以团宽为参数的连通性问题的紧致算法

FalkoHegerfeld ！ 德国柏林洪堡大学**Stefan Kratsch** 德国柏林洪堡大学

arXiv:2302.03627v1 [cs.DS] 7 Feb 2023

摘要

涉及全局约束的问题，其复杂度通常比仅涉及局部约束的问题更难理解。在图问题领域，连通性约束是一种自然的全局约束形式。我们从细粒度参数化视角研究连通性问题。在一项突破性成果中，Cygan 等人（TALG 2022）通过引入割与计数技术，首次获得了运行时间为单指数的算法 $\alpha^{\mathrm{tw}}n^{\mathcal{O}(1)}$，用于求解以树宽为参数的连通性问题；该技术将许多连通性问题归约为局部可检验计数问题。此外，在强指数时间假设（强指数时间假设）下，已证明这些算法的底数是最优的 α 。

然而，由于只有稀疏图可能具有较小的树宽，我们对稠密结构方面连通性问题的细粒度复杂度知之甚少。衡量稠密结构最常用的图参数或许是团宽，直观而言，它衡量了通过反复添加完全二部图来构造一个图的难易程度。Bergougnoux和Kanté（TCS 2019）利用基于秩的方法证明，以团宽为参数时，许多连通性问题也存在单指数算法。遗憾的是，在强指数时间假设下，这些算法的运行时间远非最优。

我们展示了如何针对团宽，为两个基准连通性问题——连通顶点覆盖和连通支配集——获得最优运行时间
针对团宽的连通性问题，这是首批紧致结果；这些结果通过开发基于割与计数技术的新算法以及新颖的下界构造获得。具体而言，我们证明存在单边错误蒙特卡罗算法，对于给定的 k-团表达式，可以在连通顶点覆盖 时间内求解 $6^k n^{\mathcal{O}(1)}$，并且可以在连通支配集 时间内求解 $5^k n^{\mathcal{O}(1)}$。根据强指数时间假设，这两个结果都被证明是紧致的。

2012年美国计算机协会主题分类 计算数学 → 路径与连通性问题；计算理论 → 参数化复杂性与精确算法；计算数学→组合算法

关键词和短语 参数化复杂性、连通性、团宽、Cut&Count、下界

资助 *Falko Hegerfeld*：部分获得DFG Emmy Noether资助（KR 4286/1）。

目录

1 引言 2

2 技术概述 **5**
2.1 算法技术.5 2.2 下界方法.6

3 预备知识 **7**
3.1 切割与计数.9 3.2 规范团表达式.10

4 动态规划算法 14
4.1 连通顶点覆盖.14 4.2 连通支配集.18

5 下界 25
5.1 连通顶点覆盖.26 5.1.1 路径构件.26 5.1.2 完整构造.32 5.2 连通支配集.36 5.2.1 路径构件.37 5.2.2 完整构造.42

6 结论与开放问题 46

A 快速卷积算法 50
A.1 修剪子集卷积.50 A.2 基于格的卷积.55

1 引言

应对问题NP困难性的一种方法是参数化复杂性理论：我们试图比最坏情况实例更快地求解具有结构的实例；此外，还引入一个参数来量化实例的结构程度。理想情况下，我们可以得到运行时间为$\mathcal{O}^*(f(k))$[1]，的固定参数可解算法，其中 k 是参数， f 是某个可计算函数。在确立了这类算法存在之后，自然的下一步是采取精细化视角，确定针对这一问题与参数组合的最小可能函数 f，从而量化所考察结构对问题复杂性的确切影响。

我们从细粒度参数化视角研究连通性问题。这一研究方向始于 Cygan 等人 [19] 的突破性成果：他们首次通过引入割与计数技术，将连通性问题归约为局部可检验计数问题，从而获得了运行时间为 $\mathcal{O}^*(\alpha^{\mathrm{tw}})$ 的算法，其中 $base$ $\alpha > 1$ 为某个常数，适用于以树宽(tw)为参数的连通性问题。此外，在强指数时间假设（SETH） [17]成立的前提下，已证明这些算法的底数 α 是最优的。

[1] $\mathcal{O}^*$ 符号隐藏了输入规模中的多项式因子。

由于只有稀疏图可能具有较小的树宽，我们对连通性问题在稠密结构下的精确复杂度仍知之甚少。在稠密图这一情形中，团宽 是最常用的参数之一。Bergougnoux [2] 将割与计数应用于若干基于结构化邻域的宽度参数，其中包括团宽。此外，Bergougnoux和 Kanté [4], 基于 Bodlaender 等人 [9], 提出的基于秩的方法，为一大类以团宽为参数的连通性问题获得了$\mathcal{O}^*(\alpha^{\mathrm{cw}})$ 的单指数运行时间。由于这两篇文章都旨在为一大类问题构造大量单指数算法，因此对于连通（共）(σ, ρ)支配集问题，特定问题所获得的底数远非最优。例如，前一篇文章给出了一个求解连通支配集的 $\mathcal{O}^*(128^{\mathrm{cw}})$ 时间算法；后一篇文章则给出了一个求解连通顶点覆盖的 $\mathcal{O}^*((27\cdot 2^{\omega+1})^{\mathrm{cw}})$ 时间算法，以及一个求解连通支配集的 $\mathcal{O}^*((8\cdot 2^{\omega+1})^{\mathrm{cw}})$ 时间算法，其中 ω 为矩阵乘法指数，参见 Alman 和 Vassilevska W. [1]。即使 $\omega = 2$, 这也只会得到分别高达 216 和 64 的底数。

我们通过提出新算法，展示了如何显著优化以团宽为参数的连通顶点覆盖和连通支配集问题的运行时间。这些更快的算法同样依赖割与计数技术，并通过精确分析哪些割与计数状态是必要的来进行优化。此外，我们还运用快速子集卷积、容斥状态，以及区分活标签和死标签等技术，以获得更优的运行时间。

▶**定理 1.1**。 存在单边错误蒙特卡罗算法，给定图 k-表达式[2] G，可以在连通顶点覆盖 问题上用时 $\mathcal{O}^*(6^k)$，连通支配集 问题上用时 $\mathcal{O}^*(5^k)$。

我们通过证明所得运行时间在强指数时间假设（SETH）下是最优的，表明这些算法对于相应的问题—参数组合而言基本上是正确的。为证明这些下界，我们沿用 Lokshtanov 等人提出的、如今已成为标准的针对宽度参数的下界构造原则。为了将这一原则应用于团宽，我们仔细研究了问题在联接过程中的行为；也就是说，研究团宽定义所依据的边结构。这项研究的结果为我们设计合适的构件提供了重要指导。具体而言，我们得到了以下紧确下界：

▶ **定理 1.2.** 假设强指数时间假设成立，则对所有 $\varepsilon > 0$: 连通顶点覆盖 无法在 $\mathcal{O}^*((6-\varepsilon)^{\mathrm{cw}})$的时间内求解。连通支配集 无法在 $\mathcal{O}^*((5-\varepsilon)^{\mathrm{cw}})$的时间内求解。

这项工作属于一个更大研究计划的一部分，旨在确定若干连通性问题相对于多种宽度参数的最优运行时间。这些参数从限制性较强到一般性越来越强，因此有助于细致理解此情形下一般性的代价。我们在表1中总结了已知结果。Cygan 等人 [18, 19] 提出的割与计数技术，结合他们给出的下界，确定了许多连通性问题相对于树宽（和路径宽度）的复杂度。Bojikian 等人 [10]考察了限制性更强的割宽，并将割与计数和基于秩的方法相结合，从而改进树宽算法；若不存在改进算法，则给出割宽较低、代价更低的下界构造。本工作以及一

[2] 一个 k-表达式证明了 G 的团宽至多为 k。

参数	割宽	树宽	模树宽	团宽
连通顶点覆盖	$\mathcal{O}^*(2^k)$	$\mathcal{O}^*(3^k)$	$\mathcal{O}^*(5^k)$	$\mathcal{O}^*(6^k)$
连通支配集	$\mathcal{O}^*(3^k)$	$\mathcal{O}^*(4^k)$	$\mathcal{O}^*(4^k)$	$\mathcal{O}^*(5^k)$
斯坦纳树	$\mathcal{O}^*(3^k)$	$\mathcal{O}^*(3^k)$	$\mathcal{O}^*(3^k)$	?
反馈顶点集	$\mathcal{O}^*(2^k)$	$\mathcal{O}^*(3^k)$	$\mathcal{O}^*(5^k)$	?
参考文献	[10]	[18, 19]	[28]	here

表1 若干连通性问题相对于多种宽度参数的最优运行时间，这些参数按一般性递增排列。最后一列中的结果由本文得出。在模树宽与团宽之间，我们仅知关系 $\mathrm{cw}(G) \leq 3 \cdot 2^{\text{mod-tw}(G)-1}$，但对于限制性更强的模路径宽度，同样的结果也是紧的；Hegerfeld 和 Kratsch [28]证明 $\mathrm{cw}(G) \leq \text{mod-pw}(G) + 2$。“?”标记的是问题与参数的组合：Bergougnoux 和 Kanté [4], 已知存在具有单指数运行时间的算法，但下界与算法之间仍存在差距。

伴随论文 [28] 给出了首批针对以稠密宽度参数为参数的连通性问题的细粒度参数化结果。伴随论文 [28] 研究参数模树宽，它通过结合树分解与模分解，将树宽拓展到稠密情形，因此是树宽与团宽之间自然的中间步骤。关于模树宽的算法结果，要么在运行时间中的底数保持不变时直接归约到树宽情形，要么应用割与计数技术和模结构，实质上将问题归约为一个更复杂的、以树宽为参数的问题；在后一种情况下，论文还给出了新的下界构造，其遵循与本文类似的高层原则，但必须遵守不同的设计限制。Cygan 等人 [19] 指出，在稀疏情形中，施加连通性约束至多会使底数增加 1。例如，顶点覆盖的最优底数为 2，见 Lokshtanov 等人 [37], ；而以树宽为参数时，连通顶点覆盖的最优底数为 3。在稠密情形中，连通性约束的影响可能更大。例如，以团宽为参数时，顶点覆盖和支配集的最优底数分别为 2 和 4 [31, 34], ；加入连通性约束后，二者分别增至 6 和 5。

其他相关工作。除了这些紧确结果之外，割与计数技术也已应用于分支宽 [44]和树深度 [26, 40]等参数。由于依赖隔离引理，割与计数技术会产生随机化算法。Bodlaender 等人 [9] 提出的基于秩的方法，以及 Fomin 等人 [22, 23] 提出的基于拟阵的技术，克服了这一缺陷，但代价是运行时间更长；此外，基于秩的方法也适用于不含连通性约束的问题。通过结合基于秩的方法与其他技术，避免诉诸高斯消元，在某些情况下可以实现最优运行时间，例如以路径宽度 [14, 16], 为参数的哈密顿回路问题，或以割宽 [25, 33]为参数的着色问题。基于秩的方法还可进一步用于相对于稠密宽度参数（如秩宽 [5] 和 mim-宽 [3]）的连通性问题。我们也推荐参阅 Nederlof 关于基于秩的方法的综述 [39]。

从连通性问题转向其他方向，我们综述一些针对稠密参数、能够获得紧确细粒度参数化算法的文献。Iwata 和 Yoshida 证明，对于任意 $\varepsilon > 0$ 顶点覆盖，若且唯若顶点覆盖可在时间 $\mathcal{O}^*((2-\varepsilon)^{\mathrm{tw}})$ 内求解，则它也可在时间 $\mathcal{O}^*((2-\varepsilon)^{\mathrm{cw}})$ 内求解 [31]；由于在我们的情形中，树宽与团宽对应的底数不同，因此似乎很难将他们的技术迁移到我们的设定中。Lampis [36] 为 q-着色问题获得了紧确的运行时间 $\mathcal{O}^*((2^q-2)^{\mathrm{cw}})$，并得到了一个紧确结果

对于以一种限制性更强的模树宽变体为参数的 q-着色问题也是如此。将结果推广到同态问题时，Ganian 等人 [24] 针对以团宽为参数的情形获得了紧确结果，其中所得底数取决于目标图的一种特殊度量。Katsikarelis 等人 [34] 针对以团宽为参数的 $(k,\ r)$-中心问题获得了紧确结果，特别是为支配集问题得到了紧确的运行时间 $\mathcal{O}^*(4^{\mathrm{cw}})$。Jacob 等人 [32]以及 Hegerfeld 和 Kratsch [27] 证明，奇圈横断集的运行时间 $\mathcal{O}^*(4^{\mathrm{cw}})$ 是紧确的；后者的论文还研究了推广到更多颜色的情形，并给出了非宽度参数的紧确结果。

2 技术概述

本节概述用于证明定理 1.1 和定理 1.2 的技术。

2.1 算法技术

切割与计数。 Cygan 等人的割与计数技术 [19] 使我们能够将连通性约束归约为一个可局部检查的计数问题。图的一个一致割 $G = (V, E)$是将顶点 V划分为两个部分（V_L, V_R）的有序划分，且没有任何边 E 跨越这两个割侧 V_L 和 V_R。一致割的关键性质是，G 恰好有 $2^{\mathrm{cc}(G)}$ 个不同的一致割，其中 $\mathrm{cc}(G)$ 表示 G的连通分量数。固定一个顶点 v_*，并且只考虑满足 $v_* \in V_L$的一致割（V_L, V_R），这个数就会降为 $2^{\mathrm{cc}(G)-1}$，因此，当且仅当 G 连通时，G 才拥有奇数个此类一致割。于是，如果我们对满足 $v_* \in X_L$的 $G[X]$ 的部分解 $v_* \in X \subseteq V$与一致割（X_L, X_R）组成的对（X,(X_L, X_R)）进行模二计数，那么只有连通解能够保留下来。当存在多个连通解时，这可能导致不希望出现的模二抵消，但借助隔离引理 [38]，可以通过随机化来避免这一问题。

将顶点状态提升为标签状态。 对于沿团表达式进行动态规划，我们必须刻画部分解与标签之间的相关交互，因为标签决定了表达式可以构造哪些联接。在所考虑的问题中，对于给定的部分解，单个顶点 v 可以处于常数种不同状态之一，我们用依赖于具体问题的集合 Ω 表示这些状态；例如，对于连通顶点覆盖，$\Omega = \{\mathbf{0}, \mathbf{1}_L, \mathbf{1}_R\}$，分别表示 $v \notin X$（状态 $\mathbf{0}$）、$v \in X_L$（状态 $\mathbf{1}_L$）和 $v \in X_R$（状态$\mathbf{1}_R$）。团表达式会反复在两组顶点之间添加联接，比如 A 和 B，也就是说，在 A 和 B 之间添加所有可能的边。添加联接后，算法必须检查部分解是否仍然可行，并可能更新某些状态。原则上，标签中顶点状态的每种取值都可能导致部分解呈现不同的行为。然而，对于所考虑的问题，关键的观察是：在 A 或 B 中，某种顶点状态出现的确切次数与联接无关；只需区分每一侧出现了哪些顶点状态、哪些没有出现即可。因此，相关的标签状态可用 Ω 的子集表示。接下来的两种技巧将进一步减少需要考虑的状态数量。

规范团表达式。 对于这两种算法，我们对标准团表达式进行细化和扩充，以区分活动标签和失活标签。在沿团表达式执行动态规划时，我们考虑由给定团表达式的子表达式所定义的诱导子图。在某个子表达式处，如果在剩余表达式中，标签为 ℓ 的顶点会接收当前子表达式中尚未出现的额外边，我们就称该标签为活动；

否则，我们称 ℓ 为失活。首先，我们注意到，无需跟踪部分解在失活标签处的状态，因为这些标签与剩余表达式中的其他状态只有平凡的交互。因此，我们只需考虑活动标签上可达到的状态，从而减少连通顶点覆盖所需考虑的状态数。为简化算法描述并避免处理边界情况，我们对团表达式进行变换，使其不出现退化情况，并添加一个死亡操作 $\perp_\ell$，用于处理标签 ℓ 从活动转为失活的情况。死亡操作类似于优美树分解 [35]中的规范树分解遗忘顶点节点。此前已有研究通过区分活动标签和失活标签来改善运行时间 [24, 34, 36]，但据作者所知，通过额外操作显式处理标签类型尚属首次。

容斥状态。对于连通支配集，我们将其转换为另一组顶点状态，称为容斥状态；此前已有研究证明，这些状态对支配问题很有帮助 [26, 41, 43, 45]。使用这些状态时，我们不再追踪某个顶点是否未被支配 或被部分解支配，而是允许 该顶点被支配，或禁止 它被支配。通常可以通过容斥论证恢复原问题的解；然而，将其提升到标签状态时，这一论证并不能直接沿用。我们证明，在模二计数下，该论证可以调整后用于标签状态；而对于顶点状态，已知这一论证也适用于非模计数。容斥状态的优势在于，在连接操作中，我们无需将顶点状态从未被支配更新为被支配，从而简化算法，并使我们能够将若干标签状态合并为一个。良好团表达式中的死亡操作为算法提供了合适的时机，以应用调整后的容斥论证。

快速卷积。为了快速计算动态规划递推式，我们采用快速子集卷积算法。在附录A中，我们对 Björklund 等人 [7] 针对截断子集卷积提出的技术进行调整，从而为连通顶点覆盖算法中出现的并集递推式获得一种快速算法。对于连通支配集，Björklund 等人 [8] 基于格的结果提供了快速计算并集递推式所需的方法。在这两种情况下，我们都得到了可应用于更一般情形的快速卷积算法，因此这些结果可能具有独立的研究价值。

2.2 下界方法

类网格构造。这两个下界都基于 Lokshtanov 等人 [37]在路径宽度/树宽参数化的强指数时间假设下界中采用的高层构造原则。所得图可以解释为由块组成的网格/矩阵，其中每个块跨越若干行和列。每一行都是一条长类路径构件，用于模拟可满足性实例中的常数个变量，并贡献一个团宽单位。我们能够在每行模拟的变量越多，就能排除越高的运行时间。每一列表示一个子句，由用于解码路径构件状态并验证所得赋值是否满足相应子句的构件组成。

路径构件与状态转移。我们的主要技术贡献是设计出位于每一行和每一列交汇处的路径构件，而对

解码构件则可以根据 Cygan 等人 [18] 已知的构造进行调整。由于每一行都应贡献一个团宽单位元，因此同一行中相邻的路径构件必须通过连接相连。我们的目标是在这些限制条件下设计一种路径构件，使其能够容纳尽可能多的不同状态。一个重要问题是，路径构件的状态如何沿着每一行进行转移，因为该归约只有在状态转移遵循某种转移顺序时才有效。

确定转移顺序。 由于可能的标签状态数量可能很大，因此如何选择合适的转移顺序并非显而易见。对于以路径宽度/树宽为参数的下界，这个问题要小得多，因为可能的状态数量少得多。因此，我们系统地分析连接处可能发生的状态转移，得到一个转移/兼容性 矩阵，展示哪些状态对能够产生全局可行解，哪些不能。对兼容性矩阵的行和列重新排序后，可能的转移顺序必须形成一个三角子矩阵。由兼容性矩阵中可能存在的最大三角子矩阵，我们便可以推导出合适的转移顺序，以指导路径构件的设计。

路径构件的结构。 我们的路径构件由三个部分组成：一个中心团，用于与解码构件通信；以及两个边界 部分，即左 部和右部，分别连接到前一个和后一个连接处。在中心团中，每个解都会恰好避开一个代表路径构件状态的顶点。为了实现转移顺序，左部和右部必须向两个相邻的路径构件传递适当的状态。通过取三角子矩阵并将主对角线上的状态配对，我们可以看出在每种情况下必须传递哪些状态。设计左部和右部的核心思路，是分离边界顶点的各项状态属性，例如它们是否属于部分解，或者是否被支配。这样便能与中心团进行简单通信，并加快转移顺序的实现。

