远小于 ε_1 的参数。随后，它使用集合族 $\mathcal{F}$ 和 $\mathcal{S}$ 运行引理 4.5，并输出是否检测到了 S_1、……、$S_k \in \mathcal{S}$、$i \in [k]$ 和 $F \in \mathcal{F}$，使得 $\cup_{j=1}^{i-1} S_j \supseteq F$ 且 $\cup_{j=i}^{k} S_j \supseteq U \setminus F$。

确保 ε_2 远小于 σ，利用马尔可夫界以及与 (8) 中类似的论证，不难证明，对于某个仅取决于 $\sigma > 0$ 的 $\varepsilon' > 0$，有 $\Pr[|{\downarrow}\mathcal{F}| \leq 2^{(1-\varepsilon')^n}] \geq 0.9$。因此，该算法的运行时间符合所声称的界。

接下来需要分析算法正确的概率。如果算法输出 YES，显然找到的集合 $S_1, \ldots, S_k$ 确实构成了该集合覆盖实例的一个解。反过来，假设 $S_1, \ldots, S_k \subseteq \mathcal{S}$ 的并集等于 U。任意选取 $S_i' \subseteq S_i$，对于每个满足 $\cup_{i=1}^{k} S_i' = U$ 的 $i \in [k]$，并且当 $i \neq j$ 时，S_i' 与 S_j' 不相交。若我们随机选取一个子集 $L \subseteq [k]$，方法是将 $[k]$ 中的每个元素以 $1/2$ 到 L 的概率独立纳入其中，那么根据霍夫丁界，有 $\Pr[n/2 \leq \sum_{i \in L} |S_i| \leq (\frac{1}{2}+\varepsilon_2)n] \geq \Omega(1)$，因为对于每个 $i \in [k]$，$|S_i| \leq \varepsilon_1 n$，且 ε_1 远小于 ε_2 和 σ。这意味着集合

$$\mathcal{W} = \left\{ \bigcup_{i \in L} S_i : n/2 \leq \sum_{i \in L} |S_i| \leq (\tfrac{1}{2}+\varepsilon_2)n \right\},$$

大小为 $\Omega(2^k) = \Omega(2^{\sigma n})$，因此 $\mathcal{F}$ 和 $\mathcal{W}$ 相交的概率为

$$1 - (1-p)^{|\mathcal{W}|} \geq 1 - \exp(-p|\mathcal{W}|) = 1 - \exp(-\Omega(1)) = 1 - \Omega(1).$$

证明成立，因为如果 $\mathcal{F}$ 和 $\mathcal{W}$ 相交，该算法显然会输出 YES。 □

集合覆盖的一个更特殊（但或许更自然）的情形是所有集合的大小都有界。以下结果是通过对“相关”子集进行动态规划得到的。

定理 4.8 ([Koi09])。对于每个 d，都存在一个仅取决于 d 的 $\varepsilon > 0$，使得任何集合大小至多为 d 的集合覆盖实例都能在 $O^*((2-\varepsilon)^n)$ 时间内求解。

我们省略证明，转而参阅原始论文 [Koi09]，或参见 [FK10 第 3 节4]中的教科书式讲解。事实上，Koivisto 的算法运行时间为 $2^{(1-1/\Omega(d))n}$。这是目前已知的最佳运行时间；颇为奇妙的是，它与第 3 节讨论的 k-合取范式可满足性问题的最快运行时间相似。

4.3 装箱问题的应用

另一个引理 4.5 背后的思想发挥关键作用的例子，是近期针对以下问题提出的一种算法。对于 w： $[n] \to \mathbb{N}$，我们使用简写记号 $W(X) = \sum_{e \in X} w(e)$ 。

装箱问题**输入：** 非负整数 $w(1) \ldots, w(n)$ c, k**问题：** 将 $[n]$ 划分为集合 $S_1, \ldots, S_k$，使得对于所有 i， $w(S_i) = \sum_{e \in S_i} w(e) \leq c$。

事实上，装箱问题是集合覆盖的一个特例，其中 $[n] = U$ 和 $\mathcal{S}$ 由所有满足 $w(S) \leq c$ 的子集$S \subseteq [n]$ 组成。因此，定理 4.2 已经能在 $2^n n^{O(1)}$ 时间内解决这个问题。不过，可以利用该实例中集合的特殊结构:

定理 4.9 ([NPSW23]). 对于每个 k，都存在一个 $\varepsilon > 0$，使得装箱问题可以在 $O^*((2-\varepsilon)^n)$ 时间内得到解决。

由于该定理的证明需要相当多超出本综述范围的（技术性）思想，我们作出两个简化假设:

A1 该实例在以下意义下是紧的： $w([n]) = k \cdot c$,

A2 存在一个解 S_1，……， S_k 和一个整数 ℓ，使得 $\sum_{j=1}^{\ell} |S_j| = n/2$。

假设 **A1** 相当强。粗略地说，在原始论文 [NPSW23] 中，通过以一种使每个箱中解的松弛量（即数量$c - w(S_i)$）变为 0 的方式对整数进行舍入，消除了这一假设。假设 **A2** 相对温和一些，可以采用类似于定理 4.7证明的方法加以处理：粗略地说，如果存在一个解 S_1，……，S_k，其中某个 S_i 非常大，我们可以通过其他方式检测出该解。否则，我们可以将该解中的集合排序为 S_1，……， S_k，使得 $\sum_{j=1}^{\ell} |S_j| = n/2$。

在这些假设下，下面这个简单引理立即表明，前述用于装箱问题的 $2^n n^{O(1)}$ 时间算法还有很大的改进空间

引理 4.10。 令 $w(1)$…， $w(n)$ k， c 为一个满足假设***A1***和***A2***的装箱问题实例，并令

$$\mathcal{F} = \{X \subseteq [n] : w(X) = c \cdot k/2\}. \tag{9}$$

对于每个 $\varepsilon > 0$，都存在一个 $\varepsilon' > 0$，使得若 $|\mathcal{F}| \leq (2-\varepsilon)^n$，则该装箱问题实例可以在$(2-\varepsilon')^n$ 时间内求解。

证明概要。 使用标准方法可以在$(2-\varepsilon)^n n^{O(1)}$ 时间内枚举集合 $\mathcal{F}$（第 6 节也概述了这些方法）。应用引理 4.5，其中 $\mathcal{S}$由所有满足 $w(S) = c$ 的子集 $S \subseteq [n]$ 构成，而 $\mathcal{F}$则按式（9）定义。 □

但是，如果 $|\mathcal{F}| \geq (2-\varepsilon)^n$，会怎样? 这看起来是一个相当特殊的情形，但其确切结构并不立刻明了。为了进一步分析，我们来定义最大频数

$$\beta(w) = \max_{v} |\{X \subseteq \{1, \ldots, n\} : w(X) = v\}|$$

T因此， $|\mathcal{F}| \leq \beta(w)$。我们还引入参数不同子集和的数量:

$$|w(2^{[n]})| = |\{w(X) : X \subseteq [n]\}|.$$

这两个参数将相互竞争，并引导我们采用一种双赢策略，具体取决于其中哪一个足够小：通过对子集和的伪多项式时间算法进行变体，可以相当直接地在时间 $|w(2^{[n]})|^k n^{O(1)}$ 内解决装箱问题。[8]

为了直观理解这些参数为何相互竞争，下面给出两个极端情形：

$$(w_a(1), w_a(2), \cdots, w_a(n)) = (0, 0, 0, \cdots, 0) \qquad |w_a(2^n)| = 1, \quad \beta(w_a) = 2^n,$$
$$(w_b(1), w_b(2), \cdots, w_b(n)) = (1, 2, 4, \cdots, 2^{n-1}) \quad |w_b(2^n)| = 2^n, \quad \beta(w_b) = 1.$$

加法组合学中的以下结果表明，这两个参数确实相互竞争，因此也提供了证明定理 4.9 所需的最后一个要素：

引理 4.11 ([NPSW23])。对于每个 $\varepsilon > 0$，都存在一个 $\varepsilon' > 0$，使得 $|w(2^{[n]})| \geq 2^{\varepsilon n}$，那么 $\beta(w) \leq (2 - \varepsilon')^n$。

虽然 [NPSW23] 的原始证明给出了一个 ε'，其值相对于 ε 呈指数级小，但 [JSS21] 将其改进为对 [的多项式依赖。

4.4 带有容器的集合覆盖

本节给出集合覆盖的一个新的承诺版本，下一节将用它来证明定理 4.16。我们的证明不同于 [Zam23], 的原始证明，但所得结果的通用性较弱。不过，从表述角度来看，我们的替代证明有一个优点：可以将关于集合覆盖的一般性结果作为一个（某种程度上自然的）黑箱来使用。

> 带容器的集合覆盖**输入：** 基数为 n 的全集 U，'容器'子集 C_1、……、$C_k \subseteq U$，以及预言机 A_o。预言机 A_o 以 U 的一个子集作为输入，并在 T_o 时间内输出**真**或**假**。保证条件是：若存在 S_1、……、S_k，满足 $\cup_{i=1}^k S_i = U$，且对于每个 $i \in \{1, \ldots, k\}$，$A_o(S_i) =$ **为真**，则存在这样的 S_1、……、S_k，并满足附加条件：对于每个 $i \in \{1, \ldots, k\}$，$S_i \subseteq C_i$。**问题：** 是否存在 S_1、……、S_k，满足 $\cup_{i=1}^k S_i = U$，且对于每个 $i \in \{1, \ldots, k\}$，$A_o(S_i) =$ **为真**?

定理 4.12。对于任意 k，存在 ε、$\hat{}\varepsilon > 0$，使得满足 $|C_i| \leq (1/2 + \varepsilon)n$ 的带容器的集合覆盖 C_1、……、C_k，可以在 $O((2 - \hat{}\varepsilon)^n \cdot T_o)$ 时间内求解。

为了证明定理 4.12，我们先证明一个关于集合族的引理。为了使用标准的图论记号，我们用二分图的形式表述该引理（即，用 $N(v)$ 表示顶点 v 的邻域，用 $d(v)$ 表示 $|N(v)|$，并用 $N(X)$ 表示 $\cup_{v \in X} N(v)$）。

引理 4.13。对于任意 k，存在 $\varepsilon, \varepsilon' > 0$，使得以下结论成立：设 $G = (A \cup B, E)$ 是一个二分图，其中 $|A| = k$ 和 $|B| = n$ 满足：对每个 $a \in A$，都有 $d(a) \leq (\frac{1}{2} + \varepsilon)n$；并设 $w : A \to \mathbb{N}$。那么存在一个 $X \subseteq A$，使得 $w(X) \geq w(A)/2$，且 $|N(X)| \leq (1 - \varepsilon')n$。

在证明该引理之前，我们先陈述一个证明中需要用到的引理（奇怪的是，我们并不清楚是否存在一种更简单的方法，可以不借助这个引理来证明引理 4.12）。

引理 4.14（[FT87]）。对于任意向量 $w = (w_1, \ldots, w_k) \in \mathbb{Z}^k$，都存在一个向量 $w' = (w'_1, \ldots, w'_k) \in \mathbb{Z}^k$，使得 $||w'||_\infty \leq 2^{O(k^3)}$，并且对于每个 $x \in \{-1, 0, 1\}^k$，都有 $sign(\langle w, x \rangle) = sign(\langle w', x \rangle)$。