3 预备知识

对于两个整数 a、b，我们用 $a \equiv_c b$ 表示它们模 $c \in \mathbb{N}$ 同余。我们使用艾弗森括号记号：对于布尔谓词 p，若 p 为真，则 $[p]$ 为 1；否则为 0。对于函数 f，我们用 $f[v \mapsto \alpha]$ 表示函数 $(f \setminus \{(v, f(v))\}) \cup \{(v, \alpha)\}$，其中将 f 视为集合；我们也用 $f[v \mapsto \alpha, w \mapsto \beta]$ 代替 $(f[v \mapsto \alpha])[w \mapsto \beta]$。我们用 $\mathbb{Z}_2$ 表示二元域。对于 n_1、$n_2 \in \mathbb{Z}$，我们写作 $[n_1, n_2] = \{x \in \mathbb{Z} : n_1 \leq x \leq n_2\}$ 和 $[n_2] = [1, n_2]$。对于函数 $f\colon V \to \mathbb{Z}$ 以及子集 $W \subseteq V$，我们写作 $f(W) = \sum_{v \in W} f(v)$。注意，对于函数 $g\colon A \to B$（其中 $B \not\subseteq \mathbb{Z}$）以及子集 $A' \subseteq B$，我们仍将 A' 在 g 下的像记为 $g(A') = \{g(v) : v \in A'\}$。若 $f\colon A \to B$ 是函数且 $A' \subseteq A$，则 $f|_{A'}$ 表示 f 在 A' 上的限制；对于子集 $B' \subseteq B$，我们将 B' 在 f 下的原像记为 $f^{-1}(B') = \{a \in A : f(a) \in B'\}$。集合的有序元组 $(A_1, \ldots, A_\ell)$ 是有序子划分，当且仅当对所有 $i \neq j \in [\ell]$ 都有 $A_i \cap A_j = \emptyset$。

图论符号

我们使用常见的图论记号，并假设读者了解参数化复杂性的基本知识。设 $G = (V, E)$ 是一个无向图。对于顶点集 $X \subseteq V$，我们用 $G[X]$ 表示由 X诱导的 G 的子图。顶点 v的开邻域 定义为 $N(v) = \{u \in V : \{u, v\} \in E\}$，而闭邻域 定义为

$N[v] = N(v) \cup \{v\}$。对于集合 $X \subseteq V$，我们定义 $N[X] = \bigcup_{v \in X} N[v]$和 $N(X) = N[X] \setminus X$。对于两个不相交的顶点子集 A、$B \subseteq V$，我们定义 $E_G(A, B) = \{\{a, b\} \in E(G) : a \in A, b \in B\}$；在 A 和 B 之间添加连接，意味着在 A 中的每个顶点与 B中的每个顶点之间添加一条边。对于顶点集 $X \subseteq V$，我们定义 $\delta_G(X) = E_G(X, V \setminus X)$；对于单个顶点 v，我们将$\delta_G(v) = \delta_G(v)$写作。我们用cc(G)表示 G 的连通分量数。G 的一个割是将其顶点划分为两部分 $V = V_L \cup V_R$、$V_L \cap V_R = \emptyset$。

团表达式与团宽

带标签图是一张图 $G = (V, E)$，以及一个标签函数 $\mathtt{lab}: V \to \mathbb{N} = \{1、2、3、.....\}$；我们通常省略明确提及 $\mathtt{lab}$。如果对于所有 $v \in V$，k-带标签 $\mathtt{lab}(v) \leq k$，则称带标签图是 k-带标签 的。我们考虑带标签图上的以下四种操作：

- 引入-操作 $\ell(v)$ 构造一个单顶点图，其中唯一的顶点v 具有标签 ℓ；并-操作 $G_1 \oplus G_2$ 构造两个带标签图G_1 和 G_2的不交并；重标记-操作 $\rho_{i \to j}(G)$ 将 G 中所有具有标签 i 的
- 顶点的标签更改为标签 j；连接-操作 $\eta_{i,\ j}(G)$，$i \neq j$，在 G 中标签为 i 的每个顶点与标签为 j的每个顶点之间添加一条边。
-
-

仅由引入、并集、重标记和连接操作组成的有效表达式称为团表达式。团表达式 μ 构造的图记为 G_μ，构造的标签函数记为 $\mathtt{lab}_\mu$：$V(G_\mu) \to \mathbb{N}$。

我们以自然的方式将团表达式 μ 与语法树 T_μ 关联起来，并将对应的操作关联到每个节点 $t \in V(T_\mu)$。对于任意节点 $t \in V(T_\mu)$，以 t 为根的子树确定一个子表达式 μ_t。固定团表达式 μ 后，对于任意 $v \in V(T_\mu)$，我们定义$G_t = G_{\mu_t}$、$V_t = V(G_t)$、$E_t = E(G_t)$和 $\mathtt{lab}_t = \mathtt{lab}_{\mu_t}$。此外，我们用 $V_t^\ell = \mathtt{lab}_t^{-1}(\ell)$表示节点 t 上标签为 ℓ 的所有顶点组成的集合，并用$L_t = \{\ell \in \mathbb{N} : V_t^\ell \neq \emptyset\}$ 表示节点 t上的非空标签集合。

我们称团表达式 μ 为 k-团表达式，或简称为 k-表达式，如果对于所有 $t \in V(T_\mu)$，G_t 都是k-标记的。图 G 的团宽 记作 cw(G)，定义为满足存在一个 k-表达式 μ 且 $G = G_\mu$ 的最小k。若每次并操作中，第二个图都只包含一个顶点，则称团表达式 μ 是线性的。相应地，我们将图 G 的线性团宽度 定义为 lin-cw(G)，即只考虑线性团表达式。

强指数时间假设

强指数时间假设 （SETH） [12, 30] 关注 q-可满足性问题的复杂度，即每个子句至多包含 q 个文字的可满足性问题。我们定义 $c_q = \inf\{\delta : q\text{-可满足性问题可在}\mathcal{O}(2^{\delta n})\}$ 时间内求解，对所有 $q \geq 3$ 均成立。Impagliazzo 和 Paturi [29] 提出的较弱假设指数时间假设（ETH）认为$c_3 > 0$，而强指数时间假设则指出 $\lim_{q \to \infty} c_q = 1$。等价地，对每个 $\delta < 1$，都存在某个 q，使得 q-可满足性问题无法在$\mathcal{O}(2^{\delta n})$时间内求解。对于我们的下界，SETH 的以下较弱变体也足够，它也称为 CNF-SETH。

▶ **猜想 3.1** (CNF-SETH)。对于每个 $\varepsilon > 0$，不存在一种算法，能够对具有 n 个变量和 m 个子句的可满足性问题，在 $O(\mathrm{poly}(m)(2 - \varepsilon)^n)$时间内求解。

隔离引理

▶ **定义 3.2**。 若存在唯一的$S' \in \mathcal{F}$，使得 $\mathbf{w}(S') = \min_{S\in\mathcal{F}} \mathbf{w}(S)$，则函数 $\mathbf{w}: U \to \mathbb{Z}$ 隔离 集合族 $\mathcal{F} \subseteq \mathcal{P}(U)$；其中，对于 U 的子集 X，我们定义 $\mathbf{w}(X) = \sum_{u\in X}\mathbf{w}(u)$。

▶ **引理 3.3** (隔离引理， [38])。设 $F \subseteq P(U)$ 是全集 U 上的一个非空集合族。设 $N \in \mathbb{N}$，并对每个 $u \in U$ 独立且均匀地随机选择一个权重 $\mathbf{w}(u) \in [N]$。那么 $\mathbb{P}[\mathbf{w}$ 将 $F] \geq 1 - |U|/N$ 隔离。

在模 2 计数对象时，隔离引理通过以高概率确保解唯一，使我们能够避免不必要的抵消。在我们的应用中，我们将选择 N，使得错误概率小于 $1/2$。

3.1 Cut andCount

设 $G = (V, E)$ 表示一个连通图。为了求解图 G 上带有连通性约束的顶点选择问题，我们作如下的一般定义。该问题的连通解族记为$\mathcal{S} \subseteq \mathcal{P}(V)$。我们需要判断$\mathcal{S}$ 是否为空。Cygan 等人的割与计数技术 [19] 分两部分完成这一任务：

- **割部分：** 通过放宽连通性约束，我们得到一个可能包含连通解的集合 $\mathcal{S} \subseteq \mathcal{R} \subseteq \mathcal{P}(V)$，称为候选解。集合 $\mathcal{A}$ 包含由候选解 $X \in \mathcal{R}$ 和 $G[X]$ 的一致割 (X_L, X_R)组成的对 $(X,(X_L, X_R))$，参见定义 3.4。**计数部分：** 我们使用一个子过程计算 $|\mathcal{A}|$ 模 2。对一致割的定义保证，非连通候选解 $X \in \mathcal{R}\setminus\mathcal{S}$ 会相互抵消，因为它们与偶数个割一致。因此，只剩下连通候选解 $X \in \mathcal{S}$，并且当 $|\mathcal{A}|$ 的奇偶性为奇数时，我们有 $\mathcal{S} \neq \emptyset$。

如果 $|\mathcal{S}|$ 能被 2 整除，那么这种方法就会失效，因为在模 2 计数时，连通候选解也会相互抵消。隔离引理（引理 3.3）使我们能够通过随机化避免这个问题。通过抽取一个随机权重函数 $\mathbf{w}: V \to [N]$，我们可以改为统计权重固定的数对；如果解存在，那么很可能存在一个权重，使得解唯一。

▶ **定义 3.4** ([17])。 无向图 $G = (V, E)$ 的割 (V_L, V_R) 若满足$u \in V_L$ 且 $v \in V_R$ 蕴含$\{u, v\} \notin E$，则称为一致的，即 $E_G(V_L, V_R) = \emptyset$。$G$ 的一致割子图是一个二元组 $(X, (X_L, X_R))$，满足 $X \subseteq V$，且 (X_L, X_R) 是 $G[X]$ 的一致割。我们用$\mathcal{C}(G)$ 表示 G 的一致割子图集合。

为确保连通解不会与偶数个一致割兼容，我们通常会强制指定一个顶点位于一致割的左侧。由此得到一致割的以下基本性质。

▶ **引理 3.5** ([17])。 设 X 是满足 $v_* \in X \subseteq V$ 的顶点子集。满足 $v_* \in X_L$ 的一致割子图$(X,(X_L, X_R))$ 的数量等于 $2^{\mathsf{cc}(G[X])-1}$。

借助引理 3.5，我们可以通过确定相应候选解的一致割数量的奇偶性，区分不连通候选解与连通候选解。

▶ **推论 3.6**。 设 $R \subseteq P(V)$ 是一个顶点集族，使得每个 $X \in R$ 都包含 v_*。如果集合 $A = \{(X, (X_L, X_R)) \in \mathcal{C}(G): X \in R\ v_* \in X_L\}$ 的基数为奇数，则存在一个 $X \in R$，使得 $G[X]$ 连通。

证明。 我们将证明其逆否命题，因此假设对于每个$X \in \mathcal{R}$， $G[X]$ 都不连通。根据引理 3.5，我们有 $|\mathcal{A}| = \sum_{X\in\mathcal{R}} 2^{\mathsf{cc}(G[X])-1} \equiv_2 0$，因为对于所有 $X \in \mathcal{R}$， $\mathsf{cc}(G[X]) \geq 2$。◀

3.2 规范团表达式

设 μ 是 $G = (V, E)$的一个 k-表达式；相应的语法树为 T_μ。若一个团表达式 μ 满足：对于任意连接操作$\eta_{i,j}(G_{t'}) = t \in V(T_\mu)$，都有$E_{t'}(V^i_{t'}, V^j_{t'}) = \emptyset$，则称其为不可约的；也就是说，连接操作添加的边此前都不存在。

▶**定理 3.7** ([13])。 任意 k-表达式 μ 都可以在多项式时间内转换为等价的，即$G_{\mu'} = G_\mu$ 不可冗余的 k-表达式 μ'。

▶ 引理 *3.8*（[6]）。如果 μ 是图 $G = (V, E)$ 和$t \in V(T_\mu)$ 的一个不可冗余 k-表达式，那么对于所有标签 $\ell \in L_t$ 和顶点 $u, v \in V^\ell_t$，我们有 $N_G(u) \setminus V_t = N_G(v) \setminus V_t$。此外，如果 $u \in V^i_t$, $v \in V^j_t$ 满足 $i \neq j \in L_t$ 且 $\{u, v\} \in E \setminus E_t$，那么$E_G(V^i_t, V^j_t) \subseteq E \setminus E_t$。

此后，我们假设给定的 k-表达式 μ 是不可冗余的。不可冗余性仍会导致若干与空标签有关的边界情况，需要在动态规划算法中特殊处理。为避免算法中的这些额外工作，我们将展示如何转换任意团表达式，使这些边界情况不再出现。

▶ **定义 3.9**。 如果图 $G = G_\mu$ 的团表达式 μ 满足以下性质，则称其为规范的：

- μ 是无冗余的；对于每个连接节点 $\eta_{i,\ j}(G_{t'}) = t \in V(T_\mu)$，其中 t' 是 t的子节点，有
- $G_t \neq G_{t'}$，即 t 至少添加一条边，并且对于每个重标记节点 $\rho_{i\to j}(G_{t'}) = t \in V(T_\mu)$，其中 t' 是 t的子节点，有$V^i_{t'} \neq \emptyset$ 和 $V^j_{t'} \neq \emptyset$。
-

▶ **引理 3.10**。 任意 k-表达式 μ 都可以在多项式时间内转换为一个等价的，即 $G_{\mu'} = G_\mu$，规范的 k-表达式 μ'。

证明。 首先运行定理 3.7 的算法，我们可以假设 μ 已经不可冗余。如果合并节点 $t = \eta_{i,\ j}(G_{t'})$ 没有添加任何边，那么根据不可冗余性，必有 $V^i_{t'} = \emptyset$ 或 $V^j_{t'} = \emptyset$。显然，我们可以直接从表达式中删除此类连接节点。下一步要注意，任何形如 $\rho_{i\to j}(G_{t'}) = t \in V(T_\mu)$ 的重标记节点，其中 t' 是 t 和 $V^i_{t'} = \emptyset$ 的子节点，都可以从 μ 中删除，而不改变生成的图，因为此类节点不会改变任何标签。

现在，假设 μ 包含 $p > 0$ 个形如 $\rho_{i\to j}(G_{t'}) = t \in V(T_\mu)$ 的重标记节点，其中 t' 是 t 和 $V^i_{t'} \neq \emptyset$ 以及 $V^j_{t'} = \emptyset$ 的子节点；我们称此类重标记为不必要的。我们选取一个这样的实例，并说明如何将其删除。如果 t 是 μ 的根，那么我们可以直接删除 t，而不改变生成的图。如果 t 不是 μ 的根，那么令 s 为 t 的父节点。我们在 T_μ 中 t 的每个真后代处交换标签 i 和 j 的角色，并删除节点 t，令 t' 代替 t 成为 s 的子节点。该变换的正确性可通过沿着 T_μ 进行简单的自底向上归纳来证明，而且这一变换不会产生任何新的不必要重标记。

对每个不必要的重新标记重复此转换，我们就能在多项式时间内得到一个等价的优美 k-表达式 μ'。◀

在设计以团宽为参数的问题的紧致算法时，人们常常会发现，某些状态只有在具有相应标签的顶点已在当前子表达式中获得了所有关联边后，该标签才能达到。但对于这类标签，动态规划算法无需存储该状态，因为在剩余表达式中，它们不会与其他标签发生有意义的交互。因此，为了改进

运行时间，我们只为活动标签存储状态，即包含仍缺少某些关联边的顶点的标签；非活动标签称为失活标签。这一思想此前已多次用于团表达式上的动态规划 [24, 34, 36]。

虽然可以预先计算给定团表达式 μ 中节点$t \in V(T_\mu)$处哪些标签处于活动状态，但我们选择在语法树中显式标记标签何时不再处于活动状态。为此，我们在至少将一个标签从活动状态改为失活状态的合并节点之后，在语法树中增添死节点 。这些死节点的作用与优美树分解 [35]中的遗忘顶点节点类似。尤其是在连通支配集算法中，这种显式标记很有帮助，因为它能让我们清晰地区分两种计算：合并节点的标准计算，以及标签转为失活时必须执行的额外计算。现在我们继续给出形式化定义。

在本节余下部分，我们假设 G 是一个至少有两个顶点的连通图。

▶ **定义 3.11**。 给定一个用于 $G = (V, E)$ 的团表达式 μ 以及一个节点 $t \in V(T_\mu)$， t 处的失活顶点集合 定义为 $D_t = \{v \in V_t : \delta_G(v) \subseteq E_t\}$。若顶点 $v \in V_t \setminus D_t$ 在处于活动状态 t，则称其为活动顶点。

▶ **引理 3.12.** 给定一个不可约的 k-表达式 μ，其对应于 $G = (V, E)$，一个节点 $t \in V(T_\mu)$，以及一个非空标签 $\ell \in L_t$，则要么 $V_t^\ell \cap D_t = \emptyset$，要么 $V_t^\ell \subseteq D_t$。

证明。若 $x \in V_t^\ell \setminus D_t \neq \emptyset$，则存在一条边 $\{x, y\} \in E \setminus E_t$。令 $\tilde{t} \in V(T_\mu)$ 为满足 $x, y \in V_{\tilde{t}}$ 的 t 的最低祖先；我们要么有 $\tilde{t} = t$，要么 $\tilde{t}$ 是 t 上方的某个并集节点。无论哪种情况，都可以假设存在 $i \neq j \in L_{\tilde{t}}$，满足 $x \in V_t^\ell \subseteq V_{\tilde{t}}^i$ 和 $y \in V_{\tilde{t}}^j$。根据引理 3.8 的第二部分可知，对于所有 $x' \in V_t^\ell \subseteq V_{\tilde{t}}^i$，都有 $\{x', y\} \in E \setminus E_{\tilde{t}} \subseteq E \setminus E_t$。因此， $V_t^\ell \setminus D_t \neq \emptyset$ 蕴含 $V_t^\ell \cap D_t = \emptyset$，从而证明了该引理。◀

如果不要求不可冗余性，那么很容易构造出使引理 3.12 不成立的团表达式，即 $\emptyset \neq V_t^\ell \cap D_t \neq V_t^\ell$。通过只考虑不可冗余的团表达式，我们可以说一个标签整体上是失活或活动的，这简化了对失活顶点的处理，因为标签一旦失活，我们就可以只进行一次计算。特别是，如果没有不可冗余性，某个节点上只包含失活顶点的标签之后可能会在表达式中获得新的活动顶点，因此这类情况往往难以处理。

下面的定义形式化地描述了如何处理活标签和死标签，以及当标签从活动变为失活时添加的死节点。

▶ **定义 3.13**。 给定一个用于 $G = (V, E)$ 的不可冗余 k-表达式，其 μ 的扩展语法˜树 T_μ 由语法树 T_μ 构成：在每个合并节点 $t = \eta_{i,j}(G_{t'})$ 的正上方插入最多两个死节点，其中 t' 是 T_μ中 t 的子节点，插入依据如下标准：

- 若 $V_t^i \subseteq D_t \setminus D_{t'}$，则插入节点 $\perp_i$；若 $V_t^j \subseteq D_t \setminus D_{t'}$，则插入节点 $\perp_j$；若节点 $\perp_i$ 和 $\perp_j$ 均需插入，则按任意顺序插入。

我们将针对 $\ell \in [k]$ 的记号 G_t、 V_t、 D_t、 V_t^ℓ 扩展至失活节点，使其继承子节点的值。

对于增广语法树的每个节点 $t \in V(\tilde{T}_\mu)$，我们归纳地定义以下集合：

活动标签 $L_t^{live} \subseteq L_t$ 的

$$L_t^{live} = \begin{cases} \{\ell\} & \text{if } t = \ell(v), \\ L_{t'}^{live} \setminus \{i\} & \text{if } t = \rho_{i\to j}(G_{t'}), \\ L_{t'}^{live} & \text{if } t = \eta_{i,j}(G_{t'}), \\ L_{t'}^{live} \setminus \{\ell\} & \text{if } t = \bot_\ell(G_{t'}), \\ L_{t_1}^{live} \cup L_{t_2}^{live} & \text{if } t = G_{t_1} \oplus G_{t_2}. \end{cases}$$

对偶地，死标签 $L_t^{dead} \subseteq L_t$ 由 $L_t^{dead} = L_t \setminus L_t^{live}$给出。

现在我们证明，除去待处理的死节点外， L_t^{live}包含在 t处仅由活顶点组成的所有非空标签。根据引理 3.12，不多余的 k-表达式中的任何标签都不能同时包含活顶点和死顶点。

▶ **引理 3.14。 </style}设 是一个关于 (V, E) 的良好 k-表达式，且 $\hat{} T_\mu$ 是其增广语法树。对于任意节点 $t \in V(\hat{} T_\mu)$ 和 $\ell \in L_t$，我们有 $V_t^\ell \cap D_t = \emptyset$ 蕴含 $\ell \in L_t^{live}$。如果 t不是死节点的子节点，那么对于每个 $\ell \in L_t$，我们甚至有 $V_t^\ell \cap D_t = \emptyset$ 当且仅当 $\ell \in L_t^{live}$。**

证明。 首先，回顾 $V_t^\ell \cap D_t \in \{\emptyset, V_t^\ell\}$ 对于每个 $t \in V(\hat{} T_\mu)$ 以及 $\ell \in L_t$，由引理 3.12 可知。^我们沿着增广语法树 T_μ 进行归纳证明，并根据当前节点的类型分类讨论。注意，死节点的子节点只能是合并节点或其他死节点。

- 如果 $t = \ell(v)$，那么根据假设， $D_t = \emptyset$ as v 无法被孤立，因此该命题成立。如果 $t = \rho_{i\to j}(G_{t'})$，
- 那么 $D_t = D_{t'}$，根据归纳法，我们有 $L_{t'}^{live} = \{\ell \in L_{t'} \colon V_{t'}^\ell \cap D_{t'} = \emptyset\}$。由于所有其他标签都保持不变，唯一值得关注的标签是 i 和 j。由于 $i \notin L_t$，标签 i无需证明。我们有 $V_t^j = V_{t'}^i \cup V_{t'}^j$，并且根据 μ的良好性，有 $i,\ j \in L_{t'}$以及 $j \in L_{t'}^{live} \iff i,\ j \in L_{t'}^{live}$。现在，

$$j \in L_t^{live} \iff i, j \in L_{t'}^{live} \iff V_{t'}^i \cap D_{t'} = \emptyset \wedge V_{t'}^j \cap D_{t'} = \emptyset \iff V_t^j \cap D_t = \emptyset$$

其中，最后一个等价关系由 $D_t = D_{t'}$ 和 $V_t^j = V_{t'}^i \cup V_{t'}^j$ 得出。

- 若 $t = \eta_{i,\ j}(G_{t'})$，则 $V_t^\ell = V_{t'}^\ell$ 对所有 $\ell \in L_t = L_{t'}$ 和 $D_{t'} \subseteq D_t$ 均成立，因此正向蕴含由 t'处的陈述得出。若 t 不是死节点的子节点，那么我们甚至有 $D_t = D_{t'}$，因此逆向蕴含也由 t'处的陈述得出。

- 若 t 是一个死节点，那么 t 的子节点或孙节点必定是合并节点 $t^* = \eta_{i,j}(G_{t'})$，它使得 t 得以存在。根据归纳法， $L_{t'}^{live} = \{\ell \in L_{t'} \colon V_{t'}^\ell \cap D_{t'} = \emptyset\}$。对于每个 $\ell \in L_t = L_{t'}$， $D_t \setminus D_{t'} = D_{t^*} \setminus D_{t'} \subseteq V_t^i \cup V_t^j$ 以及 $L_t^{live} = L_{t'}^{live} \setminus X$，我们有$V_{t^*}^\ell = V_t^\ell = V_{t'}^\ell$，其中 $\emptyset \neq X \subseteq \{i, j\}$ 是从t^*到 t期间从活标签中移除的标签集合。由于 $L_t^{live} \setminus \{i, j\} = L_{t'}^{live} \setminus \{i, j\}$，根据归纳法，对于所有 $\ell \in L_t \setminus \{i, j\}$，该等价关系均成立。对于 $\ell \in X = L_{t'}^{live} \setminus L_t^{live}$，根据死节点的构造，我们有 $V_t^\ell \subseteq D_t \setminus D_{t'}$，因此 $V_t^\ell \cap D_t \neq \emptyset$，从而正向蕴含对节点 t处的所有标签 $\ell \in L_t$均成立。最后，若 t 不是死节点的子节点，那么我们已经移除了所有满足 $V_t^\ell \subseteq D_{t^*} \setminus D_{t'} = D_t \setminus D_{t'}$的标签$\ell \in L_t$，因此在这种情况下，逆向蕴含对所有标签 $\ell \in L_t$ 也成立。

- 如果 t 是一个并集节点，那么对于所有 $\ell \in [k]$ 和 $D_t = D_{t_1} \cup D_{t_2}$， $V_t^\ell = V_{t_1}^\ell \cup V_{t_2}^\ell$。如果$\ell \in L_{t_1} \triangle L_{t_2} \subseteq L_t$，那么不失一般性，假设 $\ell \in L_{t_1} \setminus L_{t_2}$，因此$V_{t_1}^\ell = V_t^\ell$和 $V_{t_2}^\ell = \emptyset$，所以我们可以看到

$$\ell \in L_t^{live} \iff \ell \in L_{t_1}^{live} \iff V_{t_1}^\ell \cap D_{t_1} = \emptyset \iff V_{t_1}^\ell \cap (D_{t_1} \cup D_{t_2}) = \emptyset \iff V_t^\ell \cap D_t = \emptyset,$$

其中第三个等价变形用到了 $V^\ell_{t_1} \cap D_{t_2} = \emptyset$。另一方面，如果 $\ell \in L_{t_1} \cap L_{t_2} \subseteq L_t$，由引理 3.12 可知，$V^\ell_{t_1} \cap D_{t_1} = \emptyset$ 当且仅当 $V^\ell_{t_2} \cap D_{t_2} = \emptyset$。因此

$$\ell \in L^{live}_t \iff \ell \in L^{live}_{t_1} \cup L^{live}_{t_2} \iff V^\ell_{t_1} \cap D_{t_1} = \emptyset \wedge V^\ell_{t_2} \cap D_{t_2} = \emptyset \iff V^\ell_t \cap D_t = \emptyset,$$

其中，最后一个等价关系由 $V^\ell_{t_1} \cap D_{t_2} = \emptyset = V^\ell_{t_2} \cap D_{t_1}$得出。 ◀

▶ **引理 3.15.** 设 μ 是 $G = (V, E)$的一个优良 k-表达式。对于每个节点 $t \in V(\hat{T}_\mu)$，死标签集合 L^{dead}_t满足以下递推式：

$$L^{dead}_t = \begin{cases} \emptyset & \text{if } t = \ell(v), \\ L^{dead}_{t'} \setminus \{i\} & \text{if } t = \rho_{i\to j}(G_{t'}), \\ L^{dead}_{t'} & \text{if } t = \eta_{i,j}(G_{t'}), \\ L^{dead}_{t'} \cup \{\ell\} & \text{if } t = \bot_\ell(G_{t'}), \\ L^{dead}_{t_1} \cup L^{dead}_{t_2} & \text{if } t = G_{t_1} \oplus G_{t_2}. \end{cases}$$

证明。对于每个 $t \in V(\hat{T}_\mu)$，根据 L^{dead}_t的定义，非空标签集合 L_t 是 L^{live}_t和 L^{dead}_t的不交并。对于连接节点和重标记节点，有 $L_t = L_{t'}$，因此递推式可直接由 L^{live}_t的递推式推出。对于引入节点 $t = \ell(v)$，有 $L_t = L^{live}_t = \{\ell\}$，因此 $L^{dead}_t = \emptyset$。对于重标记节点 $t = \rho_{i\to j}(G_{t'})$，有$L_t = L_{t'} \setminus \{i\}$，因此可得该递推式。

对于并集节点 $t = G_{t_1} \oplus G_{t_2}$，根据引理 3.14，我们有 $L^{live}_{t_i} = \{\ell \in L_{t_i} \colon V^\ell_{t_i} \cap D_{t_i} = \emptyset\}$ 其中 $i \in \{1, 2\}$。因此，对于任意 $\ell \in L_{t_1} \cap L_{t_2} \subseteq L_t$，根据 μ 的不可冗余性，我们有 $\ell \in L^{live}_{t_1} \Leftarrow\!\Rightarrow \ell \in L^{live}_{t_2}$。因此，我们计算

$$\begin{aligned} L^{dead}_t &= L_t \setminus L^{live}_t = (L_{t_1} \cup L_{t_2}) \setminus (L^{live}_{t_1} \cup L^{live}_{t_2}) \\ &= (L_{t_1} \setminus (L^{live}_{t_1} \cup L^{live}_{t_2})) \cup (L_{t_2} \setminus (L^{live}_{t_1} \cup L^{live}_{t_2})) \\ &= (L^{dead}_{t_1} \cap (L_{t_1} \setminus L^{live}_{t_2})) \cup (L^{dead}_{t_2} \cap (L_{t_2} \setminus L^{live}_{t_1})) \\ &= L^{dead}_{t_1} \cup L^{dead}_{t_2}, \end{aligned}$$

因为前面的论证表明 $L^{dead}_{t_1} \cap L^{live}_{t_2} = \emptyset = L^{dead}_{t_2} \cap L^{live}_{t_1}$。 ◀

ˆ 由于集合 D_t 沿着增广语法树 T_μ 向上时不会缩小，失活顶点不可能再次变为活动顶点。尽管 D_t 与 L^{live}_t 之间的关系并非总是双向成立（参见引理 3.14），下一个引理仍将表明，顶点不可能从失活标签切换回活动标签。

▶ **引理 3.16.** 设 μ 是一个关于 $G = (V, E)$ 的良好 k-表达式，且 $v \in V$ 是一个顶点。对于任意满足 $v \in V^\ell_t$且 $\ell \in L^{dead}_t$的节点 $t \in V(\hat{T}_\mu)$，以及任意祖先 $t^* \in V(\hat{T}_\mu)$ 和满足 $v \in V^{\ell^*}_{t^*}$ 的 $\ell^* \in L_{t^*}$，均有 $\ell^* \in L^{dead}_{t^*}$。

证明。考察引理 3.15 的递推式，并注意，对于合并节点和死节点t，有 $L^{dead}_{t'} \subseteq L^{dead}_t$，其中 t' 是 t的子节点。对于并集节点 t，有$L^{dead}_{t_i} \subseteq L^{dead}_t$，其中 t_i、 $i \in [2]$是 t的子节点。剩下需要考虑重标记节点$t = \rho_{i\to j}(G_{t'})$，其中 t' 是 t的子节点。此处有 $L^{dead}_t = L^{dead}_{t'} \setminus \{i\}$，因为在 t处标签i 为空。若有 $i \in L^{dead}_{t'}$，那么根据μ 的不可冗余性和引理 3.14，也有 $j \in L^{dead}_{t'}$和 $j \in L^{dead}_t$。由于 $V^j_t = V^i_{t'} \cup V^j_{t'}$，这表明在重标记节点处，顶点也无法从死标签切换为活标签。◀

在 k-表达式的并集节点处，通常需要高效计算动态规划算法中的类似卷积的递推式。第一步是处理标签

在并集节点的某一个子节点处非空的标签。对于这些标签，计算通常很简单；剩下的工作是针对在两个子节点处均非空的标签，设计适用于该问题的快速卷积算法。为了概括标签集的这种划分，我们给出以下定义。

▶ 定义 3.17. 设 μ 是 $G=(V,E)$ 的一个良好 k-表达式，且 $t \in V(\hat{T}_\mu)$ 是一个并集节点，即 $t = G_{t_1} \oplus G_{t_2}$。函数 $f\colon L_t^{live} \to$ 的在 t 处的并分割，其中 States 是某个有限集，是函数 $f_{t,1} = f|_{\tilde{L}_t}$,1, $f_{t,2} = f|_{\tilde{L}_t}$,2, $f_{t,12} = f|_{\tilde{L}_t}$,12，其中$\tilde{L}_{t,1} = L_{t_1}^{live} \setminus L_{t_2}$, $\tilde{L}_{t,2} = L_{t_2}^{live} \setminus L_{t_1}$, $\tilde{L}_{t,12} = L_{t_1}^{live} \cap L_{t_2}^{live}$。

▶ **引理 3.18.** 设 μ 是 $G=(V,E)$ 的一个良好 k-表达式，且 $t \in V(\hat{T}_\mu)$ 是一个并集节点，即 $t = G_{t_1} \oplus G_{t_2}$。我们有 $L_{t_1}^{live} \cap L_{t_2}^{dead} = \emptyset = L_{t_1}^{dead} \cap L_{t_2}^{live}$，并且集合 $\tilde{L}_{t,1}$, $\tilde{L}_{t,2}$ 和 $\tilde{L}_{t,12}$ 构成 L_t^{live}的一个划分。

证明。根据定义，这三个集合显然互不相交。由于 $L_t^{live} = L_{t_1}^{live} \cup L_{t_2}^{live}$，只需说明 $\tilde{L}_{t,1} = L_{t_1}^{live} \setminus L_{t_2}^{live}$和 $\tilde{L}_{t,2} = L_{t_2}^{live} \setminus L_{t_1}^{live}$。注意，$\tilde{L}_{t,1} = L_{t_1}^{live} \setminus L_{t_2}^{live}$等价于 $L_{t_1}^{live} \cap L_{t_2}^{dead} = \emptyset$，另一个等式也同理。不失一般性，我们考虑 $\tilde{L}_{t,1}$。我们有 $\tilde{L}_{t,1} = L_{t_1}^{live} \setminus L_{t_2} \subseteq L_{t_1}^{live} \setminus L_{t_2}^{live}$，因为 $L_{t_2}^{live} \subseteq L_{t_2}$。对于另一方向，注意任意标签 $\ell \in L_{t_1}^{live} \cap L_{t_2}$ 都必须包含在 $L_{t_2}^{live}$中，否则根据引理 3.14，我们会得到 $V_{t_1}^\ell \cap D_{t_1} = \emptyset$ 和 $V_{t_2}^\ell \subseteq D_{t_2}$，这意味着$V_t^\ell \cap D_t = V_{t_2}^\ell \notin \{\emptyset, V_t^\ell\}$ 与引理 3.12 矛盾。◀

4 动态规划算法

4.1 连通顶点覆盖

在连通顶点覆盖问题中，给定一个图 $G=(V,E)$、一个代价函数 $\mathbf{c}\colon V \to \mathbb{Z}_{>0}$，以及一个称为预算 的非负整数 b，我们需要判断是否存在一个顶点子集 X，使得 $G-X$ 不包含任何边、 $G[X]$ 是连通的，并且 $\mathbf{c}(X) \le b$。我们只考虑代价在输入规模上多项式有界的连通顶点覆盖实例。此外，我们假设 G 是连通的，且至少包含两个顶点。

给定图 $G=(V,E)$ 的一个 k-表达式 μ，根据引理 3.10，经过多项式时间预处理后，我们可以假设 μ 是一个良好的 k-表达式。我们希望应用割与计数技术，在时间 $\mathcal{O}^*(6^k)$ 内求解连通顶点覆盖问题。为此，我们首先选取G中的一条边，对其一个端点 v_*进行分支，并且在该分支中只考虑包含v_*的解。此外，我们根据隔离引理（参见引理 3.3）采样一个权重函数 $\mathbf{w}\colon V \to [2|V|]$。我们沿着增广语法树 $\hat{T}_\mu$进行自底向上动态规划。在每个节点 $t \in V(\hat{T}_\mu)$，我们考虑以下部分解族

$$\mathcal{A}_t = \{(X,(X_L,X_R)) \in \mathcal{C}(G_t) : G_t - X \text{ contains no edges and } v_* \in V_t \to v_* \in X_L\}.$$

换言之，如果可能，$\mathbf{A}_t$ 包含所有满足条件的 G_t 一致切割顶点覆盖，使得 v_* 位于割的左侧。对于每个 $t \in V(\hat{T}_\mu)$， $c \in [0,\mathbf{c}(V)]$, $w \in [0,\mathbf{w}(V)]$, ，我们定义 $\mathbf{A}_t^{c\ w} = \{(X,(X_L,X_R)) \in \mathbf{A}_t : \mathbf{c}(X) = c, \mathbf{w}(X) = w\}$。令 $\hat{r}$ 表示 $\hat{T}_\mu$增广语法树的根节点。根据推论 3.6，如果存在 $c \in [0,b]$ 和 $w \in [0,\mathbf{w}(V)]$，使得 $\mathbf{A}^{c,w}_{\hat{r}}$的基数为奇数，那么就存在 G 的一个连通顶点覆盖 X，满足 $\mathbf{c}(X) \le b$。

为了便于设计动态规划算法，我们需要分析部分解 $(X,(X_L,X_R)) \in \mathcal{A}_t$ 在标签 V_t^ℓ、$\ell \in L_t$上的行为。单个顶点$v \in V_t^\ell$可以处于状态 $\Omega = \{\mathbf{0}, \mathbf{1}_L, \mathbf{1}_R\}$中的一种，分别表示 $v \notin X$ 或 $v \in X_L$,

或 $v \in X_R$。为了检验 $(X, (X_L, X_R))$ 的可行性，只需针对每个标签存储哪些顶点状态出现、哪些没有出现，因为由 $\mathcal{A}_t$ 隐含的约束属于“类约束满足问题”，并且可以在每次连接时通过考虑所有涉及的顶点状态对来进行评估。由此可得 Ω 的幂集 $\mathcal{P}(\Omega)$，作为每个标签可能状态的集合。

幂集 $\mathrm{P}(\Omega)$ 原则上会为每个标签产生八种不同的状态。不过，我们可以排除状态 $\emptyset$ 和状态 $\Omega = \{\mathbf{0}, \mathbf{1}_L, \mathbf{1}_R\}$。前者可以排除，因为我们只需要存储非空标签的状态。排除状态 $\Omega = \{\mathbf{0}, \mathbf{1}_L, \mathbf{1}_R\}$ 则更微妙：对于该状态，任何额外的关联连接都会导致不可行解，因此只有死标签（参见定义 3.13）可以取此状态。我们稍后会再讨论这个问题。由于只需存储活动标签的状态，我们令 $\mathrm{States} = \mathrm{P}(\Omega) \setminus \{\emptyset, \Omega\} = \{\{\mathbf{0}\}, \{\mathbf{1}_L\}, \{\mathbf{1}_R\}, \{\mathbf{0}, \mathbf{1}_L\}, \{\mathbf{0}, \mathbf{1}_R\}, \{\mathbf{1}_L, \mathbf{1}_R\}\}$。

给定一个节点 $t \in V(\hat{T}_\mu)$，t-签名 是一个函数 $f\colon L_t^{live} \to$ **状态**。对于每个节点$t \in V(\hat{T}_\mu)$、$\bar{c} \in [0, \mathbf{c}(V)]$、$\bar{w} \in [0, \mathbf{w}(V)]$, 以及 t-签名 f，我们定义

$$\begin{aligned}\mathcal{A}_t^{\bar{c},\bar{w}}(f) = \{(X, (X_L, X_R)) \in \mathcal{A}_t^{\bar{c},\bar{w}} : \mathbf{0} \in f(\ell) &\leftrightarrow V_t^\ell \not\subseteq X \text{ for all } \ell \in L_t^{live},\\ \mathbf{1}_L \in f(\ell) &\leftrightarrow X_L \cap V_t^\ell \neq \emptyset \text{ for all } \ell \in L_t^{live},\\ \mathbf{1}_R \in f(\ell) &\leftrightarrow X_R \cap V_t^\ell \neq \emptyset \text{ for all } \ell \in L_t^{live}\}.\end{aligned}$$

我们不直接计算集合$\mathcal{A}_t^{c\ w}{}_{,}(f)$，而只计算其基数的奇偶性，即 $A^{c}{}_{,t}{}^{w}(f) = |\mathcal{A}_t^{c\ w}{}_{,}(f)|$ 模 2。