[8] 对于 $i = 1, \ldots, n$ 和 $(c_1, \ldots, c_k) \in w(2^{[n]})^k$，定义一个表项，用于存储是否存在 $[i]$ 的一个划分 $S_1, \ldots, S_k$，使得 $w(S_i) = c_i$。

引理 *4.13* 的证明。为方便起见，假设 k 为偶数（当 k 为奇数时，可将其约简为 k 为偶数的情形，因为我们可以将 k 增加 1，并添加一个满足 $w_{k+1}=0$ 的集合 $C_{k+1}=\emptyset$）。

根据引理 4.14，存在一个函数 $w'\colon\ A\to\{0,\dots,W\}$，满足 $W=2^{O(k^3)}$，使得对所有 $X\subseteq A$，均有 $w(X)\geq w(A)/2$ 当且仅当 $w'(X)\geq w'(A)/2$：事实上，将函数 w 和 w' 解释为向量即可直接得到这一结论。

如果 $w'(N(v))>w'(A)/2$，则称顶点 $v\in B$ 坏，否则称其为 好。由于 $w'(N(v))$ 和 $w'(A)$ 都是整数，因此如果 v 是坏的，则有 $w'(N(v))\geq w'(A)/2+\frac{1}{2}$。若令 b 表示坏元素的数量，则有

$$b\left(w'(A)/2+\tfrac{1}{2}\right)\leq\sum_{a\in A}w'(a)d(a)\leq\sum_{a\in A}w'(a)(\tfrac{1}{2}+\varepsilon)n=w'(A)(\tfrac{1}{2}+\varepsilon)n.$$

因此，如果我们设置 ε 使得 $\varepsilon\leq 1/(4kW)=1/2^{O(k^3)}$，就得到

$$b\leq\left(\frac{\frac{1}{2}+\varepsilon}{\frac{1}{2}+1/(2w'(A))}\right)n\leq\left(\frac{\frac{1}{2}+1/(4w'(A))}{\frac{1}{2}+1/(2w'(A))}\right)n\leq\left(1-\frac{1}{4w'(A)}\right)n.$$

因此，B 中至少有 $n/(4w'(A))=n/2^{O(k^3)}$ 个顶点是好顶点。根据鸽巢原理，存在 $Y\subseteq A$，使得至少有 $n/(2^{O(k^3)}2^k)$ 个好顶点 v 满足 $N(v)=Y$。由于这些顶点都是好顶点，$w'(Y)\leq w'(A)/2$。若设 $\varepsilon'=1/(2^{O(k^3)}2^k)$，则引理中的集合 X 可构造为 $X=A\setminus Y$：它满足 $|N(X)|\leq(1-1/(2^{O(k^3)}2^k)|B|$，因为所有好顶点都不在 $N(X)$ 中；而 $w'(X)=w'(A)-w'(Y)\geq w'(A)/2$ 意味着 $w(X)\geq w(A)/2$。 □

引理 *4.12* 的证明。固定 k 后，令 ε 和 ε' 为引理 4.13 给出的对象。遍历所有子集 $L\subseteq[k]$，并考察 $C_L=\cup_{i\in L}C_i$。若 $|C_L|\leq(1-\varepsilon')n$，则使用引理 4.5 和 $\mathcal{F}=\binom{C_L}{\geq n/2}$、$\mathcal{S}$，检测是否存在 S_1、……、S_k 及 $i\in[k]$，使得对某个 $F\in\mathcal{F}$，有 $\cup_{j=1}^{i-1}S_j\supseteq F$ 和 $\cup_{j=i}^{k}S_j\supseteq U\setminus F$。当且仅当上述任一轮迭代检测到这样的 S_1、……、S_k 时，输出 **真**。

我们有 $|\downarrow\mathcal{F}|\leq 2^{|C_L|}\leq 2^{(1-\varepsilon')n}$，因此，引理 4.5 的算法在时间 $O((2-\hat{}\varepsilon)^n)$ 内运行，其中 $\hat{}\varepsilon>0$。

为了证明该算法的正确性，注意到如果算法输出 **真**，则它始终是正确的。反之，设 $S_1,\dots,S_k$ 是一个解，使得对于每个 $i\in\{1,\dots,k\}$ 都有 $S_i\subseteq C_i$。将引理 4.13 应用于图 G，其中 $B=U$，并且对于每个 $i=1,\dots,k$，存在一个顶点 $a\in A$，满足 $N(a)=C_i$ 以及 $w(a)=|S_i|$。由该引理可知，存在一个集合 $X\subseteq[k]$，使得 $|\cup_{i\in L}C_i|\leq(1-\varepsilon)n$ 且 $\sum_{i\in L}|S_i|\geq n/2$。如果我们尝试这个 L，集合 $\cup_{i\in L}S_i$ 属于 $\binom{C_L}{\geq n/2}$，因此引理 4.5 的算法将输出 **真**。 □

4.5 对正则图着色的应用

现在，我们将利用定理 4.12，为 k-着色问题设计一种算法。在 k 为常数且输入图是正则图（即每个顶点的邻居数量都相同）的特殊情况下，该算法的运行时间显著快于定理 4.2 所隐含的 $O^*(2^n)$ 时间算法。

为此，我们将使用一个由下述引理给出的容器族；该引理是组合学中“容器方法”最基本的结果之一。另见 [AS16, 定理 1.6。1]。

引理 4.15 ([Sap07]). 设 $G=(V,E)$ 是一个 n 个顶点的 d 正则图，且 $\varepsilon>0$。于是，可以在 $\ell\cdot$ 多项式(n) 时间内构造一族子集 $C_1,\dots,C_\ell\subseteq V$，使得

1. $\ell \le \binom{n}{\le n/(\varepsilon d)}$,

2. 对于每个 $i = 1, \dots, \ell$，我们都有 $|C_i| \le \frac{n}{\varepsilon d} + \frac{n}{2-\varepsilon}$，并且

3. G 的每个独立集都包含在某个 C_i 中。

定理 4.16 ([Zam23])。对于每个 k，都存在一个 $\varepsilon > 0$，使得在具有 n 个顶点的正则图上求解 k-着色问题需要 $O((2-\varepsilon)^n)$ 时间。

证明。设 $\varepsilon_0, \varepsilon' > 0$ 是定理 4.12 给出的常数，它们依赖于 k，使得在所有容器 C_i 的大小至多为 $(1/2 + \varepsilon_0)n$ 的情况下，带容器的集合覆盖问题可在 $(2-\varepsilon')^n$ 时间内求解。

令 $d_0 \ge 1/\varepsilon_0^2$ 为稍后确定的常数，并令 G 为 d-正则图。若 $d \le d_0$，则运行定理 4.4 中的算法。否则，应用引理 4.15，其中 $\varepsilon_1 := \varepsilon_0/2$。我们得到容器 $C_1, \dots, C_\ell$，满足

$$|C_i| \le n\left(\frac{1}{\varepsilon_1 d} + \frac{1}{2-\varepsilon_1}\right) \le n\left(\frac{1}{2} + \varepsilon_1 + \frac{1}{\varepsilon_1 d}\right) \le n\left(\frac{1}{2} + \varepsilon_0/2 + \frac{2}{\varepsilon_0 d}\right) \le n\left(\frac{1}{2} + \varepsilon_0\right).$$

现在，我们猜测包含独立集 $S_1, \dots, S_k$ 的容器；如果存在这样的集合，它们就构成了 G 的一个 k 着色的颜色类。由于有 $\binom{\ell}{k}$ 种这样的选择，因此运行时间将为

$$\binom{\ell}{k}(2-\varepsilon')^n \le \binom{n}{n/(\varepsilon_1 d)}^k (2-\varepsilon')^n = \left(2^{k \cdot h(1/(\varepsilon_1 d))}(2-\varepsilon')\right)^n.$$

由于当 p 趋近于零时，$h(p)$ 也趋近于零，因此我们可以将 d 选为 k 的函数（也依赖于 ε 和 ε'，但它们也由 k 决定），使得 $2^{k \cdot h(1/(\varepsilon_1 d))}(2-\varepsilon') < 2$。

□

4.6 集合覆盖与渐近张量秩

在令人振奋的新研究 BK24 和 Pra24] 中，作者证明：如果某一族三维张量具有较小的渐近张量秩，那么所有集合大小均有常数上界的集合覆盖问题可以在 $O(1.89^n)$ 时间内求解。与矩阵类似，张量 $\mathbf{T}$ 的秩 $rk(\mathbf{T})$ 是满足以下条件的秩一张量 $\mathbf{T}_1, \dots, \mathbf{T}_r$ 的最小数量 $r = rk(\mathbf{T})$：$\sum_{i=1}^{t} \mathbf{T}_i$（这里的求和是逐项进行的）。秩一张量是三个向量的外积（而矩阵的秩为 1，指的是该矩阵可以写成两个向量的外积）。张量 $\mathbf{T}$ 的渐近秩定义为 $\lim_{r\to\infty} rk(\mathbf{T}^{\otimes r})^{1/r}$。

Strassen [Str94] 猜想，任何满足某些温和条件的张量都具有较小的张量秩；有趣的是，目前尚待发现一个具有强超二次张量秩的三维张量族的显式例子。

这一新的联系能否为以下方向提供更多证据，仍有待观察：例如，将其与假设 3.2 联系起来，以证明眼下这些特定张量确实具有较大的张量秩（从而反驳 Strassen 猜想）；或者直接以开发更快的集合覆盖算法为目标。

5 路径寻找

近年来，另一个取得诸多进展的领域是寻找长路径和环的算法设计。形式化地说，我们考虑以下问题：

（无）向哈密顿性**输入：** 一个包含 n 个顶点的 G （无）向图。**问题：** G是否有哈密顿回路?

取得了若干突破性成果，其中最突出的是用于无向哈密顿性的 $O(1.66^n)$ 时间算法 [Bjö14]。这项研究始于寻找长度至少为 k [的路径的参数化算法 08], ，而这一研究又受到更早一系列论文的启发；那些论文通过行列式，以代数方法判定图是否存在完美匹配。我们将概述这项早期工作，以便在此基础上继续展开：

定义 5.1 (Tutte 矩阵 [Tut47])。 设 $G = (V, E)$ 为一个带有线性序 $\prec$ 的图，其中 V；设$\mathbb{F}$ 为一个域，对于每个 $i < j$ ，令 $x_{ij} \in \mathbb{F}$。定义

$$\mathbf{A}_G^{(x)}[i,j] = \begin{cases} x_{ij} & \text{if } \{i,j\} \in E \text{ and } i \prec j, \\ -x_{ji} & \text{if } \{i,j\} \in E \text{ and } j \prec i, \\ 0 & \text{otherwise .} \end{cases}$$

对于集合 V，我们用 $\Pi_{\mathrm{m}}(V)$ 表示顶点集为 V 的完全图的所有完美匹配构成的族。

引理 5.2 （[Tut47]）。 行列式$\det(\mathbf{A}_G^{(x)})$ 是关于变量 $x_{i,\ j}$ 的多项式，满足 g

$$\det(\mathbf{A}_G^{(x)}) = \sum_{M \in \Pi_{\mathrm{m}}(V)} \prod_{\substack{\{i,j\} \in M \\ i \prec j}} x_{ij}^2.$$

我们将多项式 $\det(\mathbf{A}_G^{(x)})$ 称为 $P_G(x)$。我们首先简要介绍一下这个引理在 [Lov79] 中的一个应用，用于获得一种快速的随机化算法，以确定 G 是否具有完美匹配：容易看出，当且仅当 G 不具有完美匹配时， P_G 是零多项式。由于 P_G 是次数至多为 n的多项式，我们可以使用下面的引理来检查 G 是否具有完美匹配：

引理 5.3 （多项式恒等式测试， [DL78, Sch80, Zip93]）。 设 $\mathbb{F}$ 为一个域，且 P（x_1，...， x_z）是一个关于 z个变量、取值于 $\mathbb{F}$ 的非零多项式，其次数至多为 d。若独立且均匀随机地选取 r_1，...， $r_z \in \mathbb{F}$，则 $\Pr[P$（r_1，...， r_z） $= 0] \leq d/|\mathbb{F}|$。

特别地，固定 $\mathbb{F}$ 为一个大小至少为 $2n$的域（其大小至少是 P_G的次数的两倍），并用来自 $\mathbb{F}$的随机元素替换变量 x_{ij}，再用基于高斯消元的算法在 n^ω 时间内计算 P_G [9]，若计算结果为非零数，则输出 **真**，当且仅当如此。

自 [Bjö14] 以来，已经出现了几种通过计算（指数数量的）行列式来解决（无）向哈密顿性问题的算法。我们将综述其中一些有代表性的方法，但不再详细讨论 [Bjö14] ，因为已有多篇相关论述见于教科书 [CFK+15]或综述（例如 [FK13、KW16]）。

[9]我们令 ω 表示使得 $n \times n$ 矩阵能够在 $n^{\omega+o(1)}$ 时间内相乘的最小常数， $2 \leq \omega \leq 2.73$。众所周知， $n \times n$ 矩阵的行列式可以在 $n^{\omega+o(1)}$ 时间内计算。

5.1 通过矩阵分解研究哈密顿性

我们首先给出一个相对简单的 $O^*(2^n)$ 时间算法，以说明随后将介绍的更快算法背后的主要思想。为此，我们先定义以下两个矩阵：

定义 5.4 (匹配连通性矩阵)。 对于偶数 $t \geq 2$，定义 $\mathbf{H}_t \in \{0,1\}^{\Pi_{\mathrm{m}}([t]) \times \Pi_{\mathrm{m}}([t])}$ 为

$$\mathbf{H}_t[A,B] = \begin{cases} 1, & \textit{if } A \cup B \textit{ is a Hamiltonian Cycle}, \\ 0, & \textit{otherwise}. \end{cases}$$

对于集合 V，我们用 $\Pi_2(V)$ 表示 V 的所有割构成的族，即将 V 划分为两个块的无序划分。

定义 5.5 (分裂矩阵)。 若匹配 $A \in \Pi_{\mathrm{m}}([t])$ 被割 $C \in \Pi_2([t])$ 分裂，则 A 的每条边要么包含在 C 中，要么与 C 不相交。对于偶数 $t \geq 2$，定义 $\mathbf{S}_t \in \{0,1\}^{\Pi_{\mathrm{m}}([t]) \times \Pi_2([t])}$ 为

$$\mathbf{S}_t[A,C] = \begin{cases} 1, & \textit{if } A \textit{ is split by } C, \\ 0, & \textit{otherwise}. \end{cases}$$

一个 $O^*(2^n)$时间算法，用于求解无向哈密顿性。 设 $A, B \in \Pi_{\mathrm{m}}([t])$。容易看出，同时分裂 A 和 B 的割 $C \in \Pi_2([t])$ 的数量为 2^{k-1}，其中 k 是图 $([t], A \cup B)$ 的连通分量数。因此，由于 2^{k-1} 为奇数当且仅当 $k = 1$，对于任意特征为 2 的域，我们有 $\mathbf{H}_t = \mathbf{S}_t \mathbf{S}_t^{\intercal}$。于是，令 $\mathbf{p}$,$\mathbf{q}$ 表示

$$\mathbf{p}[A] = \begin{cases} \prod\limits_{\substack{\{i,j\} \in A \\ i \prec j}} x_{ij}^2, & \text{if } A \in \Pi_{\mathrm{m}}([t]), \\ 0, & \text{otherwise}, \end{cases} \qquad \mathbf{q}[A] = \begin{cases} \prod\limits_{\substack{\{i,j\} \in A \\ i \prec j}} y_{ij}^2, & \text{if } A \in \Pi_{\mathrm{m}}([t]), \\ 0, & \text{otherwise}. \end{cases}$$

根据前面的观察，在任意特征为 2 的域中，$\mathbf{p}^{\intercal}\mathbf{H}_t\mathbf{q}$ 是零多项式，当且仅当 G 不存在哈密顿回路。因此，只需将域 $GF(2^k)$ 中的随机值代入 $k = \log_2 8n$，再代入 y_{ij}，并以直接的方式计算以下多项式（其次数至多为 $4n$），就能在 $O^*(2^n)$ 时间内检查是否存在哈密顿回路：

$$\mathbf{p}^{\intercal}\mathbf{H}_t\mathbf{q} = (\mathbf{p}^{\intercal}\mathbf{S}_t)(\mathbf{S}_t^{\intercal}\mathbf{q}) = \sum_{C \in \Pi_2([n])} \det(\mathbf{A}^{(x)}_{G[C]}) \det(\mathbf{A}^{(x)}_{G[V \setminus C]}) \det(\mathbf{A}^{(y)}_{G[C]}) \det(\mathbf{A}^{(y)}_{G[V \setminus C]}).$$

一个 $O^*(3^{n/2})$ 时间算法，用于 无向哈密顿性。 现在我们来看一个采用与前一个算法相同思路的更快算法。为了得到更快的算法，我们使用以下对$\mathbf{H}_t$的更高效因子分解：

引理 5.6 (窄割分解， [Ned20])。 设 $t \geq 2$ 为一个偶整数。存在一个多项式时间可计算函数 $C\colon \{0,1,2\}^{t/2-1} \to \Pi_2([t])$ 使得，若令 $\mathcal{C}_t = \{C(x)\colon x \in \{0,1,2\}^{t/2-1}\}$ 则对于特征为 2 的域 $\mathbf{F}$，有

$$\mathbf{H}_t = \mathbf{S}_t[\cdot, \mathcal{C}_t] \cdot \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}^{\otimes t/2-1} \cdot (\mathbf{S}_t[\cdot, \mathcal{C}_t])^{\intercal}.$$

有趣的是，在特征为 2 的域上，$\mathbf{H}_t$ 的秩等于 $2^{t/2-1}$ （见 [CKN18]），但我们不知道关于 **S**的更窄因式分解，而 [CKN18]中的更窄因式分解似乎更难与此处概述的算法方法结合。

定理 5.7。 二分图中的有向哈密顿性和无向哈密顿性可在 $O^*(3^{n/2})$ 时间内求解。

这些算法结果简化并改进了 [CKN18], 提出的类似方法，但反过来又不如 [Bjö14] 和 [BKK17]的结果。尽管如此，我们仍在此给出该结果，因为它可直接由引理 5.6 与标准方法结合推出。

定理 5.7 的证明概要。 我们重点讨论定理 5.7 的第二项。[10] 算法在算法 5 中概述。请注意，它需要 $O^*(3^{n/2})$时间，因为循环需要迭代 $3^{n/2-1}$ 次，循环位于

无向哈密顿性算法($G = (V, E)$) 输出：若 G 是哈密顿图，则以至少 $1/2$ 的概率输出真；否则输出假。

1 $\{i, j\} \in E$ $i < j$ $x_{ij}\ y_{ij} \in_R GF\ 2^k$ $k =$ $_2 8n$：

2

对于每个，选取，( )，其中 log：对于 $a \in \{0,1,2\}^{n/2-1}$ 执行 3：l$[a] \leftarrow$ det(A$^{(x)}_{G[C}$($_a$)) det(A$^{(x)}_{\otimes n/(}$($_a$)) 4：r$[a] \leftarrow$ det(A$^{(y)}_{G[C}$($_a$)) $\cdot$ det(A$^{(y)}_{G[V\setminus C}$($_a$)) 5：

$res \leftarrow$ l$^\intercal \cdot \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$ $\cdot$r 6：如果 $res \neq 0$ 则返回真，否则返回假

算法 5： 通过窄割分解求无向哈密顿性。

第 2 行中，第 3、4 行的行列式可用标准算法在多项式时间内计算；第 5 行的向量-矩阵-向量乘积可使用 Yates 算法（引理 4.1）和一次内积计算，在 $O^*(3^{n/2})$内计算。为证明正确性，令**Q**表示第 5 行的(3×3)-矩阵。注意，算法的输出 res 是多项式

[10]第一项可以用类似的方法证明，只需对一个方向上的弧使用 x-变量，并对另一个方向上的所有弧使用 y-弧。

$P(x,y)$ 在随机点 x, y 处的取值，其中 $P(x,y)$ 等于

$$= \sum_{a,b\in\{0,1,2\}^{n/2-1}} \left(\prod_{i=1}^{n/2-1} \mathbf{Q}[a_i,b_i]\right) \det\left(\mathbf{A}^{(x)}_{G[C(a)]}\right) \det\left(\mathbf{A}^{(x)}_{G[V\setminus C(a)]}\right) \det\left(\mathbf{A}^{(y)}_{G[C(b)]}\right) \det\left(\mathbf{A}^{(y)}_{G[V\setminus C(b)]}\right)$$

根据引理 5.2

$$= \sum_{a,b\in\{0,1,2\}^{n/2-1}} \left(\prod_{i=1}^{n/2-1} \mathbf{Q}[a_i,b_i]\right) \left(\sum_{\substack{M_1\in\Pi_{\mathrm{m}}(V)\\ C(a)\text{ splits }M_1}} \prod_{\substack{\{i,j\}\in M_1\\ i\prec j}} x_{ij}^2\right) \left(\sum_{\substack{M_2\in\Pi_{\mathrm{m}}(V)\\ C(b)\text{ splits }M_2}} \prod_{\substack{\{i,j\}\in M_2\\ i\prec j}} y_{ij}^2\right)$$

$$= \sum_{M_1,M_2\in\Pi_{\mathrm{m}}(G)} \left(\sum_{a,b\in\{0,1,2\}^{n/2-1}} [C(a)\text{ splits }M_1] \left(\prod_{i=1}^{n/2-1} \mathbf{Q}[a_i,b_i]\right) [C(b)\text{ splits }M_2]\right) \cdot \left(\prod_{\substack{\{i,j\}\in M_1\\ i\prec j}} x_{ij}^2\right)\left(\prod_{\substack{\{i,j\}\in M_2\\ i\prec j}} y_{ij}^2\right)$$

根据引理 5.6

$$= \sum_{M_1,M_2\in\Pi_{\mathrm{m}}(G)} \mathbf{H}_t[M_1,M_2] \left(\prod_{\substack{\{i,j\}\in M_1\\ i\prec j}} x_{ij}^2\right)\left(\prod_{\substack{\{i,j\}\in M_2\\ i\prec j}} y_{ij}^2\right).$$