现在我们可以更正式地论证，排除状态 $\emptyset$ 和 Ω 不会造成问题。首先，对于任意非空的 V_t^ℓ，三种情况 $V_t^\ell \not\subseteq X$、$X_L \cap V_t^\ell \neq \emptyset$ 或$X_R \cap V_t^\ell \neq \emptyset$ 中至少有一种必须发生，因此状态 $\emptyset$ 不可能由任何满足 $\ell \in L_t^{live}$ 的 V_t^ℓ 达到。

其次，考虑某个节点 t，它不是死节点的子节点，并且 $(X, (X_L, X_R)) \in \mathcal{A}_t^{c\ w}$，使得存在某个活动标签 $\ell \in L_t^{live}$，令三种情况 $V_t^\ell \not\subseteq X$、$X_L \cap V_t^\ell \neq \emptyset$ 和 $X_R \cap V_t^\ell \neq \emptyset$ 同时发生。由于 ℓ 是一个活动标签，根据引理 3.14，存在某个$v \in N_G(V_t^\ell) \setminus N_{G_t}(V_t^\ell)$。我们声称，$(X, (X_L, X_R))$无法扩展为 $G' = G[V_t \cup \{v\}]$ 的一个一致割顶点覆盖 $(X', (X'_L, X'_R))$（因而也无法扩展为 G的一个一致割顶点覆盖）。如果 $v \notin X'$，那么在 G' 中存在一条位于 V_t^ℓ和 v之间的未覆盖边。如果 $v \in X'$，那么 G' 中存在一条穿过割 (X'_L, X'_R) 的边，因此该割不可能是一致的。因此，我们可以放心地丢弃所有达到带有活动标签的状态 $\{\mathbf{0}, \mathbf{1}_L, \mathbf{1}_R\}$ 的部分解，因为它们永远无法扩展为全局解。

当两个顶点集在同一个公共标签下合并时，即在重新标记操作或并操作期间，可以得到状态 $\Omega = \{\mathbf{0}, \mathbf{1}_L, \mathbf{1}_R\}$。我们将给出量 $\mathcal{A}_t^{c\ w}{}_{,}(f)$ 的递推式，其中 f 是一种 t-签名，不允许在任何标签下取状态 Ω，因此，算法会隐式地过滤掉这类情形，因为递推式不会考虑会导致状态 Ω 的状态组合。

我们接下来将根据所考虑节点 t 的类型，给出计算每个 $t \in V(\hat{T}_\mu)$、t-签名f、$\bar{c} \in [0, \mathbf{c}(V)]$、$\bar{w} \in [0, \mathbf{w}(V)]$ 的 $A^{c}{}_{,t}{}^{w}(f)$ 递推式。

引入节点。如果对于某个 $\ell \in [k]$，有 $t = \ell(v)$，那么 $L_t^{live} = \{\ell\}$ 并且

$$\begin{aligned}A_t^{\bar{c},\bar{w}}(f) = &[v \neq v_* \vee f(\ell) = \{\mathbf{1}_L\}]\\ &\cdot [(f(\ell) = \{\mathbf{0}\} \wedge \bar{c} = \bar{w} = 0) \vee (f(\ell) \in \{\{\mathbf{1}_L\}, \{\mathbf{1}_R\}\} \wedge \bar{c} = \mathbf{c}(v) \wedge \bar{w} = \mathbf{w}(v))],\end{aligned}$$

因为在单点图中，任何单点状态的选择都会得到一个有效解；但如果 $v = v_*$，则只允许左侧为 v_* 的解。

重标记节点。如果 $t = \rho_{i\to j}(G_{t'})$，其中 t' 是 t 的子节点，对于某个 i、$j \in [k]$，那么根据 μ 的良好性可知，$i \in L_{t'}$、$j \in L_{t'}$、$L_t = L_{t'} \setminus \{i\}$，并且或者 i、$j \in L_{t'}^{live}$，或者 i、$j \in L_{t'}^{dead}$。如果标签 i 和 j 在 t' 处是活动的，那么标签 j 在 t 处是活动的，递推式为

$$A_t^{\bar{c},\overline{w}}(f) = \sum_{\substack{\mathbf{S}_1,\mathbf{S}_2\in\mathbf{States}:\\ \mathbf{S}_1\cup\mathbf{S}_2=f(j)}} A_{t'}^{\bar{c},\overline{w}}(f[i\mapsto \mathbf{S}_1, j\mapsto \mathbf{S}_2]),$$

因为 $V_t^j = V_{t'}^i \cup V_{t'}^j$，我们只需遍历标签 i 和 j 处所有可能的先前状态组合，从而得到所需状态 f（j）。

- 如果标签 i 和 j 在 t' 处是失活的，那么标签 j 在 t 处也是失活的。由于我们不跟踪失活标签的状态，因此可以直接复制前一个表，即，

$$A_t^{\bar{c},\overline{w}}(f) = A_{t'}^{\bar{c},\overline{w}}(f).$$

合并节点。为检查在其标签之间添加连接后，两个状态能否得到可行解，我们引入一个辅助函数 $\mathrm{feas}\colon$ **状态** $\times$ **状态** $\to\{0,1\}$，定义为 $\mathrm{feas}(\mathbf{S}_1,\mathbf{S}_2) = [\mathbf{0}\notin\mathbf{S}_1 \vee \mathbf{0}\notin\mathbf{S}_2][\mathbf{1}_L\in\mathbf{S}_1 \to \mathbf{1}_R\notin\mathbf{S}_2][\mathbf{1}_R\in\mathbf{S}_1\to\mathbf{1}_L\notin\mathbf{S}_2],$，或者等价地由下表定义：

feas	$\{\mathbf{0}\}$	$\{\mathbf{1}_L\}$	$\{\mathbf{1}_R\}$	$\{\mathbf{0},\mathbf{1}_L\}$	$\{\mathbf{0},\mathbf{1}_R\}$	$\{\mathbf{1}_L,\mathbf{1}_R\}$
$\{\mathbf{0}\}$	0	1	1	0	0	1
$\{\mathbf{1}_L\}$	1	1	0	1	0	0
$\{\mathbf{1}_R\}$	1	0	1	0	1	0
$\{\mathbf{0},\mathbf{1}_L\}$	0	1	0	0	0	0
$\{\mathbf{0},\mathbf{1}_R\}$	0	0	1	0	0	0
$\{\mathbf{1}_L,\mathbf{1}_R\}$	1	0	0	0	0	0

不可行有两个原因：一条连接边未被覆盖，即 $\mathbf{0}$ 出现在两侧；或者一条连接边连接了割的两侧，即 $\mathbf{1}_L$ 出现在一侧，而 $\mathbf{1}_R$ 出现在另一侧。利用这个辅助函数，现在可以陈述连接情形的递推式。

我们有 $t = \eta_{i,j}(G_{t'})$，其中某个 $i \neq j \in L_{t'}$，且 t' 是 t 的子节点。我们必须有 i，$j \in L_{t'}^{live}$；如果死顶点集合发生变化，即 $D_t \neq D_{t'}$，则后续的死节点会处理这一情况。因此，我们只需筛除因新的连接而变得不可行的所有部分解：

$$A_t^{\bar{c},\overline{w}}(f) = \mathrm{feas}(f(i),f(j))A_{t'}^{\bar{c},\overline{w}}(f).$$

死节点。我们有 $t = \bot_\ell(G_{t'})$，其中 t' 是 t，$\ell \notin L_t^{live}$ 的子节点，且 $L_t^{live} = L_{t'}^{live}\setminus\{\ell\}$。由于唯一的变化是 t-签名不再追踪标签 ℓ 的状态，我们必须将标签 ℓ 所有先前状态的贡献相加。因此，递推式为

$$A_t^{\bar{c},\overline{w}}(f) = \sum_{\mathbf{S}\in\mathbf{States}} A_{t'}^{\bar{c},\overline{w}}(f[\ell\mapsto\mathbf{S}]).$$

并集节点。我们有 $t = G_{t_1}\oplus G_{t_2}$，其中 t_1 和 t_2 是 t 的子节点，并且有 $L_t^{live} = L_{t_1}^{live}\cup L_{t_2}^{live}$。给定一个 t-签名 f，我们考虑 t处 f 的并分割 $f_{t,1}, f_{t,2}, f_{t,12}$，参见定义 3.17。对于每个标签 $\ell \in \tilde{L}_{t,12} = L_{t_1}^{live}\cap L_{t_2}^{live}$，我们需要考虑所有状态 $\mathbf{S}_1,\mathbf{S}_2 \in$ **状态**，使得 $\mathbf{S}_1\cup\mathbf{S}_2 = f(\ell)$，其中 $\mathbf{S}_i$ 是 t_i处标签 ℓ 的状态。

此外，我们还必须将 c 和 w 分配到 t_1 处的部分解以及 t_2 处的部分解中。因此，我们得到递推式

$$A_t^{\bar{c},\bar{w}}(f) = \sum_{\substack{\bar{c}_1+\bar{c}_2=\bar{c}\\ \bar{w}_1+\bar{w}_2=\bar{w}}} \sum_{\substack{g_1,g_2\colon \tilde{L}_{t,12}\to\mathbf{States}:\\ g_1(\ell)\cup g_2(\ell)=f_{t,12}(\ell)\ \forall \ell\in\tilde{L}_{t,12}}} A_{t_1}^{\bar{c}_1,\bar{w}_1}(g_1\cup f_{t,1}) A_{t_2}^{\bar{c}_2,\bar{w}_2}(g_2\cup f_{t,2}),$$

其中，我们将 g_i 和 $f_{t,i}$ 视为 $g_i \cup f_{t,i}$ 中的集合，其中 $i \in [2]$。

现在我们说明，如何在固定 $c \in [0, \mathsf{c}(V)]$、$w \in [0, \mathsf{w}(V)]$ 的情况下，将该递推式在时间 $O^*(6^{|L_t^{live}|})$ 内计算出来，并适用于所有 t-签名 f。首先，我们对 (c_1, c_2, w_1, w_2) 的所有可能情况进行分支；由于根据假设 $\mathsf{c}(V) \le n^{\mathcal{O}(1)}$ 且 $\mathsf{w}(V) \le 2n^2$，共有 $n^{\mathcal{O}(1)}$ 种可能情况。固定其中一种可能情况后，我们进一步对 $f_t,1$ 和 $f_t,2$ 进行分支，从而得到 $6^{|L_t,1|+|L_t,2|}$ 种选择。现在，量 $A^{c_1,w_1}_{t_1}(g_1 \cup f_t,1)$ 和 $A^{c_2,w_2}_{t_2}(g_2 \cup f_t,2)$ 可分别视为 g_1 和 g_2 的函数，而递推式中对它们的内层求和，就是在 $f_t,12$ 处求值的、关于状态的逐分量覆盖乘积。

由于集合 **状态** 显然是一个闭包差，我们可以应用定理 A.6，在时间 $\mathcal{O}^*(6^{|\tilde{L}_{t,12}|})$ 内计算所有可能的 $f_{t,12}$ 对应的逐分量覆盖乘积。由引理 3.18 可知，L_t^{live} 被划分为 $\tilde{L}_{t,1}$、$\tilde{L}_{t,2}$ 和 $\tilde{L}_{t,12}$，因此我们总共需要时间 $\mathcal{O}^*(6^{|L_t^{live}|})$，以计算所有 c、w 和 f 的取值下的 $A^{c,w}_t(f)$。

▶ **引理 4.1.** 给定 $G = (V, E)$ 的一个良好 k-表达式 μ，存在一种算法，可在时间 $O^*(6^k)$ 内，计算所有节点 $t \in V(T_\mu)$、所有 t-签名 f 以及所有 $c \in [0, \mathbf{c}(V)], w \in [0, 2n^2]$ 对应的量 $A^{c,w}_t(f)$。

证明。 该算法沿增广语法树 T_μ 对良好的 k-表达式 μ 执行自底向上动态规划，并根据给定的递推式计算量 $A^{c,w}_t(f)$。对于引入节点、重标记节点或合并节点，在固定 f、c 和 w 的情况下，$A^{c,w}_t(f)$ 的递推式显然可以在多项式时间内计算，因为 $\mathbb{Z}_2$ 中的加法和乘法均可在常数时间内完成。对于并集节点 t，我们已经说明，如何在 $\mathcal{O}^*(6^{|L_t^{live}|})$ 时间内同时计算所有 f、c 和 w 的递推式。由于 μ 是一个 k-表达式，对于所有 $t \in V(T_\mu)$，我们都有 $|L_t^{live}| \le k$，因此任意节点 $t \in V(T_\mu)$ 至多有 6^k 个签名。于是，运行时间的结论成立。

接下来只需证明这些递推式的正确性。引入节点、重标记节点、合并节点和并集节点的正确性证明是直接的，因而略去。考虑一个死节点 $t = \perp_\ell(G_{t'})$，其中 t' 是 t 的子节点。我们断言，对于所有 t-签名 f、c 和 w，有 $\mathcal{A}^{c,w}_t(f) = \bigcup_{\mathbf{S}\in\text{状态}} \mathcal{A}^{c,w}_{t'}(f[\ell \mapsto \mathbf{S}])$。由于右侧的并集显然是不相交的，该断言立即证明了死节点递推式的正确性。下面证明这一断言。右侧包含于左侧，因为左侧的集合由较少的约束定义。

对于该断言的另一个方向，假设存在某个 $(X, (X_L, X_R)) \in \mathsf{A}^{c,w}_t(f) \setminus \left(\bigcup_{\mathsf{S}\in\text{状态}} \mathsf{A}^{c,w}_{t'}(f[\ell \mapsto \mathsf{S}])\right)$ 并考虑 V^ℓ_t。由于 $V^\ell_t \neq \emptyset$，$V^\ell_t \not\subseteq X$、$X_L \cap V^\ell_t \neq \emptyset$、$X_R \cap V^\ell_t \neq \emptyset$ 中至少有一个必须成立。事实上，这三个陈述会同时成立，因为其余所有情况都由状态涵盖。考虑导致死节点 t 存在的合并节点 t^*。节点 t^* 添加了与 $V^\ell_t = V^\ell_{t^*}$ 关联的最后一个连接，即 V^ℓ_t 与某个 $V^i_t = V^i_{t^*}$, $i \neq \ell$ 之间的连接；由 μ 的良好性可知，$V^i_t \neq \emptyset$。因此，$V^i_t \not\subseteq X$、$X_L \cap V^i_t \neq \emptyset$、$X_R \cap V^i_t \neq \emptyset$ 中至少有一个必须成立。于是，V^ℓ_t 与 V^i_t 之间存在一条未覆盖边，或者存在一条跨越割 (X_L, X_R) 的边，这与 $(X, (X_L, X_R)) \in \mathsf{A}^{c,w}_t(f)$ 矛盾。因此，$\mathsf{A}^{c,w}_t(f) \subseteq \bigcup_{\mathsf{S}\in\text{状态}} \mathsf{A}^{c,w}_{t'}(f[\ell \mapsto \mathsf{S}])$，从而证明了该断言。

◀

▶**定理 4.2**。 存在一种随机化算法，给定图 $G=(V,E)$ 的良好 k-表达式 μ，可在 $O^*(6^k)$ 时间内求解连通顶点覆盖。该算法不会返回假阳性，且返回假阴性的概率至多为 $1/2$。

证明。 我们首先从$\mathbf{w}\colon V\to[2n]$ 中均匀随机抽取一个权重函数。然后，我们从 G 中选取一条边，并对其端点进行分支；选定的端点在当前分支中充当 v_*的角色。接着，我们运行引理 4.1 中的算法，计算量$A^{c,w}_{r,t}(f)$。令$\hat{r}$ 表示表达式 μ的根节点。在根节点处，我们有 $L^{live}_{\hat{r}}=\emptyset$。如果某个分支中存在 $c\le b$, $w\in[0,2n^2]$的某种选取，使得 $A^{c,w}_{\hat{r}}(\emptyset)\not\equiv 0$，则算法返回 true；否则返回 false。

运行时间直接由引理 4.1 得出。关于正确性，首先注意到在根节点处，我们有$\mathcal{A}^{c,w}_{\hat{r}}(\emptyset)=\mathcal{A}^{c,w}_{\hat{r}}$。只有在存在 $c\in[0,b]$, $w\in[0,2n^2]$ 使得 $\mathcal{A}^{c,w}_{\hat{r}}$ 具有奇数基数时，算法才会返回 true。根据推论 3.6，这意味着存在 G 的一个连通顶点覆盖 X，且$\mathbf{c}(X)=c\le b$，因此算法不会产生假阳性。

关于错误概率，假设权重函数 $\mathbf{w}$ 将 X^* 的最优连通顶点覆盖 G 隔离出来，并且 $\mathbf{c}(X^*)\le b$；根据引理 3.3，该隔离以大于或等于 $1/2$ 的概率发生。此外，考虑一个满足 $v_*\in X^*$ 的分支。令 $c=\mathbf{c}(X^*)\le b$ 且 $w=\mathbf{w}(X^*)$。根据引理 3.5，该连通顶点覆盖 X^* 对 $|\mathcal{A}^{c,w}_r|$，贡献奇数，而由于隔离，其他所有作出贡献的集合都不可能是连通的，因此对 $|\mathcal{A}^{c,w}_r|$，贡献偶数。因此 $A^{c,w}_{\hat{r}}(\emptyset)\equiv_2|\mathcal{A}^{c,w}_r|=1\neq 0$，由此在给定肯定实例的情况下，算法以至少 $1/2$ 的概率返回 true。◀

4.2 连通支配集

在连通支配集问题中，给定一个图 $G=(V,E)$、一个代价函数 $\mathbf{c}\colon V\to\mathbb{Z}_{>0}$ 和一个非负整数 b，任务是判断是否存在一个顶点集合 $X\subseteq V$，满足 $\mathbf{c}(X)\le b$，使得 X 是 G 的支配集，即$N[X]=V$，并且 $G[X]$ 是连通的。我们只考虑代价在输入规模上多项式有界的连通支配集实例。此外，我们假设 G 是连通的，并且至少包含两个顶点。

我们首先说明用于求解连通支配集的算法方法。沿用连通顶点覆盖的思路，我们会考虑部分解（X，(X_L,X_R)），它由部分支配集 X 和由 X 诱导的子图上的一致割（X_L,X_R）组成。对于（X，(X_L,X_R)），单个顶点 v 可以具有四种状态：$\mathbf{01}$、$\mathbf{00}$、$\mathbf{1}_L$、$\mathbf{1}_R$；其中前两种表示 $v\notin X$，下标表示v 是否被 X 支配，后两种表示 $v\in X$，下标表示包含 v 的割侧。我们同样可以存储每个标签中出现的顶点状态，从而所有 $\{\mathbf{01},\mathbf{00},\mathbf{1}_L,\mathbf{1}_R\}$ 的子集都可能成为标签状态。注意到状态$\mathbf{0}_1$ 不会对后续的连接施加任何约束，我们甚至可以认为只需考虑 $\{\mathbf{00},\mathbf{1}_L,\mathbf{1}_R\}$ 的子集。此外，与连通顶点覆盖类似，标签状态 $\{\mathbf{00},\mathbf{1}_L,\mathbf{1}_R\}$ 无法由活动标签合理地达到，因此每个活动标签只剩下七种状态。该方法的运行时间为 $O^*(7^{\mathrm{cw}(G)})$，但我们还可以获得更快的算法。

为获得改进后的运行时间$\mathcal{O}^*(5^{\mathrm{cw}(G)})$，我们转而使用一组不同的顶点状态，这组状态常用于支配问题 [26, 41, 43, 45]。我们不再考虑包含被支配和未被支配顶点的部分解，而是考虑允许顶点（状态$\mathbf{A}$）和禁止顶点（状态$\mathbf{F}$）。顾名思义，允许顶点可以被支配，也可以未被支配，但禁止顶点不能被支配。在将

顶点状态提升为标签状态时，可以忽略状态**A** ，因为它不会对连接施加任何约束，因此我们得到{**F**，**L**，**R**} 的子集作为标签状态。这组状态的优点在于，所有大小至少为二的子集在连接方面的表现都相同，因此可以将它们合并为一个状态；而且处理连接时无需将状态从未被支配更新为被支配。由此便得到所需的五种标签状态。