由于 $P(x,\ y)$ 的度数至多为 $4n$，正确性由引理 5.3 得出 □

5.2 有向哈密顿性

现在我们概述证明以下定理的方法

定理 5.8。 *[[BKK17]]* 存在一种算法，给定一个含 n 个顶点的有向图 $G = (V, E)$ 和素数 p，可以在时间 $2^{n\left(1-O\left(\frac{1}{\log p}\right)\right)}$ 内计算哈密顿回路的数量模 p 的结果。

更强的版本见于 [BKK17]，但我们稍微简化了表述。

有趣的是，尽管有这种算法，目前仍然没有已知算法能在 $O^*((2-\varepsilon)^n)$ 时间内求解有向哈密顿性，其中 $\varepsilon > 0$ 为某个值。对于许多算法，包括本节前面介绍的两个算法，可以将那些计算模素数 p 的解的数量的算法与引理 5.3 结合起来，以求解检测是否存在解的问题；或者，也可以求解该问题的加权模计数版本，再通过隔离引理 [MVV87]将判定版本约简到这个版本。然而，5.8 背后的算法明确依赖于这样一个事实：许多中间计算的结果为 0（因此可以跳过），这使得确保解彼此不会抵消的目标变得复杂。

设 G 是一个 n 顶点的有向多重图。由于 G 是多重图，其边集 $E(G)$ 是一个多重集合。固定两个顶点 $s,\ t \in V(G)$，并假设从 t 到 s 恰好有一条边（如果有更多条边，只需将结果乘以该边的重数 $(t,\ s)$ 即可轻松调整此方法）。令 $\mathbf{A}_G$ 表示 G 的邻接矩阵，因此 $\mathbf{A}_G[v,w] \in \mathbb{Z}_{\geq 0}$ 描述了 $v,\ w \in E(G)$ 的弧数。我们假设 $\mathbf{A}_G[v,v] = 0$，即该图没有自环。记作

$d_G(w) = \sum_{v \in V} \mathbf{A}_G[v,w]$ 表示一个顶点的入度，并令 $\mathbf{D}_G$ 为对角矩阵，其中对于每个顶点 v，对应的条目为 $\mathbf{D}_G[v,v] = d_G(v)$。拉普拉斯矩阵 $\mathbf{L}_G$ 定义为 $\mathbf{D}_G - \mathbf{A}_G$。此外，定义关联矩阵

$$\mathbf{I}_G[u,(v,w)] = \begin{cases} 1, & \text{if } u = w, \\ 0, & \text{otherwise}, \end{cases} \qquad \text{and} \qquad \mathbf{O}_G[u,(v,w)] = \begin{cases} 1, & \text{if } u = v, \\ 0, & \text{otherwise}. \end{cases}$$

注意

$$\begin{aligned} \left((\mathbf{I}_G - \mathbf{O}_G) \cdot \mathbf{I}_G^\intercal\right)[u,x] &= \sum_{(v,w) \in E(G)} ([u=w] - [u=v])([w=x]) \\ &= \begin{cases} |\{(v,w) \in E(G) : u = w\}|, & \text{if } u = x, \\ -|\{(v,w) \in E(G) : u = v, w = x\}|, & \text{otherwise} \end{cases} \\ &= \begin{cases} d_G(u), & \text{if } u = x, \\ -\mathbf{A}_G[u,x], & \text{otherwise} \end{cases} \\ &= (\mathbf{D}_G - \mathbf{A}_G)[u,x] = \mathbf{L}_G[u,x]. \end{aligned} \tag{10}$$

令$\mathbf{L}_G^{-s}$ 表示从$\mathbf{L}_G$ 中删去由 s索引的行和列后得到的矩阵，并令$\mathbf{I}_G^{-s}$ 和$\mathbf{O}_G^{-s}$ 分别表示从$\mathbf{I}_G^{-s}$ 和 $\mathbf{O}_G^{-s}$ 中删去由 s 索引的行后得到的矩阵。由 (10) 可知

$$\mathbf{L}_G^{-s} = (\mathbf{I}_G^{-s} - \mathbf{O}_G^{-s})(\mathbf{I}_G^{-s})^\intercal.$$

我们称子集 $X \subseteq E(G)$ 为出树，如果 X 是有根生成树，且所有弧都从根向外指向。

引理 5.9。 如果 $X \in \binom{E(G)}{n-1}$ 我们有

$$\det\left((\mathbf{I}_G^{-s} - \mathbf{O}_G^{-s})[\cdot, X]\right) \cdot \det\left(\mathbf{I}_G^{-s}[\cdot, X]\right) = [X \textit{ is an out-branching rooted at } s]. \tag{11}$$

证明。 我们分四种情况讨论：

- 如果对于某个 $v \in V(G)$，有 $(v,s) \in X$，那么 $\det(\mathbf{I}_G^{-s}[\cdot, X]) = 0$，因为$\mathbf{I}_G^{-s}[\cdot, X]$中由 (v,s) 索引的列全为零
- 如果存在两条不同的边（u，w），（v，w）$\in X$那么 $\det$（$\mathbf{I}_G^{-s}[\cdot, X]$）$= 0$，因为某个顶点$x$ 没有入边，而 X 中与其对应的行全部为零
- 如果 X 构成一个环，则该环无法经过 s，因此以环中顶点为索引的$\mathbf{I}_G^{-s}$ –$\mathbf{O}_G^{-s}$ 的列之和为 0，这意味着 $\det\left(\right.$（$\mathbf{I}_G^{-s} - \mathbf{O}_G^{-s}$）$\left.[\cdot, X]\right) = 0$
- 否则， X 是以 s为根的出树。于是$\mathbf{I}_G^{-s}[\cdot, X]$ 是一个置换矩阵，因此其行列式为 $sgn(\sigma)$，其中 σ 将每个顶点映射到X中唯一一条进入该顶点的弧。但 σ 也是唯一一个对$\mathbf{I}_G^{-s}$ –$\mathbf{O}_G^{-s}$ 的行列式有贡献的置换，并且它也贡献因子 $sgn(\sigma)$。因此，(11)约简为 $sgn(\sigma)^2 = 1$。

□

现在，根据引理 5.9 和柯西-比内公式[11]，我们有

$$\begin{aligned}\det(\mathbf{L}_G^{-s}) &= \det((\mathbf{I}_G^{-s} - \mathbf{O}_G^{-s})(\mathbf{I}_G^{-s})^T) \\ &= \sum_{X \subseteq E(G)} \det\left((\mathbf{I}_G^{-s} - \mathbf{O}_G^{-s})[\cdot, X]\right) \cdot \det\left(\mathbf{I}_G^{-s}[\cdot, X]\right) \\ &= |\{B : B \text{ is an outbranching of } G \text{ rooted at } s\}|.\end{aligned} \tag{12}$$

引理 5.10. 令 $out(F)$ 表示起点在 F 的所有边，并令 $G - out(F)$ 表示从 G 中删除所有边 $out(F)$ 后得到的图。G 中包含弧 (t, s) 的哈密顿回路数量等于

$$\sum_{F \subseteq V(G) \setminus \{t\}} (-1)^{|F|} \det(\mathbf{L}_{G - out(F)}^{-s}). \tag{13}$$

证明将 (12) 与容斥结合起来：

证明。 根据 (12)，我们可以将 (13) 改写为

$$\sum_B \sum_{F \subseteq V(G) \setminus \{t\}} (-1)^F [B \cap out(F) = \emptyset] = \sum_B \sum_{F \subseteq sinks(B) \setminus \{t\}} (-1)^F,$$

其中，求和遍历所有以 s 为根的 G 的出树 B，而 $sinks(B)$ 表示 $\{v \in V(G) : \forall w \in V(G) : (v, w) \notin B\}$。由于每个非空集合的奇数大小子集数量与偶数大小子集数量相等，因此只有具有一个汇点（即 t）的集合 B 对 (13) 有贡献。这些集合对 (13) 恰好贡献一；又由于它们恰好是从 s 到 t、遍历所有顶点的哈密顿路径，因此（添加弧 (t, s) 后）是包含弧 (t, s) 的哈密顿回路，故引理得证。 □

那么，通过 13 计算哈密顿回路的数量究竟带来了什么好处，因为它仍然由 2^n 个求和项组成？关键在于，对于许多求和项，F $\det(\mathbf{L}_{G-out(F)}^{-s})$ 将等于 0（模一个较小的数 p）。特别地，在 $\mathbf{L}_{G-out(F)}^{-s}$ 中对应于 F 中某个顶点的任意一行里，除对角元外的所有元素都将为零。因此，如果我们在模一个较小的 p 下进行计算，并且该对角元也等于零（模 p），那么实际上该矩阵不具有满秩，因此行列式等于 0（模 p）。但是，如何确保该对角元模 p 等于 0 呢？

注意，从 t 到 v 的弧数量对于上述算法的结果完全无关紧要。因此，我们也可以从 t 向每个顶点 $v \neq s$ 添加 r_v 条（均匀且独立选取的）随机边。

注意，只要对于某个 $v \in F$，$\sum_{u \in V \setminus (F \cup \{t\})} \mathbf{A}[u, v] + r_v$ 模 p 等于 0，$\det(\mathbf{L}_{G-out(F)}^{-s})$ 就模 p 等于零。因此，一个加项 F 对 (13) 有贡献的概率为 $(1 - 1/p)^{|F|}$。算法背后的定理 5.8 会在所声称的时间界内枚举这些贡献项的一个超集，之后利用枚举得到的列表直接计算 (13)。我们略去枚举贡献项的过程细节，详情请参阅原始论文 [BKK17]。

[11] 柯西-比内公式表述如下：如果 $\mathbf{A}$ 是一个 $a \times b$ 矩阵，且 $\mathbf{B}$ 是一个 $b \times a$ 矩阵，那么 $\det(\mathbf{AB}) = \sum_{X \subseteq \binom{[b]}{a}} \det(\mathbf{A}[\cdot, X]) \det(\mathbf{B}[X, \cdot])$。

5.3 旅行商问题

如果我们为无向哈密顿性问题加入边权，就会得到以下著名问题

旅行商问题（TSP）**输入：** 一个无向图 $G = (V,\ E)$，距离 w：$E \to \mathbb{N}$ **问题：** 找出 G 的一个哈密顿回路 C，使 $w(C)$ 最小。

上一节的算法稍作调整后也适用于这个问题，但需要付出一定代价：例如，[Bjö14] 的一份论述给出了一个运行时间为 $O^*(1.66^n W)$ 的 TSP 算法，它还利用标准舍入技巧，给出了一个运行时间为 $O^*(1.66^n/\varepsilon)$ 的 $(1+\varepsilon)$-近似算法。不过，能否在该运行时间中避免伪多项式因子 W，仍然值得探讨。尤其是，之前讨论的代数算法似乎不可能在不承担这一伪多项式运行时间开销的情况下，精确求解这个优化变体。

因此，Bellman [Bel62], Held 和 Karp [HK62] 提出的自然的 $O^*(2^n)$ 时间动态规划算法能否改进，仍无定论。在热带电路模型中（在一定程度上模拟动态规划算法），甚至已经证明不存在更快的算法 [JS82]（另见近期出版的教科书 [Juk23]）。

针对有界度图 [BHKK12], 、路径宽和树宽较小的图 [BCKN15], ，以及二分图 [Ned20]，都提出了更快的算法；对于二分图的结果是在假设 $\omega = 2$ 成立的情况下，其中 ω 是满足 $q \times q$ 矩阵可在 $q^{\omega+o(1)}$ 时间内相乘的最小数。

[GS87] 证明，旅行商问题可以在 $4^n n^{O(\log n)}$ 时间内用多项式空间求解（例如参见 [FK10，第 10.1] 节）；但尚不清楚能否在 $O^*((4-\varepsilon)^n)$ 时间内用多项式空间求解，其中 $\varepsilon > 0$。

6 子集和

近十年来，另一个从细粒度复杂性角度取得令人瞩目进展的计算问题是：

子集和**输入：** 一个权重函数 w：$[n] \to [W]$，以及一个目标整数 t。**问题：** 是否存在一个子集 $S \subseteq [n]$，使得 $w(S) = t$。

这里我们使用前面章节中的记号，即 $w(X) = \sum_{e \in X} w(e)$；如果 $\mathcal{F} \subseteq 2^{[n]}$ 是一个集合族，我们也将 $w(\mathcal{F}) = \{w(F) :\ F \in \mathcal{F}\}$记作。

由于在 n 较小的情形下，可以将更一般的背包问题归约为子集和（参见 [NvLvdZ12]），因此以下所有评论同样适用于（或许更自然、也更核心的）背包问题。不过，为简洁起见，我们将注意力限制在子集和上。

6.1 中途相遇法与表示法

早在 50 年前，人们就提出了一种优雅的算法，它能以快于完全朴素的 $O^*(2^n)$ 时间算法的速度求解子集和 [HS74]。在这一算法以及我们随后将要讨论的算法中，两个“列表”将发挥核心作用。

在第一种方法中，这些列表的定义如下：将集合 $[n]$ 划分为两个大小均为 $n/2$ 的集合 L, R（为方便起见，在此处及下文均假设 n 是 4 的倍数，因此也是偶数），

并定义

$$\mathcal{L} := 2^L, \qquad \mathcal{R} := 2^R. \tag{14}$$

现在，算法如下：首先，枚举并排序 $w(\mathcal{R})$ 中的数字，然后对于每个 $X \in \mathcal{L}$，我们通过二分搜索检查是否存在 $t - w(X) \in w(\mathcal{R})$。如果存在这样的元素，我们就可以输出 **真**（因为我们可以取 $S = X \cup Y$，其中 Y 是满足 $w(Y) = t - w(X)$ 的 R 的子集，而 X 和 Y 不相交）。如果在任何一次迭代中都不存在这样的元素，我们就可以输出**假**（因为如果存在一个集合 S，满足 $w(S) = t$，那么在处理 $X = S \cap L$ 的那次迭代中，我们就会发现它）。由于二分搜索需要 $\log |\mathcal{R}|$ 时间，这个过程的运行时间为 $O^*(2^{n/2})$。

一个值得关注的开放问题是，能否将其改进到例如 $O^*(2^{0.4999n})$ 时间。在 [HJ10]，针对随机实例（而非最坏情况分析），人们利用一种优雅的方法——表示法——在这一问题上取得了令人瞩目的进展。[HJ10] 的算法概述见算法 6。为了准确说明它能实现什么，我们需要以下定义：

定义 **n 6.1.** 一个伪解是一个二元组 $(X, Y) \in \binom{[n]}{n/4} \times \binom{[n]}{n/4}$，满足 $w(X) + w(Y) = t$.

我们作出如下假设（称为**A1-A3**）：

A1 伪解的数量至多为 $2^{0.9n}$。

此外，如果该实例是一个是实例，则存在一个集合 $S \subseteq [n]$，使得 $w(S) = t$,

A2 $|S| = n/2$，并且

A3 $|w(2^S)| = 2^{n/2}$.

可以证明，如果所有的 $w(1) \dots, w(n)$ 都是从 $[2^n]$ 和 $t := w([n/2])$ 中均匀且独立地随机选取的，那么假设 **A1-A3** 确实以高概率成立。

算法如算法 6 所示。该算法背后的思路如下：如果我们定义 $\mathcal{L} = \mathcal{R} = \binom{[n]}{n/4}$，那么能够见证解的配对 $(X, Y) \in \mathcal{L} \times \mathcal{R}$ 的数量（即 X 和 Y 不相交且 $X \cup Y = S$）与列表大小 $\binom{n}{n/4}$ 的比值为

$$\binom{|S|}{n/4} / \binom{n}{n/4} \geq \tfrac{1}{n} 2^{\left(h\left(\frac{1}{2}\right)/2 - h\left(\frac{1}{4}\right)\right)n} \geq 2^{-0.32n},$$

其中第一个不等式用到了 (1) 及其后续说明。这个比值大于 14 中定义的 $\mathcal{L}$ 和 $\mathcal{R}$ 的类似比值 $1/2^{n/2}$。我们可以通过采样一个见证来利用这一点：选取一个随机素数 p 并猜测 $t_L \equiv_p w(X)$。关键在于，这个单独的猜测也能确定模 p 的 $w(Y)$，因为 $w(Y) \equiv_p t - t_l$。

第 2 行可以按如下方式实现，使其在时间 $O^*(2^{0.45n} + |\mathcal{L}| + |\mathcal{R}|)$ 内运行：创建一个有向图 $G = (\{0, \dots, n\} \times \mathbb{Z}_p \times \{0, \dots, n/4\}, A)$，其中对于每个 $0 \leq i \leq n-1, j \in \mathbb{Z}_p, k \in \{0, \dots, n/4\}$，都有弧

$$((i, j, k), (i+1, j, k)), \quad \text{and} \quad ((i, j, k), (i+1, j + w(i+1) \bmod p, k+1))$$

位于 A 中。容易看出，从 $(0, 0)$ 到 $(n, t_L, n/4)$（分别到 $(n, t - t_L, n/4)$）的路径分别对应于 $\mathcal{L}$（分别对应于 $\mathcal{R}$）中的元素，并且可以用标准的（回溯）方法在所声称的时间界内枚举这些路径。

此外，在第 5 行，我们可以利用第 3 行构建的有序数据结构，通过二分搜索，在每个 $\mathcal{R}$ 项目上以 $O^*(|\mathcal{L}| \in \mathcal{R})$ 的时间枚举所有相关的 O^*。

因此，可以实现算法 6，使其以 $O^*(2^{0.45n} + |\mathcal{L}| + |\mathcal{R}| + P)$ 的时间运行，其中 P 是伪解的数量（X，Y）$\in \mathcal{L} \times \mathcal{R}$，且满足 $w(X) = t_L$（模 p）。由于 t_L 是均匀随机选取的，根据 **A1**，此类伪解的期望数量至多为 $2^{0.9n}/p = 2^{0.45n}$。因此，在假设 **A1** 成立的情况下，算法 6 的运行时间为 $O^*(2^{0.45n})$。

算法 RepMethod($w\colon [n] \to [W]$，t)**输出：** **真** 的概率至少为 $1/2$，如果存在一个 $X \subseteq [n]$，满足 $w(X) = t$；否则为 **假**。1：均匀且独立地采样一个素数 $p \in \{2^{0.45n}, \ldots, 2^{0.45n+1}\}$ 和 $t_L \in \mathbb{Z}_p$2：构造

$$\mathcal{L} := \left\{ X \in \binom{[n]}{n/4} : w(X) \equiv_p t_L \right\}, \qquad \mathcal{R} := \left\{ Y \in \binom{[n]}{n/4} : w(Y) \equiv_p t - t_L \right\}.$$

3：对每个 $Y \in \mathcal{R}$，以 $w(Y)$ 为键对 $\mathcal{R}$ 排序4：**对所有** $X \in \mathcal{L}$**执行**5：**对所有** $Y \in \mathcal{R}$，使得 $w(X) + w(Y) = t$ **执行**6：如果 X 和 Y 不相交，**返回 真**7：**返回假**

算法 6： 子集和表示方法

定理 6.2。 如果满足假设***A1-A3***，那么算法 *6* 在 $O^*(2^{0.45n})$ 时间内以至少恒定概率输出**真**。

证明概要。 由于已讨论过假设**A1** 下的运行时间，我们只关注正确性。显然，如果算法输出**真**，它就找到了两个不相交集合 X，Y，满足$w(X) + w(Y) = t$，因此 $X \cup Y$是一个解。根据假设**A2** 和**A3**，还需证明：如果存在一个集合 S，满足 $|S| = n/2$ 和 $w(2^S) = 2^{n/2}$，那么算法确实会以至少 $1/2$的概率返回**真** 。注意，$w(2^S) = 2^{n/2}$ 意味着 S 的所有子集相对于整数 w都会产生不同的和。因此 $w\left(\binom{S}{n/4}\right) = \binom{n/2}{n/4}$。根据将一个数对随机素数 p 取模所具有的哈希性质，可以推出以高概率

$$\left| \left\{ x \pmod p : x \in w\left(\binom{S}{n/4}\right) \right\} \right| \geq \Omega(2^{0.45n}).$$

在以此事件为条件时，以常数概率会选中 t_L，使得存在 $X \in \binom{S}{n/4}$ 满足 $w(X) \equiv_p t_L$，因此在第 6 行会观察到具有 $Y = S \setminus X$ 的数对 (X, Y) 互不相交（除非在更早的一次迭代中已经返回了**真** ）。

□

6.2 近期的更多进展

关于能否改进 [HS74] 的 $O^*(2^{n/2})$ 运行时间所形成的“中间相遇障碍”，也针对类似问题进行了研究（并得到了肯定的回答） [CJRS22, MNPW19, JW24]。

另一个广受关注的研究主题是算法的空间使用量。 [HS74]的一个著名改进是 [SS81], 提出的 $O^*(2^{n/2})$ 时间、 $O^*(2^{n/4})$空间算法；最近，该算法又被改进为 $O^*(2^{n/2})$时间、 $O^*(2^{0.249999n})$空间算法 [NW21] ，并进一步改进为 $O^*(2^{n/2})$时间

以及 [BCKM24]中的 $O^*(2^{0.246n})$空间算法。如果限制为只能使用多项式空间，那么很长一段时间以来，已知的最佳算法都是朴素的 $O^*(2^n)$算法。在 [BCJ11] 中，作者提出了使用环查找算法求解随机实例的思路；后来， [BGNV18] 将其扩展为一种 $O^*(2^{0.86n})$时间、多项式空间算法，该算法假设可以随机只读访问随机比特（这些比特不存储在内存中）。近年来关于伪随机性的研究成果可以去除这一假设 [CJWW22, LZ23]。

7 其他主题

如前所述，本综述绝无意穷尽该领域中困难问题细粒度复杂性的所有（近期）进展。下面我们将以极简的篇幅（顺序任意）讨论其中一些值得关注的方向，这些方向本可以收录在篇幅更长的综述中。

NP 完全问题的参数化复杂性。虽然我们在本综述开篇提到了参数化复杂性，但需要强调的是，在参数化复杂性领域内，也有不少研究探讨了本综述讨论的细粒度问题。例如，对于许多以树宽或路径宽为参数的问题，研究人员发现了运行时间为 $O^*(c^k)$ 的算法，并证明改进到 $O^*((c-\varepsilon)^k)$ 的时间将推翻假设 3。2 [LMS18]。其他值得关注且研究充分的例子包括 k-Path（参见 [BHKK17]，其中介绍了无向图上目前最快的算法）和反馈顶点集（参见 [LN22] ，其中介绍了无向图上目前最快的算法）。人们还证明，按解的大小参数化的 NP 难“子集”问题的细粒度复杂性，会通过一种称为“单调局部搜索”的方法，直接影响按搜索空间大小参数化的细粒度复杂性（例如参见 [FGLS19]）；其方式与算法 2 有些相似。