然而，要从这组状态中恢复原问题的解，通常需要某种容斥论证。对于团宽而言，这一步的应用并不常见。对于树宽等稀疏图参数，容斥论证可以应用于单个 顶点；也就是说，如果我们从顶点 v 处于状态 **F** 的部分解中，减去顶点 v 处于状态 **A**的部分解，那么剩下的就只有支配了 v 的部分解。然而，对于团宽，我们必须将这一论证应用于顶点组，这样的减法只能得出该标签中的某些 顶点被支配，而无法得出我们所希望的所有顶点都被支配。此外，将多个标签状态合并为一个状态，会使容斥论证更加复杂。令人惊讶的是，在模 2 下运算可以同时解决所有这些问题，而且这也是割与计数技术的自然框架。最后，只有当与某个标签关联的所有边都已构造完成时，才应应用容斥论证；也就是说，所考虑的标签已失活，因此我们再次使用增广语法树。

下面给出该算法的形式化细节。给定一个 k-表达式 μ for$G = (V, E)$，经过多项式时间预处理后，我们可以假设 μ 是良构的，参见引理 3.10。我们为隔离引理采样一个权重函数 $\mathbf{w}\colon V \to [2n]$ ，参见引理 3.3。为求解连通支配集，我们沿着 μ的增广语法树 T_μ 执行自底向上动态规划。现在我们选取某个 $v_* \in V$，并且只考虑包含v_* 的解。

为实现容斥原理和割计数方法，动态规划算法在节点 $t \in V(\hat{T}_\mu)$ 处考虑以下部分解族。

▶ **定义 4.3**。 在节点 $t \in V(\hat{T}_\mu)$ 处，部分解$\mathcal{A}_t$ 族由 V_t 的所有有序子划分 (X_L, X_R, F) 构成，并满足以下性质：

- $X = X_L \cup X_R$，(X_L, X_R) 是 $G_t[X]$,$v_* \in X_L$ 的一致割，当且仅当 $v_* \in V_t$,
- $N_{G_t}[X] \cap F = \emptyset$,$V_t^\ell \subseteq N_{G_t}[X]$ 对所有 $\ell \in L_t^{dead}$ 都成立。此外，对于每个节点 $t \in V(\hat{T}_\mu)$，给定 $c \in [0,\mathbf{c}(V)]$, 和 $w \in [0,\mathbf{w}(V)]$, ，我们定义 $\mathcal{A}_t^{c\ w} =$ $\{(X_L, X_R, F) \in \mathcal{A}_t : \mathbf{c}(X_L \cup X_R) = c, \ \mathbf{w}(X_L \cup X_R) = w\}$。

本质上，每个 $(X_L, X_R, F) \in \mathcal{A}_t$ 都由一个经过一致切割的部分支配集$X = X_L \cup X_R$ 构成，该集合支配所有带有死标签的顶点，且不支配 F 中的顶点。对于任意 $(X_L, X_R, F) \in \mathcal{A}_t$，还存在一个与之关联的顶点集$A = V_t \setminus (X_L \cup X_R \cup F)$，其中的顶点可以被支配。

请注意，对于任意 $(X_L, X_R, F) \in \mathcal{A}_t$，必有 $F \cap \left(\bigcup_{\ell \in L_t^{dead}} V_t^\ell\right) = \emptyset$，否则定义 $\mathcal{A}_t$ 中的第三和第四个性质无法同时满足。特别地，对于根节点 $\hat{r}$，必有 $F = \emptyset$ 和 $L_{\hat{r}}^{dead} = L_{\hat{r}}$，因此 $\mathcal{A}_{\hat{r}}$ 仅包含 G 的一致切割支配集。因此，如果存在 $c \in [0, \underline{b}]$、$w \in [0, 2n^2]$，使得$\mathcal{A}^{c,w}{}_{\hat{r}}$ 具有奇数基数，那么根据推论 3.6，存在 G 的一个连通支配集 X，满足 $\mathbf{c}(X) \leq \underline{b}$ 。

为了通过动态规划计算集合$\mathcal{A}_t^{c\ w}$，我们根据部分解在活标签 L_t^{live} 上的状态对其进行划分。部分解的状态 $(X_L, X_R, F) \in$

$\mathcal{A}_t$ 在标签 $\ell \in L_t^{live}$ 处取决于 V_t^ℓ 与集合 X_L、X_R 和 F 中哪些集合相交。为此，我们作如下定义。

▶ **定义 4.4.** 设 $t \in V(\hat{T}_\mu)$ 是 $\hat{T}_\mu$ 的一个节点，$\ell \in L_t$ 是一个非空标签。给定 V_t 的一个有序子划分 (X_L, X_R, F)，我们将集合 $\mathbf{S}_t^\ell(X_L, X_R, F) \subseteq \{\mathbf{F}, \mathbf{L}, \mathbf{R}\}$ 定义为

- $\mathbf{F} \in \mathbf{S}_t^\ell(X_L, X_R, F) \iff F \cap V_t^\ell \neq \emptyset$,
- $\mathbf{L} \in \mathbf{S}_t^\ell(X_L, X_R, F) \iff X_L \cap V_t^\ell \neq \emptyset$,
- $\mathbf{R} \in \mathbf{S}_t^\ell(X_L, X_R, F) \iff X_R \cap V_t^\ell \neq \emptyset$。

朴素地说，使用集合 $\mathsf{S}_t^\ell(X_L, X_R, F)$ 会得到每个标签 $2^3 = 8$ 个状态。注意，对于 $(X_L, X_R, F) \in \mathsf{A}_t$ 和非空标签 $\ell \in L_t$，$\mathsf{S}_t^\ell(X_L, X_R, F) = \emptyset$ 也是合理的，因为这仅意味着 $V_t^\ell \subseteq A = V_t \setminus (X_L \cup X_R \cup F)$，也就是说，$V_t^\ell$ 中的所有顶点都可以被支配，且不属于部分支配集 $X_L \cup X_R$。令人惊讶的是，结果表明，所有满足 $|\mathsf{S}| \geq 2$ 的 $\mathsf{S} \subseteq \{\mathsf{F}, \mathsf{L}, \mathsf{R}\}$ 都可以用相同的方式处理，因为所有这些 S 都只能与状态 $\emptyset$ 进行可行连接。这样一来，我们便可以用每个标签 5 个状态而不是 8 个状态来求解连通支配集问题。我们定义状态集合 $= \{\emptyset, \{\mathsf{F}\}, \{\mathsf{L}\}, \{\mathsf{R}\}\ \mathbf{2}_+\}$，其中 $\mathbf{2}_+$ 不是 $\{\mathsf{F}, \mathsf{L}, \mathsf{R}\}$ 的子集，而是一个形式符号，表示大小至少为 2 的子集。

▶ **定义 4.5**。给定一个节点 $t \in V(\hat{T}_\mu)$，t-签名是一个函数 $f\colon L_t^{live} \to$ **状态**。

X_L、X_R、F 对 V_t 的一个子划分与 t-签名 f 兼容，当且仅当对所有 $\ell \in L_t^{live}$，以下两个性质均成立：$\mathsf{S}_t^\ell(X_L, X_R, F) = f(\ell)$，若 $f(\ell) \neq \mathbf{2}_+$；$|\mathsf{S}_t^\ell(X_L, X_R, F)| \geq 2$，若 $f(\ell) = \mathbf{2}_+$。此外，对于每个节点 $t \in V(\hat{T}_\mu)$、t-签名 f、$c \in [0, \mathbf{c}(V)]$，和 $w \in [0, \mathbf{w}(V)]$，我们定义 $\mathsf{A}_t^{c,w}(f) = \{(X_L, X_R, F) \in \mathsf{A}_t^{c,w} : (X_L, X_R, F)$ 与 f 兼容$\}$。

与连通顶点覆盖不同，在连通支配集中，不存在只能出现在死标签上的状态，因为状态 $\emptyset$ 可以与任意状态连接，而不会使部分解变得不可行。尽管如此，区分活标签和死标签仍然有用，因为我们只要求支配具有死标签的顶点；具有活标签的顶点可能由当前节点中缺失的一条边支配。因此，T_μ 的 $\hat{}$ 死节点是应用容斥步骤的自然节点。

和往常一样，我们不直接计算集合 $\mathcal{A}_t^{c,\overline{w}}(f)$，而是计算其基数的奇偶性，即 $A^{c,w}_t(f) = |\mathcal{A}_t^{c,\overline{w}}(f)|$ 模 2。接下来，我们将根据节点 t 的类型，给出节点 $t \in V(\hat{T}_\mu)$、t-签名 f，$c \in [0, \mathbf{c}(V)]$，$w \in [0, \mathbf{w}(V)]$，对应的 $A^{c,w}_t(f)$ 的各种递推式。

引入节点。如果对于某个 $\ell \in [k]$ 和 $v \in V$，$t = \ell(v)$ 成立，那么 $L_t^{live} = \{\ell\}$，因为根据假设，v 不可能是孤立顶点，并且

$$A_t^{\overline{c},\overline{w}}(f) = [v \neq v_* \vee f(\ell) = \{\mathbf{L}\}] \cdot [(\overline{c} = \overline{w} = 0 \wedge f(\ell) \in \{\emptyset, \{\mathbf{F}\}\}) \vee (\overline{c} = \mathbf{c}(v) \wedge \overline{w} = \mathbf{w}(v) \wedge f(\ell) \in \{\{\mathbf{L}\}, \{\mathbf{R}\}\})],$$

首先检查边的情况，即 $v = v_*$，然后检查 c 和 w 是否与选定的状态一致。请注意，状态 $\mathbf{2}_+$ 在此处无法实现。

重标记节点。如果 $t = \rho_{i \to j}(G_{t'})$，其中 t' 是 t 的唯一子节点，那么在节点 t' 处，标签 i 和 j 均非空，因为 μ 是一个良好表达式。根据表达式 μ 的不可冗余性，要么 $\{i, j\} \subseteq L_{t'}^{dead}$，要么 $\{i, j\} \cap L_{t'}^{dead} = \emptyset$。在第一种情况下，我们有 $j \in L_t^{dead}$，并且

递推式就是 $A^{c,w}_t(f) = A^{c,w}_{t'}(f)$，因为我们不存储死标签的状态，而且死标签的支配要求仍然得到满足。

在第二种情况下，我们有 $j \in L^{live}_t$，并且必须遍历所有能够合并为标签 j 处当前状态的状态对。由于 $V^j_t = V^i_{t'} \cup V^j_{t'}$，递推式为

$$A^{\overline{c},\overline{w}}_t(f) = \sum_{\substack{\mathbf{S}_1,\mathbf{S}_2 \in \mathbf{States}: \\ \mathrm{merge}(\mathbf{S}_1,\mathbf{S}_2)=f(j)}} A^{\overline{c},\overline{w}}_{t'}(f[i \mapsto \mathbf{S}_1, j \mapsto \mathbf{S}_2]),$$

其中 合并：**状态** $\times$ **状态** $\to$ **状态** 由下式给出：

合并	$\emptyset$	$\{\mathbf{F}\}$	$\{\mathbf{L}\}$	$\{\mathbf{R}\}$	$\mathbf{2}_+$
$\emptyset$	$\emptyset$	$\{\mathbf{F}\}$	$\{\mathbf{L}\}$	$\{\mathbf{R}\}$	$\mathbf{2}_+$
$\{\mathbf{F}\}$	$\{\mathbf{F}\}$	$\{\mathbf{F}\}$	$\mathbf{2}_+$	$\mathbf{2}_+$	$\mathbf{2}_+$
$\{\mathbf{L}\}$	$\{\mathbf{L}\}$	$\mathbf{2}_+$	$\{\mathbf{L}\}$	$\mathbf{2}_+$	$\mathbf{2}_+$
$\{\mathbf{R}\}$	$\{\mathbf{R}\}$	$\mathbf{2}_+$	$\mathbf{2}_+$	$\{\mathbf{R}\}$	$\mathbf{2}_+$
$\mathbf{2}_+$	$\mathbf{2}_+$	$\mathbf{2}_+$	$\mathbf{2}_+$	$\mathbf{2}_+$	$\mathbf{2}_+$

注意，$\mathbf{S}^j_t(X_L, X_R, F) = \mathbf{S}^i_{t'}(X_L, X_R, F) \cup \mathbf{S}^j_{t'}(X_L, X_R, F)$；若标签 i 在 t' 处具有状态 $\mathbf{S}_1$，且标签 j 在 t' 处具有状态 $\mathbf{S}_2$，则标签 j 在节点 t 处的状态为 $\mathrm{merge}(\mathbf{S}_1,\mathbf{S}_2)$。

合并节点。如果 $t = \eta_{i,j}(G_{t'})$，$i \neq j$，其中 t' 是 t 的唯一子节点，那么我们知道 $i, j \in L^{live}_{t'}$，因为表达式 μ 是良好的。由于这次连接，标签为 i 或 j 的顶点可能会获得其最终关联边，从而可能导致 $V^i_t \subseteq D_t$ 或 $V^j_t \subseteq D_t$。如果发生这种情况，此合并节点之后最多会跟随两个死节点。在此合并节点处，我们筛除因新添加的边而失效的部分解，递推式为

$$A^{\overline{c},\overline{w}}_t(f) = \mathrm{feas}(f(u), f(v)) A^{\overline{c},\overline{w}}_{t'}(f),$$

其中 feas：**状态** $\times$ **状态** $\to \{0, 1\}$ 由下表给出：

feas	$\emptyset$	$\{\mathbf{F}\}$	$\{\mathbf{L}\}$	$\{\mathbf{R}\}$	$\mathbf{2}_+$
$\emptyset$	1	1	1	1	1
$\{\mathbf{F}\}$	1	1	0	0	0
$\{\mathbf{L}\}$	1	0	1	0	0
$\{\mathbf{R}\}$	1	0	0	1	0
$\mathbf{2}_+$	1	0	0	0	0

部分解（X_L，X_R，F）可能因两种原因而失效；一条新边连接了 F 和 $X = X_L \cup X_R$，或者一条新边连接了 X_L 和 X_R。

死节点。假设 $t = \bot_\ell(G_{t'})$，其中 $\ell \in L'_t$ 和 t' 是 t 的子节点。我们有 $L^{live}_t = L^{live}_{t'} \setminus \{\ell\}$ 和 $\ell \in L^{dead}_t$。根据 $\mathcal{A}_t$ 的定义，我们只想计算 $\mathcal{A}_{t'}$ 中完全支配 V^ℓ_t 的部分解。如果 V^ℓ_t 只包含一个顶点，那么利用给定的状态就很容易检查：我们只需计算一个 t-签名 f

$$A^{\overline{c},\overline{w}}_{t'}(f[\ell \mapsto \{\mathbf{L}\}]) + A^{\overline{c},\overline{w}}_{t'}(f[\ell \mapsto \{\mathbf{R}\}]) + (A^{\overline{c},\overline{w}}_{t'}(f[\ell \mapsto \emptyset]) - A^{\overline{c},\overline{w}}_{t'}(f[\ell \mapsto \{\mathbf{F}\}])),$$

其中最后一部分是容斥论证，用于检查 V^ℓ_t 中的顶点是否被支配。然而，如果 $|V^\ell_t| \geq 2$，那么我们还需要处理状态 $\mathbf{2}_+$；对于这种状态，我们不清楚该如何将其纳入容斥论证，因为在这种情况下，我们甚至不知道 $\mathbf{S}^\ell_t(X_L, X_R, F)$ 的确切值。此外，前面的

如果 $|V_t^\ell| \geq 2$，那么容斥论证在 $\mathbb{Z}$ 上无效，因为 V_t^ℓ 中含有多个未被支配顶点的部分解会被 $A^{c,}{}_{t'}{}^{w}(f[\ell \mapsto \{\mathbf{F}\}])$ 重复计数。

令人惊讶的是，存在一个非常简单的递推式，可以在模 2 下避免所有这些问题。如果 f 是一个 t-签名，且 $c \in [0,n]$、$w \in [0,2n^2]$，那么递推式如下：

$$A_t^{c,w}(f) = \sum_{\mathbf{S}\in\mathbf{States}} A_{t'}^{c,w}(f[\ell \mapsto \mathbf{S}]).$$

这个递推式之所以成立，是因为任何恰好包含 u 个未被支配顶点的 V_t^ℓ 中的部分支配集，在递推式右侧都会被计数 2^u 次，因此在模 2 下对于 $u > 0$ 会相互抵消。接下来，我们将给出该递推式正确性的形式化证明。

证明。 固定 f、c 和 w。左侧计算 $\mathcal{A}_t^{c,w}(f)$ 的基数模 2，而右侧显然计算出 $\sum_{\mathbf{S}\in\text{状态}} |\mathcal{A}_{t'}^{c,w}(f[\ell \mapsto \mathbf{S}])|$ 模 2。我们需要证明这两项相等。为便于阅读，令 $\mathcal{A}_{LHS} := \mathcal{A}_t^{c,w}(f)$，并令 $\mathcal{A}_{RHS} := \bigcup_{\mathbf{S}\in\text{状态}} \mathcal{A}_{t'}^{c,w}(f[\ell \mapsto \mathbf{S}])$。

回想一下，对于所有 $i \in L_t$，$G_t = G_{t'}$, $V_t^i = V_{t'}^i$，以及满足 $\ell \notin L_{t'}^{dead}$ 的 $L_t^{dead} = L_{t'}^{dead} \cup \{\ell\}$。首先，注意 $\mathrm{A}_{LHS} \subseteq \mathrm{A}_{RHS}$，因为对于 $\mathrm{S}_{t'}^{\ell}(X_L, X_R, F)$ 的每种可能性，都存在一个 $\mathrm{S} \in$ 状态，使得 (X_L, X_R, F) 与节点 t' 处的 $f[\ell \mapsto \mathrm{S}]$ 兼容。还要注意，对于不同的 $\mathrm{S} \in$ 状态，集合 $\mathrm{A}_{t'}^{c,w}(f[\ell \mapsto \mathrm{S}])$ 两两不相交，因此 A_{RHS} 中的每个元素在等式右侧都恰好被计数一次。

接下来，我们看到，对于任意 $(X_L, X_R, F) \in \mathcal{A}_{RHS}$，$(X_L, X_R, F) \in \mathcal{A}_{LHS}$ 成立，当且仅当 $V_t^\ell \subseteq N_{G_t}[X]$，其中 $X = X_L \cup X_R$，因为发生变化的唯一要求是与 f 的兼容性；在 t 处比在 t' 处更容易满足这一要求。此外，还要求所有死标签都被支配，这除了 $(X_L, X_R, F) \in \mathcal{A}_{RHS}$ 外，还要求 $V_t^\ell \subseteq N_{G_t}[X]$。

接下来只需证明 $\mathrm{A}_{RHS} \setminus \mathrm{A}_{LHS}$ 包含偶数个元素，因此在模 2 下抵消。考虑 $(X_L, X_R, F) \in \mathrm{A}_{RHS} \setminus \mathrm{A}_{LHS}$，令 $X = X_L \cup X_R$，并令 $U_t^\ell := U_t^\ell(X_L, X_R) := V_t^\ell \setminus N_{G_t}[X]$ 为节点 t 处标签为 ℓ 的未被支配顶点集合。由 $(X_L, X_R, F) \in \mathrm{A}_{RHS} \setminus \mathrm{A}_{LHS}$ 可知 $U_t^\ell \neq \emptyset$，否则 $V_t^\ell \subseteq N_{G_t}[X]$，这与前一段中的 $(X_L, X_R, F) \notin \mathrm{A}_{LHS}$ 矛盾。对于任意 $F' \subseteq U_t^\ell$，我们有 $(X_L, X_R, (F \setminus V_t^\ell) \cup F') \in \mathrm{A}_{RHS} \setminus \mathrm{A}_{LHS}$，因为 U_t^ℓ 保持不变，我们只改变了哪些顶点被声明为禁用。由于 $U_t^\ell \neq \emptyset$，对于 F' 共有偶数种选择，即 $2^{|U_t^\ell|}$ 种。