（条件性）下界／归约。在这篇（乐观取向的）综述中，我们略过了一个重要且广泛的主题：（条件性）下界。也就是说，有一些证据，或者在某个假设成立的前提下给出的证明，表明某些朴素算法无法得到改进。

在这一方向上，最受关注且最相关的假设是假设 3.2，但令人意外的是，对于本综述讨论的少数问题，研究人员能够根据假设 3.2 推导出下界。一些值得注意的下界包括：击中集和集合划分 [CDL+16] 的若干等价关系，子集和 [ABHS22] 的伪多项式运行时间的紧下界，以及由 [LMS18]，开创的大量针对以树宽为参数的运行时间的下界；不过，后两者或许更适合归为精细化参数化复杂性结果。

一个尚未解决的重要问题是：对于某个 $\varepsilon > 0$，是否存在一个运行时间为 $O^*((2-\varepsilon)^n)$ 的集合覆盖算法会推翻假设 3.2；或者反过来，推翻假设 3.2 是否意味着存在一个运行时间为 $O^*((2-\varepsilon)^n)$ 的集合覆盖算法（对于某个 $\varepsilon > 0$）。目前也不清楚关于集合覆盖的新结果 [BK24, Pra24] 与此有何关联。

这类联系由 [CDL+16] 针对这些问题的奇偶性 版本给出：粗略地说，如果且仅如果可以在 $O^*((2-\varepsilon')^n)$ 时间内（对于某个 $\varepsilon' > 0$）统计一个 n 变量 k-合取范式公式的解的数量，那么就可以在 $O^*((2-\varepsilon)^n)$ 时间内（对于某个 $\varepsilon > 0$）计算一个包含 n 个元素的集合系统的集合覆盖数量模 2。这些联系由以下人员建立：

将这两个问题都与计算二分图中独立集数量的奇偶性这一任务联系起来。受此启发，研究它的判定问题或许也很有趣：

受约束二分图独立集**输入：** 一个二分图 $G = (A \cup B,\ E)$ 和整数 $t_A,\ t_B$**问题：** 是否存在一个大小为 G 的独立集 I，使得 $\lvert I \cap A\rvert = t_A$ 且 $\lvert I \cap B\rvert = t_B$?

基于上述动机，我们认为一个有趣的问题是：对于某个 $\varepsilon > 0$，这个问题能否在 $O^*((2-\varepsilon)^{|A|})$ 时间内解决?

由于基于假设 3.2 的强下界不足，一些研究人员开始研究某些算法的宽松版本是否存在（例如证明系统 [CGI+16] 以及 Merlin-Arthur [Wil16] ），或研究多项式公式 [BKM+24, KM24, BGK+23]），以此为“证明困难性的困难”提供证据。

分支算法。 在 NP 完全问题的细粒度复杂性研究中，分支算法是一种受到广泛研究的重要范式。借助这一范式和先进的分析方法（例如“度量与征服” [FGK09]），研究人员得以为独立集、支配集等基础问题，在输入图顶点数方面取得最佳的最坏情况运行时间界。更多详情请参阅 [FK10 第 2 章和 6]。

单边概率多项式时间算法。 一种自然但研究较少的指数时间算法模型是单边概率多项式时间（OPP）算法。这类算法在多项式时间内运行，输出**假**时总是正确的，但只有在是实例上才保证以指数倒数级的小概率输出**真** 。[12]

尽管这是一个自然且有趣的模型（因为它涵盖了大多数分支算法），文献中已提出了若干针对该模型所涵盖算法的有趣下界。例如，对于 n 输入的电路可满足性问题，[13] 在假设 3.1 成立的条件下，没有多项式时间蒙特卡洛算法能以概率 $(2-\varepsilon)^{-n}$ 输出**真** ，如 [PP10], 中所示；并且，除非多项式层级坍缩 [Dru13]，否则不存在多项式时间算法，能对每个 n 输入的 k 合取范式可满足性公式，以至少 $2^{-n^{1-\Omega(1)}}$ 的概率输出一个满足赋值。

电路下界。 近期，一条旨在证明电路下界的研究路线总结了算法优于朴素算法的一项显著应用。研究表明，对于布尔电路可满足性问题，即使算法仅比朴素算法略有改进，也意味着电路下界；而这类算法可以通过采用基于快速矩阵乘法或引理 4.1 的批量求值技术来实现。更多详情请参阅综述 [Wil14] 。

粗粒度复杂性NP完全问题。即使在粗粒度复杂性范围内，近期也出现了一些令人惊讶的结果，而一些基础问题的复杂性仍难以捉摸。例如，对于子图同构， [CFG+17]证明了

[12]一个用于 k-合取范式可满足性问题的 OPP 算法示例，是随机抽取一个解，并输出该解是否满足给定公式。
[13]在这个问题中，给定一个包含 n 个输入的布尔电路，要求判断是否存在一种输入赋值，使电路的计算结果为**真**。

简单的 $n^{O(n)}$ 时间算法无法改进到 $n^{o(n)}$，除非假设 3.1 不成立，其中 n表示图的顶点数。另一方面，有些出人意料的是，[BNvdZ16] 证明了，平面图上的子图同构可以在 $2^{O(n/\log n)}$时间内求解；而除非指数时间假设不成立，否则无法在 $2^{o(n/\log n)}$ 时间内求解。对于多重访问旅行商问题，[CP84] 提出的一个已有的 $n^{O(n)}$ 时间算法，近期被 [BKMV20]改进为 $2^{O(n)}$ 时间算法。

这一方向一个值得关注的开放问题是边染色问题的复杂性：它能否在 $2^{o(n^2)}$ 时间内求解？另见 [KM24]。

量子加速。一个相对较新的研究主题是量子算法能在多大程度上加速 NP 完全问题的精确算法。直接应用格罗弗搜索可知，k-合取范式可满足性问题可在 $2^{n/2}$ 时间内求解；而在 [BPS21] 中提出了假设 3.2 的量子类比，认为这一结果无法得到显著改进。对于其他问题，找到量子加速并不简单，但仍然可以做到（例如参见 [ABI+19]）。

致谢。

作者感谢编辑们耐心等待这篇逾期稿件，感谢 Carla Groenland 就第 4 节进行的有益讨论，感谢 László Kozma 对这篇综述的撰写提出改进建议，也感谢 IPL 的审稿人和编辑对第 3.2 小节的先前版本提出详尽意见。

参考文献

[ABHS22] Amir Abboud、Karl Bringmann、Danny Hermelin 和 Dvir Shabtay。基于 SETH 的子集和与双准则路径下界。*ACM* 算法汇刊，18(1):6:1– 6:22，2022年。[ABI+19] Andris Ambainis、Kaspars Balodis、Janis Iraids、Martins Kokainis、Krisjanis Prusis 和 Jevgenijs Vihrovs。指数时间动态规划算法的量子加速。载于 Timothy M. Chan（编），第*30*届*ACM-SIAM*离散算法年度研讨会论文集，*SODA 2019*，美国加利福尼亚州圣迭戈，*2019*年*1*月*6*日至*9*日，第1783–1793页。SIAM，2019年。[AS16] Noga Alon 和 Joel H Spencer。概率方法。John Wiley & Sons，2016年。[BCJ11] Anja Becker、Jean-Sébastien Coron 和 Antoine Joux。困难背包问题的改进通用算法。载于 Kenneth G. Paterson（编），密码学进展—*EURO-CRYPT 2011*：第*30*届密码技术理论与应用国际年度会议，爱沙尼亚塔林，*2011*年*5*月*15*日至*19*日，论文集，计算机科学讲义第6632卷，第364–385页。施普林格，2011年。[BCKM24] Tatiana Belova、Nikolai Chukhin、Alexander S. Kulikov 和 Ivan Mihajlin。子集和的改进空间界。载于 Timothy M. Chan 、 Johannes Fischer 、 John Iacono 和 Grzegorz Herman （编），第*32*届欧洲算法研讨会，*ESA 2024*，英国伦敦皇家霍洛威学院，*2024*年*9*月*2*日至*4*日，第308卷*LIPIcs*，第21:1–21:17页。达格斯图尔城堡—莱布尼茨信息学中心，2024年。

[BCKN15] Hans L. Bodlaender、Marek Cygan、Stefan Kratsch 和 Jesper Nederlof。针对以树宽为参数的连通性问题的确定性单指数时间算法。Inf. Comput.，243:86–111，2015年。[BE05] Richard Beigel 和 David Eppstein。用 $O(1.3289^n)$ 时间进行3-着色。《算法杂志》，54(2):168–204，2005年。[Bea94] Paul Beame。切换引理入门。技术报告，技术报告 UW-CSE-95-07-01，华盛顿大学计算机科学与工程系，1994年。[Bel62] Richard Bellman。动态规划处理旅行商问题。《ACM学报》，9(1):61–63，1962年。[BGK+23] Tatiana Belova、Alexander Golovnev、Alexander S. Kulikov、Ivan Mihajlin 和 Denil Sharipov。多项式公式构成基于归约的困难性证明的障碍。载于 Nikhil Bansal 和 Viswanath Nagarajan（编），2023年ACM-SIAM离散算法研讨会论文集，SODA 2023，意大利佛罗伦萨，2023年1月22日至25日，第3245–3281页。SIAM，2023年。[BGNV18] Nikhil Bansal、Shashwat Garg、Jesper Nederlof 和 Nikhil Vyas。子集和、k-和及相关问题的更快空间高效算法。SIAM计算杂志，47(5):1755–1777，2018年。[BHK09] Andreas Björklund、Thore Husfeldt 和 Mikko Koivisto。通过容斥实现集合划分。SIAM计算杂志，39(2):546–563，2009年。[BHKK10] Andreas Björklund、Thore Husfeldt、Petteri Kaski 和 Mikko Koivisto。截断的莫比乌斯反演与有界度图。Theory Comput. Syst.，47(3):637–654，2010年。[BHKK12] Andreas Björklund、Thore Husfeldt、Petteri Kaski 和 Mikko Koivisto。有界度图上的旅行商问题。ACM 算法汇刊，8(2):18:1–18:13，2012年。[BHKK17] Andreas Björklund、Thore Husfeldt、Petteri Kaski 和 Mikko Koivisto。用于参数化路径与打包问题的窄筛。《计算机与系统科学杂志》，87:119–139，2017年。[Bjö14] Andreas Björklund。无向哈密顿性的行列式和。SIAM计算杂志，43(1):280–299，2014年。[BK24] Andreas Björklund 和 Petteri Kaski。渐近秩猜想与集合覆盖猜想并非同时为真。载于 Bojan Mohar、Igor Shinkar 和 Ryan O’Donnell（编），第56届ACM计算理论年度研讨会论文集，STOC 2024，加拿大不列颠哥伦比亚省温哥华，2024年6月24日至28日，第859–870页。ACM，2024年。[BKK17] Andreas Björklund、Petteri Kaski 和 Ioannis Koutis。通过广义拉普拉斯矩阵处理有向哈密顿性与出分支。载于 Ioannis Chatzigiannakis、Piotr Indyk、Fabian Kuhn 和 Anca Muscholl（编），第44届国际自动机、语言与程序设计学术讨论会，ICALP 2017，2017年7月10日至14日，波兰华沙，第80卷 LIPIcs，第91:1–91:14页。达格斯图尔城堡—莱布尼茨信息学中心，2017年。