固定某个 $\emptyset \neq U \subseteq V_t^\ell$ 后，这表明元素 $(X_L, X_R, F) \in \mathcal{A}_{RHS} \setminus \mathcal{A}_{LHS}$ 的数量为偶数，且满足 $U_t^\ell(X_L, X_R) = U$。由于 $\mathcal{A}_{RHS} \setminus \mathcal{A}_{LHS}$ 中的每个元素都被某个 U 的选择覆盖，因此 $\mathcal{A}_{RHS} \setminus \mathcal{A}_{LHS}$ 的基数为偶数。◀

并集节点。我们有 $t = G_{t_1} \oplus G_{t_2}$，其中 t_1 和 t_2 是 t 的子节点，并且有 $L_t^{live} = L_{t_1}^{live} \cup L_{t_2}^{live}$。给定一个 t-签名 f，我们考虑 f 在 t 处的并分割 $f_{t,1,2}$，参见定义 3.17。对于每个标签 $\ell \in \tilde{L}_{t,12} = L_{t_1}^{live} \cap L_{t_2}^{live}$，我们需要考虑所有满足 $\mathrm{merge}(\mathrm{S}_1, \mathrm{S}_2) = f(\ell)$ 的状态 $\mathrm{S}_1, \mathrm{S}_2 \in \mathrm{States}$，其中 S_i 是 t_i 处标签为 ℓ 的状态。对于两个函数 $g, h\colon B \to \mathrm{States}$，其中 B 是某个集合，我们将 $\mathrm{merge}(g,h)\colon B \to \mathrm{States}$ 记为逐分量应用 merge，即对所有 $\ell \in B$，$\mathrm{merge}(g,h)(\ell) = \mathrm{merge}(g(\ell), h(\ell))$。此外，我们还必须将 c 和 w 分配到 t_1 处的部分解以及 t_2 处的部分解中，从而得到递推式

$$A_t^{c,w}(f) = \sum_{\substack{c_1+c_2=c \\ w_1+w_2=w}} \sum_{\substack{g_1,g_2\colon \tilde{L}_{t,12}\to\mathbf{States}\colon \\ \mathrm{merge}(g_1,g_2)=f}} A_{t_1}^{c_1,w_1}(g_1 \cup f_{t,1}) A_{t_2}^{c_2,w_2}(g_2 \cup f_{t,2}),$$

其中，我们将 g_i 和 $f_{t,i}$ 视为 $g_i \cup f_{t,i}$ 中的集合，适用于 $i \in [2]$。在这里，对于仅在一个子节点中处于活动状态的标签，我们简单地使用部分 $f_{t,1}$ 和 $f_{t,2}$ 来固定其状态；而对于在两个子节点中都处于活动状态的标签，我们对所有有效的状态组合求和。

现在我们说明，对于固定的 $c \in [0, \mathbf{c}(V)]$、$w \in [0, \mathbf{w}(V)]$，以及所有 t-签名 f，如何在时间 $O^*(5^{|L_t^{live}|})$ 内计算该递推式。首先，我们对数值 c_1、$c_2 \in [0, \mathbf{c}(V)]$、$w_1$、$w_2 \in [0, \mathbf{w}(V)]$，以及函数 $h_1 \colon L_{t,1} \to$ 状态、$h_2 \colon L_{t,2} \to$ 状态进行分支。固定这些选择后，我们可以通过设置 $B_i(g) := A^{c_i}_{t_i, w_i}(g \cup h_i)$，为所有满足 $f_{t,1} = h_1$ 和 $f_{t,2} = h_2$ 的 t-签名 f 计算递推式的内层求和，其中 $i \in [2]$ 和 $g \colon L_{t,12} \to$ 状态，并计算 CDS 乘积

$$(B_1 \otimes_{CDS} B_2)(g) := \sum_{\substack{g_1, g_2 \colon \tilde{L}_{t,12} \to \mathbf{States} \colon \\ \mathrm{merge}(g_1, g_2) = g}} B_1(g_1) B_2(g_2)$$

对于所有 $g \colon L_{t,12} \to$ **状态**。根据即将给出的引理 4.6，我们可以在 $\mathcal{O}^*(5^{|L_{t,12}|})$ 时间内计算 CDS 乘积。由于有 $\mathcal{O}^*(5^{|L_{t,1}|+|L_{t,2}|})$ 个分支，并且根据引理 3.18，L_t^{live} 被划分为三个集合 $\tilde{L}_{t,1}$、$\tilde{L}_{t,2}$、$\tilde{L}_{t,12}$，因此我们需要 $\mathcal{O}^*(5^{|L_t^{live}|})$ 时间，为 c、w 和 f 的所有取值计算递推式。

▶ **引理 4.6**。给定两个表 B_1、B_2：**状态**$^I \to \mathbb{Z}_2$，其中 I 是某个索引集，它们的 *CDS* 乘积 $B_1 \otimes_{CDS} B_2$：**状态**$^I \to \mathbb{Z}_2$ 可以在 $O^*(|$**状态**$|^{|I|}) = O^*(5^{|I|})$ 时间内计算。

证明。考虑集合族 $\mathrm{L} = \{\emptyset, \{\mathrm{F}\}, \{\mathrm{L}\}, \{\mathrm{R}\}, \{\mathrm{F}, \mathrm{L}, \mathrm{R}\}\} \subseteq \mathrm{P}(U)$，其定义在 $U = \{\mathrm{F}, \mathrm{L}, \mathrm{R}\}$ 上，并由集合包含 $\subseteq$ 诱导出 L 上的偏序。容易验证，这构成一个格，因为任意一对元素都有最大下界（交/∧）和最小上界（连接/∨）。特别地，最小上界如下：

$\vee$	$\emptyset$	{F}	{L}	{R}		U
$\emptyset$	$\emptyset$	{F}	{L}	{R}		U
{F}	{F}	{F}		U	U	U
{L}	{L}	U	{L}	U	U	
{右}	{右}	U	U	{R}	U	
U	U	U	U	U	U	

双射 κ：**状态** $\to \mathcal{L}$，其中 $\kappa(\mathbf{S}) = \mathbf{S}$ 对于 $\mathbf{S} \neq \mathbf{2}_+$，且 $\kappa(\mathbf{2}_+) = U = \{\mathbf{F}, \mathbf{L}, \mathbf{R}\}$，将 **状态** 上的合并转换为 $\vee$ 在 $\mathcal{L}$ 上，即对于所有 $\mathbf{S}_1, \mathbf{S}_2 \in$ **状态**，有 $\kappa(\mathrm{merge}(\mathbf{S}_1, \mathbf{S}_2)) = \kappa(\mathbf{S}_1) \vee \kappa(\mathbf{S}_2)$。因此，我们可以写成

$$\begin{aligned}
(B_1 \otimes_{CDS} B_2)(g) &= \sum_{\substack{g_1, g_2 \colon I \to \mathbf{States} \colon \\ \mathrm{merge}(g_1, g_2) = g}} B_1(g_1) B_2(g_2) \\
&= \sum_{\substack{h_1, h_2 \colon I \to \mathcal{L} \colon \\ h_1(i) \vee h_2(i) = \kappa(g(i)) \ \forall i \in I}} B_1(\kappa^{-1} \circ h_1) B_2(\kappa^{-1} \circ h_2) \\
&= \sum_{\substack{h_1, h_2 \in \mathcal{L}^I \colon \\ h_1 \vee h_2 = \kappa \circ g}} B_1(\kappa^{-1} \circ h_1) B_2(\kappa^{-1} \circ h_2) = \sum_{\substack{h_1, h_2 \in \mathcal{L}^I \colon \\ h_1 \vee h_2 = \kappa \circ g}} B_1'(h_1) B_2'(h_2) \\
&= (B_1' \otimes_{\mathcal{L}^I} B_2')(\kappa \circ g),
\end{aligned}$$

其中 $B_j'(h) = B_j(\kappa^{-1} \circ h)$ 对于 $h \in \mathcal{L}^I$，$j = 1, 2$，且 $\vee$ 是 $\mathcal{L}$ 或 $\mathcal{L}^I$ 中的连接，具体取决于上下文。通过以自然方式将 $\mathcal{L}^I$ 与 $\mathcal{L}^{|I|}$ 对应起来，我们因此可以应用推论 A.10，在 $\mathcal{O}^*(|\mathcal{L}|^{|I|}) = \mathcal{O}^*(5^{|I|})$ 时间内计算 CDS 乘积，因为对 $\mathcal{A}_{\mathcal{L}}$ 的调用可以在常数时间内得到应答。◀

▶**引理 4.7.** 给定图 G 的一个良好 k-表达式 μ，存在一种算法，可在 $O^*(5^k)$ 时间内，计算所有节点 $t \in V(T_\mu)$、所有 t-签名 f 以及所有 $c \in [0, \mathbf{c}(V)], w \in [0, 2n^2]$ 对应的量 $A^{c,w}_t(f)$。

证明。 该算法沿良好团表达式 μ 的增广语法树 T_μ，通过自底向上动态规划进行计算，并利用给定的递推式计算量 $A^{c,w}_t(f)$。对于引入节点、重标记节点、合并节点或死节点，固定 f、c 和 w 后，显然可以在多项式时间内计算 $A^{c,w}_t(f)$ 的递推式，因为在 $\mathbb{Z}_2$ 中进行加法和乘法只需常数时间。对于并集节点 t，我们已经说明，如何在 $\mathcal{O}^*(5^{|L_t^{live}|})$ 时间内同时计算所有 f、c 和 w 的递推式。由于 μ 是一个 k-表达式，因此对于所有 $t \in V(T_\mu)$，均有 $|L_t^{live}| \le |L_t| \le k$；特别地，对任意节点 $t \in V(T_\mu)$，签名至多有 5^k 个。因此，运行时间结论成立。

接下来需要证明这些递推式的正确性。我们已经证明了死节点对应递推式的正确性。对于其他类型的节点，证明都很直接；下面着重说明其中一些有趣之处。

对于合并节点，我们强调，函数 $\mathrm{feas}: \mathbf{状态} \times \mathbf{状态} \to \{0,1\}$ 满足 $\mathrm{feas}(\mathbf{S}_1,\mathbf{S}_2) = 1 - [\mathbf{S}_1 \neq \emptyset][\mathbf{S}_2 \neq \emptyset][\mathbf{S}_1 = \mathbf{S}_2 = \mathbf{2}_+ \vee \mathbf{S}_1 \neq \mathbf{S}_2]$，它刻画了这样的状态对：在其底层标签之间添加一条连接后，会在 $X = X_L \cup X_R$ 与 F 之间形成一条边，或在割 (X_L, X_R) 两侧形成一条边，因此得到一个不可行解。

对于重标记节点和并集节点，我们强调，对于任意两个子集 $\mathrm{S}_1,\mathrm{S}_2 \subseteq \{\mathrm{F},\mathrm{L},\mathrm{R}\}$，都有 $\rho(\mathrm{S}_1 \cup \mathrm{S}_2) = \mathrm{merge}(\rho(\mathrm{S}_1)\ \rho(\mathrm{S}_2))$，其中 $\rho: \mathrm{P}(\{\mathrm{F},\mathrm{L},\mathrm{R}\}) \to$ 状态，且当 $|\mathrm{S}| \le 1$ 时，$\rho(\mathrm{S}) = \mathrm{S}$；当 $|\mathrm{S}| \ge 2$ 时，$\rho(\mathrm{S}) = \mathbf{2}_+$。因此，merge 能正确更新重标记节点和并集节点的状态。◀

▶**定理 4.8**。 存在一个随机化算法，给定图 $G = (V, E)$ 的一个良好 k-表达式 μ，可以在 $O^*(5^k)$ 时间内求解连通支配集问题。该算法不会返回假阳性，返回假阴性的概率至多为 $1/2$。

证明。 我们首先均匀随机地采样一个权重函数 $\mathrm{w}: V \to [2n]$。然后，任取一个顶点 v，并对其闭邻域 $N_G[v]$ 进行分支，因为每个支配集都至少与 $N_G[v]$ 中的一个顶点相交；所选顶点在当前分支中充当 v_* 的角色。接着，我们运行引理 4.7 中的算法来计算量 $A^{c,w}_t(f)$。在根处，有 $L^{live}_{\hat{r}} = \emptyset$。如果存在某个分支以及 $c \le b$、$w \in [0, 2n^2]$ 的某种选择，使得 $A^{c,w}_{\hat{r}}(\emptyset) \neq 0$，则算法返回 true；否则返回 false。

运行时间可直接由引理 4.7 得出。关于正确性，首先注意，在根处，我们有 $\mathrm{A}^{c,w}_{\hat{r}}(\emptyset) = \mathrm{A}^{c,w}_{\hat{r}}$。由于在 $\hat{r}$ 处所有标签均已失活，对于任意 $(X_L, X_R, F) \in \mathrm{A}_{\hat{r}}$，我们有 $(X, (X_L, X_R)) \in \mathrm{C}(G)$，$v_* \in X_L$，$V \subseteq N_G[X]$，因此 $F = \emptyset$，其中 $X = X_L \cup X_R$。所以，X 是 G 的一个支配集，且包含 v_*；反之，$\mathrm{A}_{\hat{r}}$ 包含此类支配集的所有一致割。该算法仅在存在某些 $c \in [0, b]$，$w \in [0, 2n^2]$ 使得 $\mathrm{A}^{c,w}_{\hat{r}}$ 的基数为奇数时才返回 true。定义 $\mathrm{R} = \{X \subseteq V: V \subseteq N_G[X], v_* \in X\}$ 和 $\mathrm{R}^{c,w} = \{X \in \mathrm{R}: |X| = c, \mathrm{w}(X) = w\}$，由推论 3.6 可知，这意味着存在一个 G 的连通支配集 X，满足 $\mathrm{c}(X) = c \le b$，因此该算法不会产生假阳性。

对于错误概率，假设权重函数 w 隔离了一个最优连通支配集 X^*，且 $\mathrm{c}(X^*) \le b$；根据引理 3.3，这种情况发生的概率为 $\ge 1/2$。在满足 $v_* \in X^*$ 的一个分支中，我们有 $X^* \in \mathrm{R}^{c,w}$，其中 $c = |X^*|$ 和 $w = \mathrm{w}(X^*)$。根据引理 3.5，该连通支配集对 $|\mathrm{A}^{c,w}_r|$，$\hat{}$ 的贡献为奇数，而所有其他贡献集合 $Y \in \mathrm{R}^{c,w}$ 都不可能连通，因为它们被隔离了，因此对 $|\mathrm{A}^{c,w}_r|$，$\hat{}$ 的贡献为偶数。因此 $A^{c,w}_{\hat{r}}(\emptyset) \equiv_2 |\mathrm{A}^{c,w}_r| = 1 \neq 0$，故对于一个肯定实例，算法返回 true 的概率至少为 $1/2$。◀

5 下界

构造原则。 本节中，我们证明以线性团宽为参数的连通顶点覆盖和连通支配集问题的紧下界。总体构造原则沿用了 Lokshtanov 等人 [37] 的风格。这意味着生成的图可以解释为一个块矩阵，其中每个块跨越若干行和列。每一行都是一条很长的类路径构件 ，它模拟可满足性实例中的常数个变量，并贡献 1 个线性团宽单位。模拟变量的数量取决于我们希望排除的运行时间。出于技术原因，我们考虑由若干行组成的束，用以模拟适当大小的变量组 。每一列对应一个子句，并由若干构件组成，这些构件解码 路径构件上的状态，并检查得到的赋值是否满足该子句；每列只考虑一个子句，可以加快构造低宽度图的过程。

路径构件。 在这两个下界中，主要的技术贡献是设计路径构件。而解码构件 的设计则可以改编自已知构造。每行与每列的交点处都有一个路径构件。目标是构造一种路径构件，使其针对目标问题能够容纳尽可能多的状态，因为这类状态的数量对应于我们能够证明下界的运行时间底数。由于每一行都应贡献一个线性团宽单位元，因此我们必须通过连接将一行中相邻的路径构件连接起来，从而限制了设计空间。

状态转移。 在证明目标问题的一个解能够导出可满足性问题的一个满足赋值时，我们要求状态沿每一行保持稳定。否则，如果允许状态发生变化，就可以为每个子句分别选取一个满足赋值，而这未必能得到一个同时满足所有子句的赋值。遗憾的是，通常无法构造出状态保持稳定的路径构件。不过，我们可以构造出状态只能以受控方式转移的路径构件。

控制状态转移。假设状态之间存在某种转移顺序，即只有当 $i \leq j$ 时，状态 i 才能转移到状态 j；这意味着沿着每一行，状态只能改变有限次。通过将行设置得足够长并重复子句，利用鸽巢原理，我们可以找到一个跨越所有子句的列区域，其中所有状态都保持稳定，从而为每个子句得到相同的赋值。因此，我们的目标是找到一个足够大的状态集合，以及一个允许这种转移顺序的合适路径构件。

确定转移顺序。 对于路径宽度这类稀疏宽度参数，确定适当的转移顺序要简单得多，因为可能的状态数量非常有限，例如，在所考虑的基准问题中，顶点状态最多只有四种。团宽对应的可能状态集合则大得多，通常我们需要选取一个特定的状态子集，因为并非所有状态都允许存在转移顺序。由于相邻路径构件通过连接相连，一些状态转移会因连接而被强制确定。因此，我们首先分析连接两侧可能的状态转移，并得到一个转移/兼容性矩阵，显示哪些状态对能够形成全局可行解，哪些不能。对兼容性矩阵的行和列重新排序后，转移顺序必须导出一个对角线上为 1 的三角子矩阵，其中

对角线上的 1 表示状态可以保持稳定。因此，兼容性矩阵中一个较大的三角子矩阵便成为我们设计适当路径构件的起点。

路径构件的结构。我们的路径构件由三部分组成：一个中心团，用于与解码构件通信；以及两个边界部分，即左侧和右侧，分别与前一个和后一个连接相连。在中心团中，每个解都会恰好避开一个顶点，该顶点代表路径构件的状态。为实现转移顺序，左侧和右侧必须向两个相邻的路径构件传递适当的状态。通过取三角子矩阵，并沿主对角线将状态配对（由于行和列的重新排序，这些配对可能不对称），我们可以看出每种情况下必须传递哪些状态。

设计路径构件的边界。部分解中的顶点状态由若干属性构成：该顶点是否包含在部分解中、一个连通性属性，以及该顶点是否被支配（对于连通支配集问题）。我们设计边界部分的策略是：隔离与连接相接的顶点的这些属性，并用成对的指示顶点表示每个属性是否满足。我们确保一个解在每对指示顶点中只能选取一个，从而使中心部分与边界部分之间能够进行简单的通信。至此，我们完成了路径构件构造的高层思路介绍。

根连通性。 为刻画连通顶点覆盖和连通支配集中的连通性约束，我们创建一个特殊顶点 $\hat{r}$ ，称为根 ，并通过连接一个度为 1 的顶点到 $\hat{r}$ ，确保每个连通顶点覆盖或连通支配集都必须包含 $\hat{r}$。给定一个顶点子集 $X \subseteq V(G)$ ，其中含有 $\hat{r} \in X$，如果存在一条 $v,\hat{r}$-路径，且该路径位于 $G[X]$ 中，我们就称顶点 $v \in X$ 在 X 中是与根连通 的。如果上下文明确，我们就简称与根连通 ，即 X 。图 $G[X]$ 连通，当且仅当 X 中的所有顶点在 X 中都与根连通。对于部分解 X 的状态而言，考虑哪些顶点在 X 中与根连通、哪些顶点没有与根连通十分重要。

5.1 连通顶点覆盖

本小节旨在证明，除非 CNF-SETH 不成立，否则对于某个 $\varepsilon > 0$ ，连通顶点覆盖（单位代价）无法在时间$\mathcal{O}^*((6-\varepsilon)^{\text{lin-cw}(G)})$ 内求解。我们首先按照概述中提出的思路设计路径构件，并单独分析该构件；随后给出完整构造及正确性证明。解码构件直接改编自 Cygan 等人 [17] 给出的、针对以路径宽度为参数的连通顶点覆盖的下界。

5.1.1 路径构件

为了排除任意 $\varepsilon > 0$ 下的运行时间 $\mathcal{O}^*((6-\varepsilon)^{\text{lin-cw}(G)})$，我们必须构造一个路径构件，它具有 6 种不同的状态，并将范围缩小到单个标签，从而使构造中的每一行都贡献一个线性团宽单位元。正如前文所述，我们首先分析部分解在一个标签上的可能行为。