[BKM+24] Tatiana Belova、Alexander S. Kulikov、Ivan Mihajlin、Olga Ratseeva、Grigory Reznikov 和 Denil Sharipov。利用多项式求值预言机进行计算：排除基于 SETH 的超线性下界。载于 David P. Woodruff 编，*2024*年*ACM-SIAM*离散算法研讨会论文集，*SODA 2024*，美国弗吉尼亚州亚历山德里亚，*2024*年*1*月*7*日至*10*日，第1834–1853页。SIAM，2024年。[BKMV20] André Berger、László Kozma、Matthias Mnich 和 Roland Vincze。适用于多次访问旅行商问题的时间与空间最优算法。*ACM* 算法汇刊，16(3):35:1–35:22，2020年。[BNvdZ16] Hans L. Bodlaender、Jesper Nederlof 和 Tom C. van der Zanden。用于嵌入不含 h-小图的图的次指数时间算法。载于 Ioannis Chatzigiannakis、Michael Mitzenmacher、Yuval Rabani 和 Davide Sangiorgi 编，第*43*届自动机、语言与编程国际研讨会，*ICALP 2016*，*2016*年*7*月*11*日至*15*日，意大利罗马，第55卷，*LIPIcs*，第9:1–9:14页。达格斯图尔城堡—莱布尼茨信息学中心，2016年。[BPS21] Harry Buhrman、Subhasree Patro 和 Florian Speelman。量子强指数时间假设框架。载于 Markus Bläser 和 Benjamin Monmege 编，第*38*届理论计算机科学专题研讨会，*STACS 2021*，*2021*年*3*月*16*日至*19*日，德国萨尔布吕肯（线上会议），第187卷*LIPIcs*，第19:1–19:19页。达格斯图尔城堡—莱布尼茨信息学中心，2021年。[CDL+16] Marek Cygan、Holger Dell、Daniel Lokshtanov、Dániel Marx、Jesper Nederlof、Yoshio Okamoto、Ramamohan Paturi、Saket Saurabh 和 Magnus Wahlström。论与合取范式可满足性问题同样困难的问题。*ACM* 算法汇刊，12(3):41:1–41:24，2016年。[CFG+17] Marek Cygan、Fedor V. Fomin、Alexander Golovnev、Alexander S. Kulikov、Ivan Mihajlin、Jakub Pachocki 和 Arkadiusz Socala。图嵌入问题的紧下界。《*ACM*学报》，64(3):18:1–18:22，2017年。[CFK+15] Marek Cygan、Fedor V. Fomin、Lukasz Kowalik、Daniel Lokshtanov、Dániel Marx、Marcin Pilipczuk、Michal Pilipczuk 和 Saket Saurabh。参数化算法。施普林格，2015年。[CGI+16] 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影响。载于 Madhu Sudan 编，*2016*年*ACM*理论计算机科学创新会议论文集，美国马萨诸塞州剑桥，*2016*年*1*月*14*日至*16*日，第261–270页。ACM，2016年。[CIP06] Chris Calabro、Russell Impagliazzo 和 Ramamohan Paturi。可满足性问题中子句宽度与子句密度的对偶关系。载于第*21*届*IEEE*计算复杂性年度会议（*CCC 2006*），*2006*年*7*月*16*日至*20*日，捷克共和国布拉格，第252–260页。IEEE计算机学会，2006年。[CJRS22] Xi Chen、Yaonan Jin、Tim Randolph 和 Rocco A. Servedio。平均情形的子集平衡问题。载于 Joseph (Seffi) Naor 和 Niv Buchbinder 编，*2022*年*ACM-SIAM*离散算法研讨会论文集，*SODA 2022*，线上会议／美国弗吉尼亚州亚历山德里亚，*2022*年*1*月*9*日至*12*日，第743–778页。SIAM，2022年。

[CJWW22] Lijie Chen、Ce Jin、R. Ryan Williams 和 Hongxun Wu。通过伪随机哈希函数以极低空间求解元素互异性和子集和问题。载于 Joseph (Seffi) Naor 和 Niv Buchbinder 编，2022年ACM-SIAM离散算法研讨会论文集，SODA 2022，线上会议／美国弗吉尼亚州亚历山德里亚，2022年1月9日至12日，第1661–1678页。SIAM，2022年。[CKN18] Marek Cygan、Stefan Kratsch 和 Jesper Nederlof。通过完美匹配基快速判定哈密顿性。《ACM学报》，65(3):12:1–12:46，2018年。[CP84] Stavros S. Cosmadakis 和 Christos H. Papadimitriou。多次访问少数城市的旅行商问题。SIAM计算杂志，13(1):99–108，1984年。[DGH+02] Evgeny Dantsin、Andreas Goerdt、Edward A. Hirsch、Ravi Kannan、Jon M. Kleinberg、Christos H. Papadimitriou、Prabhakar Raghavan 和 Uwe Schöning。基于局部搜索的 k-SAT 确定性$O((2-2/(k+1))^n)$算法。《理论计算机科学》，289(1):69–83，2002年。[DL78] Richard A. DeMillo 和 Richard J. Lipton。关于代数程序测试的一则概率性评述。《信息处理快报》，7(4):193–195，1978年。[Dru13] Andrew Drucker。非确定性直接积归约与 SAT 求解器的成功概率。载于第54届IEEE计算机科学基础年会，FOCS 2013，2013年10月26日至29日，美国加利福尼亚州伯克利，第736–745页。IEEE计算机学会，2013年。[FG06] Jörg Flum 和 Martin Grohe。参数化复杂性理论。理论计算机科学丛书：EATCS丛书。施普林格，2006年。[FGK09] Fedor V. Fomin、Fabrizio Grandoni 和 Dieter Kratsch。分析精确算法的测度与征服方法。《ACM学报》，56(5):25:1–25:32，2009年。[FGLS19] Fedor V. Fomin、Serge Gaspers、Daniel Lokshtanov 和 Saket Saurabh。通过单调局部搜索设计精确算法。《ACM学报》，66(2):8:1–8:23，2019年。[FIK08] Fedor V. Fomin、Kazuo Iwama 和 Dieter Kratsch。08431执行摘要——中等指数时间算法。载于 Fedor V. Fomin、Kazuo Iwama 和 Dieter Kratsch 编，《中等指数时间算法》，第8431卷《达格施图尔研讨会论文集》（DagSemProc），第1–2页，德国达格施图尔，2008年。达格施图尔城堡—莱布尼茨信息学中心。[FK10] Fedor V. Fomin 和 Dieter Kratsch。《精确指数算法》。理论计算机科学丛书：EATCS丛书。施普林格，2010年。[FK13] Fedor V. Fomin 和 Petteri Kaski。精确指数算法。《ACM通讯》，56(3):80–88，2013年。[FT87] András Frank 和 Éva Tardos。在组合优化中对同时丢番图逼近的应用。《组合数学》，7(1):49–65，1987年。[GS87] Yuri Gurevich 和 Saharon Shelah。哈密顿路径问题的期望计算时间。SIAM计算杂志，16(3):486–502，1987年。

[Hås89] Johan Håstad。小深度电路的近乎最优下界。计算机科学进展，5:143–170，1989。[HJ10] Nick Howgrave-Graham 和 Antoine Joux。困难背包问题的新通用算法。载于 Henri Gilbert 编，密码学进展——第*29*届密码技术理论与应用国际年会，摩纳哥/法国里维埃拉，*2010*年*5*月*30*日至*6*月*3*日。论文集，计算机科学讲义第6110卷，第235–256页。施普林格，2010年。[HK62] Michael Held 和 Richard M Karp。排序问题的动态规划方法。《工业与应用数学学会杂志》，10(1):196–210，1962年。[HKPS11] Thore Husfeldt、Dieter Kratsch、Ramamohan Paturi 和 Gregory B. Sorkin。10441论文集摘要——NP难问题的精确复杂度。载于 Thore Husfeldt、Dieter Kratsch、Ramamohan Paturi 和 Gregory B. Sorkin 编，*NP*难问题的精确复杂度，《达格施图尔研讨会论文集（*DagSemProc*）》第10441卷，第1–22页，德国达格施图尔，2011年。达格施图尔城堡—莱布尼茨信息学中心。[HKZZ19] Thomas Dueholm Hansen、Haim Kaplan、Or Zamir 和 Uri Zwick。使用 biased-ppsz 的更快*k*-SAT算法。载于 Moses Charikar 和 Edith Cohen 编，第*51*届*ACM SIGACT*计算理论研讨会论文集，*STOC 2019*，美国亚利桑那州菲尼克斯，*2019*年*6*月*23*日至*26*日，第578–589页。ACM，2019年。[HPSW13] Thore Husfeldt、Ramamohan Paturi、Gregory B. Sorkin 和 Ryan Williams。指数算法：多项式时间之外的复杂度（达格施图尔研讨会13331）。《达格施图尔报告》，3(8):40–72，2013年。[HS74] Ellis Horowitz 和 Sartaj Sahni。计算划分及其在背包问题中的应用。《*ACM*学报》，21(2):277–292，1974年。[IMP11] Russell Impagliazzo、William Matthews 和 Ramamohan Paturi。一个用于 AC^0的可满足性算法。*CoRR*，abs/1107.3127，2011年。[IP01] Russell Impagliazzo 和 Ramamohan Paturi。关于 k-SAT 的复杂性。《计算机系统科学杂志》，62(2):367–375，2001年。[IPZ98] Russell Impagliazzo、Ramamohan Paturi 和 Francis Zane。哪些问题具有强指数复杂度？载于<style id='46'>第39届年度计算机科学基础研讨会，FOCS ’98，1998年11月8日至11日，美国加利福尼亚州帕洛阿尔托，第653–663页。IEEE计算机学会，1998年。[JS82] Mark Jerrum 和 Marc Snir。半环上直线型计算的一些精确复杂度结果。《*ACM*学报》，29(3):874–897，1982年。[JS99] David S. Johnson 和 Mario Szegedy。最大独立集问题中最难处理的实例是什么？载于 Robert Endre Tarjan 和 Tandy J. Warnow 编，第十届*ACM-SIAM*离散算法研讨会论文集，*1999*年*1*月*17*日至*19*日，美国马里兰州巴尔的摩，第927–928页。ACM/SIAM，1999年。