每个单独的顶点 v 相对于部分解 X有 3 种状态之一： $v \notin X$ （状态**0**）、 $v \in X$ 与根连通（状态 $\mathbf{1}_1$）或不与根连通（状态 $\mathbf{1}_0$）。首先，我们注意到，每种顶点状态在一个标签中出现多少次并不重要；我们只关心某种顶点状态是否出现在标签中，还是

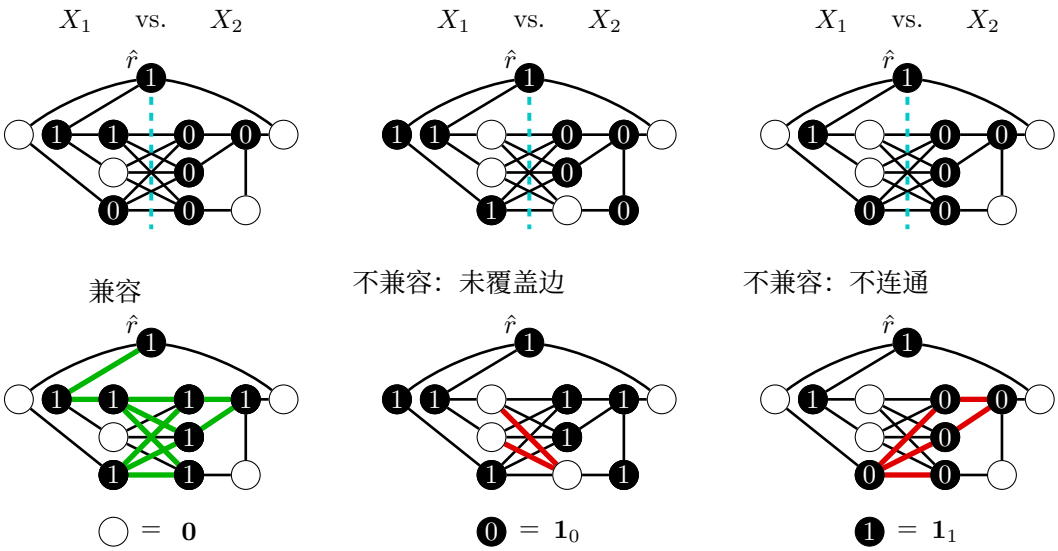

图 1 连接处部分解兼容性的几种情况。第一行展示了由虚线分隔的 X_1 和 X_2 中的顶点状态。第二行展示了 $X_1 \cup X_2$ 中的顶点状态，并从左到右依次突出显示诱导边、未覆盖边，以及一个不包含根 $\hat{r}$ 的连通分量。

X_1 与 X_2	$\{\mathbf{0}\}$	$\{\mathbf{1}_0\}$	$\{\mathbf{1}_1\}$	$\{\mathbf{1}_0, \mathbf{0}\}$	$\{\mathbf{1}_1, \mathbf{0}\}$	$\{\mathbf{1}_1, \mathbf{1}_0\}$	$\{\mathbf{1}_1, \mathbf{1}_0, \mathbf{0}\}$
$\{\mathbf{0}\}$	0	0	1	0	0	0	0
$\{\mathbf{1}_0\}$	0	0	1	0	1	1	1
$\{\mathbf{1}_1\}$	1	1	1	1	1	1	1
$\{\mathbf{1}_0, \mathbf{0}\}$	0	0	1	0	0	1	0
$\{\mathbf{1}_1, \mathbf{0}\}$	0	1	1	0	0	1	0
$\{\mathbf{1}_1, \mathbf{1}_0\}$	0	1	1	1	1	1	1
$\{\mathbf{1}_1, \mathbf{1}_0, \mathbf{0}\}$	0	1	1	0	0	1	0

表 2 连通顶点覆盖的兼容性矩阵。各行描述部分解 X_1 的标签状态，各列描述 X_2 的标签状态。1 表示标签状态对兼容，0 表示不兼容。

否。因此，我们可以将一个标签的状态描述为 $\{\mathbf{0}, \mathbf{1}_0, \mathbf{1}_1\}$ 的一个子集，并排除空子集，因为我们不考虑空标签。

接下来，我们研究这些标签状态在连接处的兼容性，但这里只作非正式说明。基本上，我们假设所考虑的连接是两个部分解 X_1，$X_2 \subseteq V$（它们位于连接的两侧，但 $r \in X_i$、$i \in [2]$ 位于连接处，r 除外）彼此连通的最后机会。因此，当 $X_1 \cup X_2$ 中每个与所考虑连接关联的顶点的状态为 $\mathbf{0}$ 或 $\mathbf{1}_1$，且连接中的每条边都由 $X_1 \cup X_2$ 覆盖时，我们认为部分解 X_1 和 X_2 是兼容的；参见图 1。如果 X_1 和 X_2 兼容，那么在所考虑的连接之外满足所有约束的情况下，$X_1 \cup X_2$ 可以成为一个全局解。由于前述状态捕捉了 X_i、$i \in [2]$ 与连接相应一侧之间的相互作用，因此我们得到了表 2 中的兼容性矩阵。

为确定转移顺序以及哪些状态应由路径构件实现，我们在重新排列行和列后，找出表3中的三角子矩阵。请注意，该三角子矩阵仅涉及至多由两个顶点状态组成的状态，因此，由两个顶点组成的标签应足以生成这些状态。事实上，在接下来的构造中，与连接相邻的标签是大小为二的独立集

X_1 对 X_2	$\{\mathbf{0}\}$	$\{\mathbf{1}_0, \mathbf{0}\}$	$\{\mathbf{1}_0\}$	$\{\mathbf{1}_1, \mathbf{0}\}$	$\{\mathbf{1}_1, \mathbf{1}_0\}$	$\{\mathbf{1}_1\}$
$\{\mathbf{1}_1\}$	1	1	1	1	1	1
$\{\mathbf{1}_1, \mathbf{1}_0\}$	0	1	1	1	1	1
$\{\mathbf{1}_1, \mathbf{0}\}$	0	0	1	0	1	1
$\{\mathbf{1}_0\}$	0	0	0	1	1	1
$\{\mathbf{1}_0, \mathbf{0}\}$	0	0	0	0	1	1
$\{\mathbf{0}\}$	0	0	0	0	0	1

表3 连通顶点覆盖的兼容性矩阵中的一个大型三角子矩阵。行和列已重新排序。

并且状态集将由以下有序的顶点状态对表示：$(\mathbf{0}, \mathbf{0})$、$(\mathbf{1}_0, \mathbf{0})$、$(\mathbf{1}_0, \mathbf{1}_0)$、$(\mathbf{1}_1, \mathbf{0})$、$(\mathbf{1}_1, \mathbf{1}_0)$、$(\mathbf{1}_1, \mathbf{1}_1)$。随后，沿对角线配对状态，即可针对每种情况确定路径构件分别应向左侧和右侧边界传递哪些状态。例如，对于对角线上的第三个位置，我们将$(\mathbf{1}_0, \mathbf{1}_0)$与$(\mathbf{1}_1, \mathbf{0})$配对，这意味着对于转移顺序中的第三个状态，路径构件应向左侧边界传递状态$(\mathbf{1}_0, \mathbf{1}_0)$，并向右侧边界传递状态$(\mathbf{1}_1, \mathbf{0})$。

状态的形式化定义。 我们定义三个原子状态 **原子组件** $= \{\mathbf{0}, \mathbf{1}_0, \mathbf{1}_1\}$，并定义两个谓词 $\texttt{sol}, \texttt{conn}$：**原子组件** $\to \{0, 1\}$，其定义为 $\texttt{sol}(\mathbf{a}) = [\mathbf{a} \in \{\mathbf{1}_0, \mathbf{1}_1\}]$ 以及$\texttt{conn}(\mathbf{a}) = [\mathbf{a} = \mathbf{1}_1]$。原子状态$\mathbf{0}$ 表示顶点不属于部分解；$\mathbf{1}_1$ 和 $\mathbf{1}_0$ 则表示顶点属于部分解，下标表示该顶点是否与根连通。基于这些原子状态，我们定义六种（构件）状态，每种状态由四个原子状态组成：

$$\begin{aligned}
\mathbf{s}^1 &= (\mathbf{0}\ , \mathbf{0}\ , \mathbf{1}_1, \mathbf{1}_1),\\
\mathbf{s}^2 &= (\mathbf{1}_0, \mathbf{0}\ , \mathbf{1}_1, \mathbf{1}_0),\\
\mathbf{s}^3 &= (\mathbf{1}_0, \mathbf{1}_0, \mathbf{1}_1, \mathbf{0}\ ),\\
\mathbf{s}^4 &= (\mathbf{1}_1, \mathbf{0}\ , \mathbf{1}_0, \mathbf{1}_0),\\
\mathbf{s}^5 &= (\mathbf{1}_1, \mathbf{1}_0, \mathbf{1}_0, \mathbf{0}\ ),\\
\mathbf{s}^6 &= (\mathbf{1}_1, \mathbf{1}_1, \mathbf{0}\ , \mathbf{0}\ ).
\end{aligned}$$

构件状态按照转移顺序编号。我们将这六种构件状态收集为集合**状态** $= \{\mathbf{s}^1, \ldots, \mathbf{s}^6\}$，并使用记号$\mathbf{s}_i^\ell \in$ **原子组件**，$i \in [4]$, $\ell \in [6]$来表示状态$\mathbf{s}^\ell$的第 i个原子分量。注意，$\mathbf{s}^\ell$ 可以通过将第一、第二个分量与第三、第四个分量互换，从$\mathbf{s}^{7-\ell}$ 得到，即对于所有 $\ell \in [6]$，$\mathbf{s}_1^\ell = \mathbf{s}_3^{7-\ell}$ 且$\mathbf{s}_2^\ell = \mathbf{s}_4^{7-\ell}$。

给定一个部分解 $Y \subseteq V(G)$，我们通过映射 **状态**$_Y$，将每个顶点在 Y 中的状态正式关联起来：$V(G) \setminus \{\hat{r}\} \to$ **原子组件**，其定义如下：

$$\mathbf{state}_Y(v) = \begin{cases} \mathbf{0} & \text{if } v \notin Y, \\ \mathbf{1}_0 & \text{if } v \in Y \text{ and } v \text{ is not root-connected in } Y \cup \{\hat{r}\}, \\ \mathbf{1}_1 & \text{if } v \in Y \text{ and } v \text{ is root-connected in } Y \cup \{\hat{r}\}. \end{cases}$$

形式化构造。我们接下来描述如何构造路径构件 P。我们创建 4 个连接顶点 $u_1, \ldots, u_4$、12 个辅助顶点 a_1,1, a_1,2, a_1,3, a_2,1, $\ldots$, a_4,3、8 个解指示顶点 b_1,0, b_1,1, b_2,0, b_2,1, $\ldots$, b_4,1、8 个连通性指示顶点 c_1,0, c_1,1, c_2,0, c_2,1, $\ldots$, c_4,1，以及 6 个团顶点 $v_1, \ldots, v_6$。对于每个 $i \in [4]$，我们添加边 $\{u_i, a_i,1\}$、$\{u_i, a_i,3\}$、$\{u_i, b_i,0\}$、$\{a_i,1, a_i,2\}$、$\{a_i,1, b_i,0\}$、$\{a_i,1, c_i,1\}$、$\{a_i,3, b_i,1\}$、$\{b_i,0, b_i,1\}$ 和 $\{c_i,0, c_i,1\}$。此外，对于所有 $i \in [4]$，我们使 a_i,3、所有解指示顶点、所有连通性指示

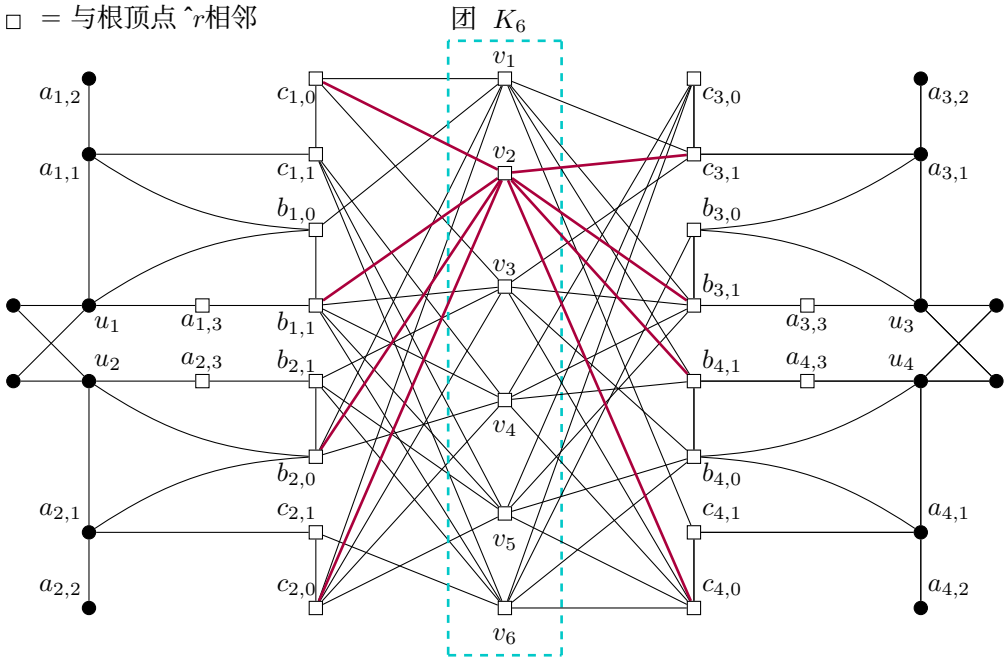

图 2 P 路径构件，其中连接顶点 u_1, u_2 和 u_3, u_4 已分别连接到其他顶点。图中以矩形表示的所有顶点都与根顶点r相邻。青色虚线矩形内的顶点构成一个团。为便于理解，我们以 v_2 为例，突出显示与其关联的边。

顶点以及所有团顶点均与根顶点 $\hat{r}$ 相邻。我们在团顶点 v_ℓ, $\ell \in [6]$ 之间添加所有可能的边，使它们构成一个大小为 6 的团。

最后，我们说明如何将指示顶点连接到团顶点。团顶点 v_ℓ 对应于在连接顶点 (u_1, u_2, u_3, u_4)上选择状态 s^ℓ 。 P 所需的行为是： $P + \hat{r}$ 的部分解 X 包含 b_i,1，当且仅当 X 包含 u_i; 对于连通性指示顶点， X 包含 c_i,1，当且仅当 X 包含 u_i ，且 u_i 在 X中与根连通。因此，对于所有 $i \in [4]$ 和 $\ell \in [6]$，我们添加边 $\{v_\ell, b_{i,\mathtt{sol}(\mathsf{s}_i^\ell)}\}$ 和 $\{v_\ell, c_{i,\mathtt{conn}(\mathsf{s}_i^\ell)}\}$。至此， P的构造完成，见图 2。

单路径构件的行为。对于接下来的引理，我们假设 G 是一个图，其中包含 $P + \hat{r}$ 作为诱导子图，并且只有连接顶点 u_i、 $i \in [4]$和团顶点 v_ℓ、 $\ell \in [6]$的邻居位于该 $P + \hat{r}$副本之外。此外，设 X 是 G 的一个连通顶点覆盖，并带有 $\hat{r} \in X$。我们研究此类连通顶点覆盖在 P 上的行为；为方便起见，我们将使用 $X \cap P$ 代替 $X \cap V(P)$。关于 P 如何连接到其余图的假设意味着，任何状态为 $_{X\cap P}(v)$ 的顶点 $v \in V(P)$ 都必须通过一条经由连接顶点 u_i、 $i \in [4]$ 离开 $P + \hat{r}$ 的路径，在 X 中与根连通。注意，任何经由团顶点 v_ℓ、 $\ell \in [6]$ 离开 $P + \hat{r}$ 的路径，都会立即在 $P + \hat{r}$ 中产生一条通往 $\hat{r}$ 的路径，即 $\{v_\ell : \ell \in [6]\} \subseteq N(\hat{r})$。

我们首先通过对子图进行顶点不相交打包，证明 $|X \cap P|$ 的下界。

▶ **引理 5.1.** 我们有 $|X \cap P| \geq 21 = 4 \cdot 4 + 5$ ，更具体地说，对于所有 $i \in [4]$ 和 $|X \cap \{v_1, \ldots, v_6\}| \geq 5$, $a_{i,1} \in X$,$|X \cap \{u_i, a_{i,3}, b_{i,1}, b_{i,0}\}| \geq 2$, $|X \cap \{c_{i,0}, c_{i,1}\}| \geq 1$ 。

证明。 对于所有的 $i \in [4]$，顶点 $a_{i,2}$ 在 G 中度为 1，其唯一邻居是 $a_{i,1}$，因此我们必须有 $a_{i,1} \in X$ ，要么连接 $a_{i,2}$ 与 $\hat{r}$ ，要么在 $a_{i,2} \notin X$ 时覆盖边 $\{a_{i,1}, a_{i,2}\}$ 。

对于所有 $i \in [4]$，顶点 $u_i, a_{i,3}, b_{i,1}, b_{i,0}$ 诱导出一个长度为 4 的环，且任意顶点覆盖都必须包含每个此类环中至少 2 个顶点。对于所有$i \in [4]$，边 $\{c_{i,0},c_{i,1}\}$ 必须被覆盖。最后，由顶点 $v_1, \ldots, v_6$ 构成的团大小为 6，任意顶点覆盖都必须包含此类团中至少 5 个顶点。由于我们只考虑了两两不相交的顶点集，这些下界直接相加，得到 $|X \cap P| \geq 4(1+2+1) + 5 = 21$。◀

利用引理 5.1，我们可以精确分析在 P 上达到下界 21 的解，并证明这类解具有所需的状态行为。对于任意$Y \subseteq V(P)$，我们定义 4 元组**状态**$(Y) = ($**状态**$_Y(u_1)$，**状态**$_Y(u_2)$，**状态**$_Y(u_3)$，**状态**$_Y(u_4))$，以下引理表明，传递到边界的状态会按预期取决于中心团的状态。

▶ 引理 *5.2*。如果 $|X \cap P| \leq 21$，那么 $|X \cap P| = 21$ 和$a_{i,1} \in X$, $X \cap \{u_i, a_i,3, b_i,1, b_i,0\} \in \{\{u_i, b_i,1\}, \{a_i,3, b_i,0\}\}$，$|X \cap \{c_i,0, c_i,1\}| = 1$ 对所有 $i \in [4]$ 以及 $|X \cap \{v_1, \ldots, v_6\}| = 5$均成立。此外，对于满足 $v_\ell \notin X$的唯一整数 $\ell \in [6]$，我们有 $state(X \cap P) = s^\ell$。

证明。由于 $|X \cap P| \leq 21$，引理 5.1 中的所有不等式都必须取等号。由于$|X \cap \{u_i,a_i,3,b_i,1,b_i,0\}| = 2$ 和 $\{u_i, a_i,3, b_i,1, b_i,0\}$构成一个长度为 4 的环，$X$必须包含该环的一对对跖顶点，因此要么是顶点对 $\{u_i,b_i,1\}$，要么是顶点对 $\{a_i,3,b_i,0\}$。接下来只需证明关于连接顶点状态的最后一个性质。