[JSS21] Vishesh Jain、Ashwin Sah 和 Mehtaab Sawhney。反集中与子集和数量之比较。《组合学进展》，6，2021年。[Juk23] Stasys Jukna。热带电路复杂性：纯动态规划的局限。Springer Nature，2023年。[JW24] Ce Jin 和 Hongxun Wu。鸽巢等和的更快算法。CoRR，abs/2403.19117，2024年。[Kas18] Petteri Kaski。为小子图计数设计一种可委托且容错的算法。载于第二十届算法工程与实验研讨会，ALENEX 2018，美国路易斯安那州新奥尔良，2018年1月7日至8日，第184–198页，2018年。[KM24] Alexander S. Kulikov 和 Ivan Mihajlin。如果边染色在 SETH 假设下是困难的，那么 SETH 为假。载于 Merav Parter 和 Seth Pettie 编，2024年算法简洁性研讨会，SOSA 2024，美国弗吉尼亚州亚历山大，2024年1月8日至10日，第115–120页。SIAM，2024年。[Koi09] Mikko Koivisto。划分为有界基数的集合。载于 Jianer Chen 和 Fedor V. Fomin 编，《参数化与精确计算》，第4届国际研讨会，IWPEC 2009，丹麦哥本哈根，2009年9月10日至11日，修订版精选论文，计算机科学讲义第5917卷，第258–263页。施普林格，2009年。[Kou08] Ioannis Koutis。路径和打包问题的更快代数算法。载于 Luca Aceto、Ivan Damgård、Leslie Ann Goldberg、Magnús M. Halldórsson、Anna Ingólfsdóttir 和 Igor Walukiewicz 编，《自动机、语言与编程》，第35届国际学术会议，ICALP 2008，冰岛雷克雅未克，2008年7月7日至11日，论文集，第一部分：主题A：算法、自动机、复杂性与博弈，计算机科学讲义第5125卷，第575–586页。施普林格，2008年。[KW16] Ioannis Koutis 和 Ryan Williams。用于更快算法的代数指纹。《ACM通讯》，59(1):98–105，2016年。[LMS18] Daniel Lokshtanov、Dániel Marx 和 Saket Saurabh。有界树宽图上的已知算法可能已达到最优。ACM 算法汇刊，14(2):13:1–13:30，2018年。[LN22] Jason Li 和 Jesper Nederlof。在 $O^\star(2.7k)$ 时间内检测大小为 k 的反馈顶点集。ACM 算法汇刊，18(4):34:1–34:26，2022年。[Lov79] László Lovász。关于行列式、匹配与随机算法。载于 FCT，第79卷，第565–574页，1979年。[LSX26] Daniel Lokshtanov、Saket Saurabh 和 Jie Xue。通过测度与征服证明稀疏化引理。载于 Sepehr Assadi 和 Eva Rotenberg 编，2026年算法简洁性研讨会，SOSA 2026，温哥华。SIAM，2026年。[LZ23] Xin Lyu 和 Weihao Zhu。通过伪随机性实现元素互异性与集合交的时间-空间权衡。载于 Nikhil Bansal 和 Viswanath Nagarajan 编，

*2023*年*ACM-SIAM*离散算法研讨会论文集，*SODA 2023*，意大利佛罗伦萨，*2023*年*1*月*22*日至*25*日，第5243–5281页。SIAM，2023年。[MNPW19] Marcin Mucha、Jesper Nederlof、Jakub Pawlewicz和Karol Wegrzycki。通过中间相遇法加速等和子集问题。载于Michael A. Bender、Ola Svensson和Grzegorz Herman编，第*27*届欧洲算法年度研讨会，*ESA 2019*，*2019*年*9*月*9*日至*11*日，德国慕尼黑／加兴，LIPIcs第144卷，第73:1–73:16页。达格斯图尔城堡—莱布尼茨信息学中心，2019年。[MS85] Burkhard Monien和Ewald Speckenmeyer。用少于 2^n 步求解可满足性问题。《离散应用数学》，10(3):287–295，1985年。[MVV87] Ketan Mulmuley、Umesh V. Vazirani和Vijay V. Vazirani。匹配问题与矩阵求逆一样容易。《组合学》，7(1):105–113，1987年。[Ned16] Jesper Nederlof。通过表示方法更快地寻找大型集合覆盖。载于Piotr Sankowski和Christos D. Zaroliagis编，第*24*届欧洲算法年度研讨会，*ESA 2016*，*2016*年*8*月*22*日至*24*日，丹麦奥胡斯，LIPIcs第57卷，第69:1–69:15页。达格斯图尔城堡—莱布尼茨信息学中心，2016年。[Ned20] Jesper Nederlof。在假设矩阵乘法为平方时间的前提下，二分图旅行商问题可在 $O(1.9999^n)$时间内求解。载于Konstantin Makarychev、Yury Makarychev、Madhur Tulsiani、Gautam Kamath和Julia Chuzhoy编，第*52*届*ACM SIGACT*计算理论年度研讨会论文集，*STOC 2020*，美国伊利诺伊州芝加哥，*2020*年*6*月*22*日至*26*日，第40–53页。ACM，2020年。[NPSW23] Jesper Nederlof、Jakub Pawlewicz、Céline M. F. Swennenhuis和Karol Wegrzycki。通过加法组合学求解固定箱数装箱问题的更快指数时间算法。《*SIAM*计算杂志》，52(6):1369–1412，2023年。[NvLvdZ12] Jesper Nederlof、Erik Jan van Leeuwen和Ruben van der Zwaan。将目标区间缩减为少量精确查询。载于Branislav Rovan、Vladimiro Sassone和Peter Widmayer编，*2012*年计算机科学数学基础——第*37*届国际研讨会，*MFCS 2012*，斯洛伐克布拉迪斯拉发，*2012*年*8*月*27*日至*31*日，论文集，计算机科学讲义第7464卷，第718–727页。施普林格，2012年。[NW21] Jesper Nederlof和Karol Wegrzycki。通过正交向量改进Schroeppel和Shamir算法以求解子集和问题。载于Samir Khuller和Virginia Vassilevska Williams编，*STOC '21*：第*53*届*ACM SIGACT*计算理论年度研讨会，线上活动，意大利，*2021*年*6*月*21*日至*25*日，第1670–1683页。ACM，2021年。[PP10] Ramamohan Paturi和Pavel Pudlák。论电路可满足性的复杂性。载于Leonard J. Schulman编，第*42*届*ACM*计算理论研讨会论文集，*STOC 2010*，美国马萨诸塞州剑桥，*2010*年*6*月*5*日至*8*日，第241–250页。ACM，2010年。[Pra24] Kevin Pratt。渐近秩猜想与集合覆盖猜想之间更强的联系。载于Bojan Mohar、Igor Shinkar和Ryan O'Donnell编，第*56*届*ACM*计算理论年度研讨会论文集，*STOC 2024*，加拿大不列颠哥伦比亚省温哥华，*2024*年*6*月*24*日至*28*日，第871–874页。ACM，2024年。

[Sap07] Aleksandr Antonovich Sapozhenko。图中独立集的数量。《莫斯科大学数学通报》，62(3):116–118，2007年。[Sch80] Jacob T. Schwartz。用于验证多项式恒等式的快速概率算法。《ACM学报》，27(4):701–717，1980年。[Sch05a] Uwe Schöning。指数时间算法。载于Volker Diekert和Bruno Durand编，《2005年理论计算机科学专题研讨会》，第22届理论计算机科学专题研讨会，德国斯图加特，2005年2月24日至26日，论文集，计算机科学讲义第3404卷，第36–43页。施普林格，2005年。[Sch05b] Alexander Schrijver。组合优化史（截至1960年）。载于K. Aardal、G. L. 内姆豪瑟和R. 魏斯曼特尔编，《离散优化》，《运筹学与管理科学手册》第12卷，第1–68页。爱思唯尔，2005年。[Sch24] Dominik Scheder。PPSZ比你想象的更好。TheoretiCS，3，2024年。[SS81] Richard Schroeppel和阿迪·沙米尔。一种用于求解某些NP完全问题的 $T = O(2^{n/2})$、 $S = O(2^{n/4})$时间算法。《SIAM计算杂志》，10(3):456–464，1981年。[SS12] 拉胡尔·桑塔南和斯里坎特·斯里尼瓦桑。论稀疏化的局限性。载于阿图尔·楚马伊、库尔特·梅尔霍恩、安德鲁·M·皮茨和罗杰·瓦滕霍弗编，《自动机、语言与编程——第39届国际学术研讨会，ICALP 2012，英国华威，2012年7月9日至13日，论文集》，第一部分，计算机科学讲义第7391卷，第774–785页。施普林格，2012年。[Str94] 福尔克·施特拉森。代数与复杂性。载于《第一届欧洲数学大会，巴黎，1992年7月6日至10日：第二卷：特邀报告（第二部分）》，第429–446页。施普林格，1994年。[Tra84] 鲍里斯·A·特拉赫滕布罗特。俄罗斯穷举搜索算法方法综述。《IEEE计算机史年刊》，6(4):384–400，1984年。[Tut47] 威廉·T·塔特。线性图的因子分解。《伦敦数学学会期刊》，s1-22(2):107–111，1947年。[Wil14] Ryan Williams。电路的算法与算法的电路：连接可处理与不可处理问题。载于《国际数学家大会论文集》，第659–682页，2014年。[Wil16] Richard Ryan Williams。强指数时间假设在梅林与亚瑟模型下不成立：批量求值的简短非交互式证明。载于兰·拉兹编，《第31届计算复杂性会议，CCC 2016》，东京，日本，2016年5月29日至6月1日，LIPIcs第50卷，第2:1–2:17页。达格斯图尔城堡—莱布尼茨信息学中心，2016年。[Woe01] Gerhard J. Woeginger。NP困难问题的精确算法：综述。载于米夏埃尔·容格、格哈德·赖内尔特和乔瓦尼·里纳尔迪编，《组合优化——尤里卡，你缩小了！》，献给杰克·埃德蒙兹的论文，第5届国际研讨会，法国奥索瓦，2001年3月5日至9日，修订版论文集，计算机科学讲义第2570卷，第185–208页。施普林格，2001年。

[Woe04] Gerhard J. Woeginger. 精确算法的空间与时间复杂性：一些未解问题（特邀报告）。载于 Rodney G. Downey、Michael R. Fellows 和 Frank K. H. A. Dehne 编，参数化与精确计算：第一届国际研讨会 *IWPEC 2004*，挪威卑尔根，*2004*年*9*月*14*日至*17*日，论文集，计算机科学讲义第3162卷，第281–290页。施普林格，2004年。[Woe08] Gerhard J. Woeginger. 关于精确算法的未解问题。离散应用数学，156(3):397–405，2008年。[Yat37] Franck Yates.《析因试验的设计与分析》。帝国土壤科学局。技术通讯。帝国土壤科学局，1937年。[Zam21] Or Zamir. 突破5着色和6着色的 2^n 障碍。载于 Nikhil Bansal、Emanuela Merelli 和 James Worrell 编，第*48*届国际自动机、语言与程序设计研讨会，*ICALP 2021*，苏格兰格拉斯哥（线上会议），*2021*年*7*月*12*日至*16*日，LIPIcs第198卷，第113:1–113:20页。达格斯图尔城堡—莱布尼茨信息学中心，2021年。[Zam23] Or Zamir. 超图与划分容器的算法应用。载于 Barna Saha 和 Rocco A. Servedio 编，第*55*届年度 *ACM*计算理论研讨会论文集，*STOC 2023*，美国佛罗里达州奥兰多，*2023*年*6*月*20*日至*23*日，第985–998页。ACM，2023年。[Zip93] Richard E. Zippel. 有效多项式计算，克鲁维尔工程与计算机科学国际丛书第241卷。克鲁维尔，1993年。