任取 $i \in [4]$，我们将证明状态$_{X \cap P}(u_i) = \mathrm{s}_i^\ell$。由于 X是一个顶点覆盖且$v_\ell \notin X$，我们必有 $N(v_\ell) \subseteq X$。根据构造，这意味着 X 尤其必须包含顶点 $b_{i,\mathtt{sol}(\mathrm{s}_i^\ell)}$ 和 $c_{i,\mathtt{conn}(\mathrm{s}_i^\ell)}$；并且根据前面的等式，$X$ 不能包含这些指示顶点对中各自的另一个顶点。由对跖顶点对的选取可知，$u_i \in X$ 当且仅当 $b_{i,1} \in X$ 当且仅当 $\mathtt{sol}(\mathrm{s}_i^\ell) = 1$ 当且仅当 $\mathrm{s}_i^\ell \in \{\mathbf{1}_0, \mathbf{1}_1\}$。若$u_i \in X$，则 $a_{i,3} \notin X$ 且 $b_{i,0} \notin X$，因此 u_i 在 $X \cap (P + \hat{} r)$ 中与根连通，当且仅当$c_{i,1} \in X$，也就是说，沿着路径 $u_i, a_i,1, c_i,1, \hat{} r$。此外，$c_{i,1} \in X \Leftrightarrow \mathtt{conn}(\mathrm{s}_i^\ell) = 1 \Leftrightarrow \mathrm{s}_i^\ell = \mathbf{1}_1$。这表明状态$_{X \cap P}(u_i) = \mathrm{s}_i^\ell$；由于这一论证适用于所有 $i \in [4]$，我们得到状态$(X \cap Y) = \mathrm{s}^\ell$。◀

接下来，我们将证明，对于每个状态 $\mathbf{s}^\ell \in$**状态**，实际上都存在一个达到该状态的部分解。此外，对于这些部分解，只需检查连接顶点处是否与根连通即可。

▶ **引理 5.3**。对于每个 $\ell \in [6]$，存在一个 P 的顶点覆盖 X_P^ℓ，使得 $|X_P^\ell| = 21, X_P^\ell \cap \{v_1, \ldots, v_6\} = \{v_1, \ldots, v_6\} \setminus \{v_\ell\}$，且 **状态**$(X_P^\ell) = \mathbf{s}^\ell$。如果 X 是 G 的一个顶点覆盖，且有 $\hat{} r \in X$ 和 $X \cap P = X_P^\ell$，并且对于每个 $i \in [4]$，$u_i \notin X$ 或 u_i 在 X 中与根连通，那么 X_P^ℓ 的每个顶点都在 X 中与根连通。

证明。我们定义

$$\begin{aligned} X_P^\ell = {} & (\{v_1, \ldots, v_6\} \setminus \{v_\ell\}) \cup \{a_{i,1} : i \in [4]\} \\ & \cup \{u_i : i \in [4] \text{ and } \mathtt{sol}(\mathrm{s}_i^\ell) = 1\} \cup \{a_{i,3} : i \in [4] \text{ and } \mathtt{sol}(\mathrm{s}_i^\ell) = 0\} \\ & \cup \{b_{i,\mathtt{sol}(\mathrm{s}_i^\ell)} : i \in [4]\} \cup \{c_{i,\mathtt{conn}(\mathrm{s}_i^\ell)} : i \in [4]\} \end{aligned}$$

并断言 X_P^ℓ 是一个满足所需性质的顶点覆盖。显然，我们有$|X_P^\ell| = 21$和 $v_\ell \notin X_P^\ell$。接下来证明 X_P^ℓ是 P的一个顶点覆盖。对于每个$i \in [4]$，与 $a_i,1$ 相 incident 的四条边均已覆盖，并且 X_P^ℓ至少包含边 $\{c_i,0,c_i,1\}$ 的一个顶点。对于每个 $i \in [4]$，由 $\{u_i, a_i,3, b_i,0, b_i,1\}$ 诱导的 C_4中的所有边都由 X_P^ℓ覆盖，因为 X_P^ℓ会根据 $\mathtt{sol}(\mathrm{s}_i^\ell)$ 选取两对对跖点对中的一对。由 $v_1, \ldots, v_6$ 诱导的团完全被 X_P^ℓ 覆盖，因为它选取了六个顶点中的五个。这些

条边。 v_ℓ 与指示顶点之间的边均已覆盖，因为根据 P的构造，我们有 $N(v_\ell) = (\{v_1, \dots, v_6\} \setminus \{v_\ell\}) \cup \{b_{i,\mathrm{sol}(\mathbf{s}_i^\ell)} \colon i \in [4]\} \cup \{c_{i,\mathrm{conn}(\mathbf{s}_i^\ell)} \colon i \in [4]\} \subseteq X_P^\ell$。这说明 X_P^ℓ 是 P 的一个顶点覆盖。与引理 5.2 的证明非常相似，我们可以看到**状态**$(X_P^\ell) = \mathbf{s}^\ell$。

接下来需要证明关于连通性的性质。根据假设，只有连接顶点 $u_1, \dots, u_4$ 和团顶点 $v_1, \dots, v_6$ 能与 $P + \hat{r}$ 外部的顶点相邻。然而，由于团顶点是根 $\hat{r}$ 的邻居，它们与 P 外部的其他连接无法提供任何新的根连通性。若对于某个 $i \in [4]$，我们有状态$^{X_P^\ell}(u_i) = \mathbf{1}_0$，那么 u_i 在 X 中通过某条离开 $P + \hat{r}$ 的路径与根连通。注意， $X_P^\ell \setminus \{u_i, a_i, \mathbf{1} \colon i \in [4]\}$ 中的所有顶点都直接与根 $\hat{r}$ 相邻，因此只需处理 a_i,**1** 的根连通性。若状态$^{X_P^\ell}(u_i) = \mathbf{0}$，则 a_i,**1** 通过路径 a_i,**1**,b_i,**0**, $\hat{r}$ 在 X 中与根连通。若状态$^{X_P^\ell}(u_i) = \mathbf{1}_0$，则我们可以将离开 P 并连接 u_i 与 $\hat{r}$ 的路径延伸至 a_i,**1**。最后，若状态$^{X_P^\ell}(u_i) = \mathbf{1}_1$，则 a_i,**1**,c_i,**1**, $\hat{r}$ 这条路径存在于$X_P^\ell + \hat{r}$ 中。至此证明完成。◀

状态转移。 在下界构造中，我们将通过反复连接路径构件 P 来创建长路径。为了研究状态如何在两个连续的路径构件之间变化，假设我们有 P 的两个副本 P^1 和 P^2，其中 P^1 中的顶点 u_3 和 u_4 分别与 P^2 中的顶点 u_1 和 u_2 相连。我们用上标 **1** 标记 P^1 的顶点，用上标 2 标记 P^2 的顶点，例如， u_3^1 指的是 P^1 的顶点 u_3。同样，假设 P^1 和 P^2 作为诱导子图嵌入更大的图 G 中，该图有一个根顶点 $\hat{r}$，并且只有顶点 u_1^1、u_2^1、u_3^2、u_4^2 以及团顶点 v_ℓ^1、v_ℓ^2、$\ell \in [6]$ 在 $\hat{r}$ 的 $P^1 + P^2 +$ 之外有邻居。此外， X 表示一个包含 $\hat{r} \in X$ 的连通顶点覆盖。

利用前面的引理，我们现在证明，状态只能按照转移顺序从一个路径构件转移到下一个路径构件，并且状态也可以保持稳定。

▶ **引理 5.4.** 假设 $|X \cap P^1| \leq 21$ 和 $|X \cap P^2| \leq 21$，则 **状态**$(X \cap P^1) = \mathbf{s}^{\ell_1}$且 **状态**$(X \cap P^2) = \mathbf{s}^{\ell_2}$，其中 $\ell_1 \leq \ell_2$。

此外，对于每个 $\ell \in [6]$，集合 $X^\ell = X_{P^1}^\ell \cup X_{P^2}^\ell$是 $P^1 + P^2$的一个顶点覆盖，其中**状态**$_{X^\ell}(\{u_3^1, u_4^1, u_1^2, u_2^2\}) \subseteq \{\mathbf{0}, \mathbf{1}_1\}$。

证明。 根据引理 5.2，我们有 **状态**$(X \cap P^1) = \mathbf{s}^{\ell_1}$ 且 **状态**$(X \cap P^2) = \mathbf{s}^{\ell_2}$ 对于某个$\ell_1 \in [6]$ 且 $\ell_2 \in [6]$。接下来需要证明 $\ell_1 \leq \ell_2$。

定义 $U^1 = \{u_3^1, u_4^1\}$、 $U^2 = \{u_1^2, u_2^2\}$ 和 $U = U^1 \cup U^2$。根据假设，只有 P^1 和 P^2 的团顶点以及顶点 $u_1^1, u_2^1, u_3^2, u_4^2$ 可以拥有 $P^1 + P^2 + \hat{r}$ 之外的邻居，因此 $\{v_\ell^i \colon i \in [2], \ell \in [6]\} \subseteq N(\hat{r})$ 将整个图 G 中的 U 与根 $\hat{r}$ 分隔开来。因此，只需考察图 $P^1 + P^2 + \hat{r}$，就能判断 $X \cap U$ 中的顶点在 X 中是否与根连通。

定义原子状态对 $^-\mathrm{s}^1 = (\mathrm{s}_3^{\ell_1}, \mathrm{s}_4^{\ell_1}) = (\text{状态}_{X \cap P^1}(u_3^1), \text{状态}_{X \cap P^1}(u_4^1))$ 和$^-\mathrm{s}^2 = (\mathrm{s}_1^{\ell_2}, \mathrm{s}_2^{\ell_2}) = (\text{状态}_{X \cap P^2}(u_1^2), \text{状态}_{X \cap P^2}(u_2^2))$。我们声称，只要满足 $\ell_1 > \ell_2$， X就不会覆盖 $G[U]$ 中的某条边，或者 $X \cap U$ 中的某个顶点在 X 中不与根连通，另见图 3。 $G[U]$ 中的某条边未被 X 覆盖，当且仅当 $^-\mathrm{s}^1$ 和 $^-\mathrm{s}^2$ 都包含一个 0。因此，$(\ell_1, \ell_2) \notin \{(3, 1), (3, 2), (5, 1), (5, 2), (5, 4), (6, 1), (6, 2), (6, 4)\}$。$X \cap U$ 中的某个顶点在 X 中不与根连通，当且仅当 $^-\mathrm{s}^1$ 和 $^-\mathrm{s}^2$ 都完全不包含 $\mathbf{1}_1$s，或者其中一个由两个 0 组成，而另一个包含一个 $\mathbf{1}_0$。此外，这也表明 $(\ell_1, \ell_2) \notin \{(2, 1), (4, 1), (4, 2), (4, 3), (5, 3), (6, 3), (6, 5)\}$，并由此完成第一部分的证明。

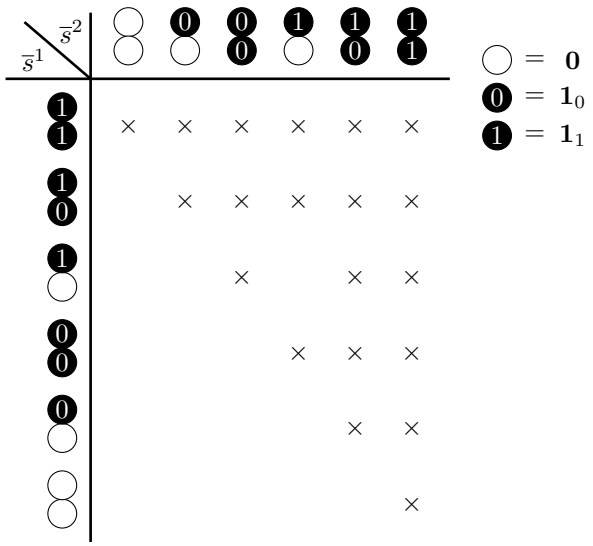

图 3 该矩阵展示了两个连续路径构件之间可能的状态转移。行的标签为 $\bar{}\mathbf{s}^1 = (\mathbf{s}_3^{\ell_1}, \mathbf{s}_4^{\ell_1})$、$\ell_1 \in [6]$；列的标签为 $\bar{}\mathbf{s}^2 = (\mathbf{s}_1^{\ell_2}, \mathbf{s}_2^{\ell_2})$、$\ell_2 \in [6]$。× 标记了连通顶点覆盖中的可能状态转移。

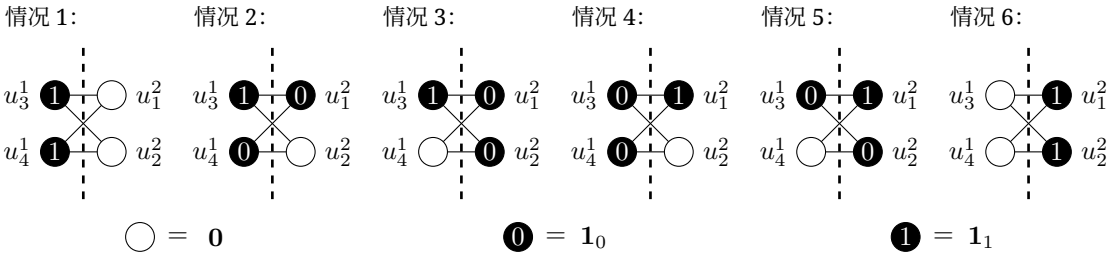

图 4 用于展示**状态**$_{X^\ell}(\{u_3^1, u_4^1, u_1^2, u_2^2\}) \subseteq \{\mathbf{0}, \mathbf{1}_1\}$的情况区分。每种情况中，左侧描绘的是 $\bar{}\mathbf{s}^1$，右侧描绘的是 $\bar{}\mathbf{s}^2$。

对于第二部分，注意到**状态**$(X_{P^1}^\ell)$ = **状态**$(X_{P^2}^\ell)$ = $\mathbf{s}^\ell$由引理 5.3 可知，并采用上一段中的相同方法，我们看到对于 $\ell = \ell_1 = \ell_2$， $G[U]$中的所有边都被覆盖，且 $X^\ell \cap U$ 中的所有顶点在 $X^\ell \cup \{\hat{r}\}$中都与根连通。不同情形见图 4。◀

如果 $\ell_1 < \ell_2$，我们称发生作弊 。通过创建任意长的路径构件P，引理 5.4 告诉我们，此类路径上至多发生 |**个状态**| $- 1 = 5 = \mathcal{O}(1)$ 次作弊，因此我们可以如前所述找到一个无作弊区域 。

5.1.2 完整构造

准备。 假设连通顶点覆盖可以在时间 $\mathcal{O}^*((6-\varepsilon)^{\text{lin-cw}(G)})$ 内求解，其中 $\varepsilon > 0$ 为某个值。给定一个包含 n 个变量和 m 个子句的可满足性实例 σ，我们构造一个等价的连通顶点覆盖实例，其线性团宽约为 $n\log_6(2)$，因此，如果存在这样的连通顶点覆盖算法，就意味着强指数时间假设不成立。

我们选取一个仅取决于 ε 的整数 β； β 的具体取值将在后文讨论。 σ 的变量被划分为若干组，每组大小至多为 β，因此得到

$t = \lceil n/\beta \rceil$ 组。此外，我们选取满足 $6^p \geq 2^\beta$ 的最小整数 p，即 $p = \lceil \log_6(2^\beta) \rceil$。现在我们开始构造连通顶点覆盖实例 $(G = G(\sigma, \beta)\ b)$。

我们创建根顶点 $\hat{r}$，并连接一个叶节点 $\hat{r}'$，迫使 $\hat{r}$ 属于任意连通顶点覆盖。对于每个组 $i \in [t]$，我们创建 p 个长路径型构件 $P^{i,j}, j \in [p]$，其中每个 $P^{i,j}$ 由 $m(5tp+1)$ 个路径构件 $P^{i,j,\ell}, \ell \in [m(5tp+1)]$, 副本组成，相邻副本通过连接相连。更准确地说，某些 $P^{i,j,\ell}$ 中的顶点沿用 P 中的名称，并带有 $P^{i,j,\ell}$ 的上标；对于每个 $i \in [t], j \in [p], \ell \in [m(5tp+1) - 1]$，顶点 $\{u^{i,j\,\ell}_1, u^{i,j\,\ell}_2, u^{i,j\,\ell}_3, u^{i,j\,\ell}_4\}$ 与顶点 $\{u^{i,j\,\ell+1}_1, u^{i,j\,\ell+1}_2\}$ 相连。每条路径 $P^{i,j}$ 的端点，即顶点 $u^{i,j\,1}_1, u^{i,j\,1}_2, u^{i,j\,m(5tp+1)}_3, u^{i,j\,m(5tp+1)}_4$，都与根 $\hat{r}$ 相邻。

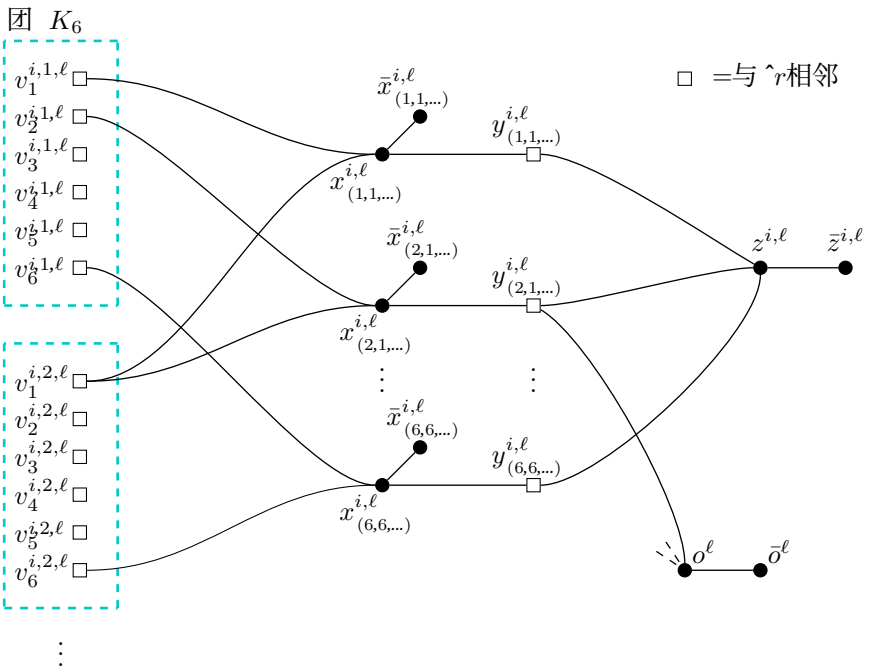

图 5 连通顶点覆盖的解码构件和子句构件。

对于每个组 $i \in [t]$ 和列 $\ell \in [m(5tp+1)]$，我们按照 Cygan 等人 [17] 针对以路径宽度为参数的连通顶点覆盖问题所采用的相同方式，构造一个解码构件 $D^{i,\ell}$。每个变量组 i 至多有 2^β 种可能的真值赋值；根据 p 的选择，我们有 $6^p \geq 2^\beta$，因此可以找到一个单射 $\kappa: \{0,1\}^\beta \to [6]^p$，将每个真值赋值 $\tau \in \{0,1\}^\beta$ 映射到一个序列 $\kappa(\tau) \in [6]^p$。对于每个序列 $\mathbf{h} = (h_1, \ldots, h_p) \in [6]^p$，我们构造顶点 $x^{i,\ell}\mathbf{h}$，$\bar{x}^{i,\ell}\mathbf{h}$，$y^{i,\ell}\mathbf{h}$，以及边 $\{x^{i,\ell}\mathbf{h}\ \bar{x}^{i,\ell}\mathbf{h}\}, \{x^{i,\ell}\mathbf{h}\ y^{i,\ell}\mathbf{h}\}, \{y^{i,\ell}\mathbf{h}\ \hat{r}\}$。此外，对于所有 $\mathbf{h} = (h_1, \ldots, h_p) \in [6]^p$ 和 $j \in [p]$，我们添加边 $\{x^{i,\ell}\mathbf{h}\ v^{i,j\,\ell}_{h_j}\}$。最后，我们构造两个相邻顶点 $z^{i,\ell}$ 和 $\bar{z}^{i,\ell}$，并添加边 $\{z^{i,\ell}, y^{i,\ell}\mathbf{h}\}$，其中 $\mathbf{h} \in [6]^p$ 取遍所有可能值。每个解码构件 $D^{i,\ell}$ 与相邻的路径构件 $P^{i,j,\ell}, j \in [p]$ 共同组成块 $B^{i,\ell}$。不同块之间连接关系的高层示意图见图 6。

最后，我们构造子句构件。我们将 σ 的子句编号为 $C_0, \ldots, C_{m-1}$。对于每一列 $\ell \in [m(5tp+1)]$，我们创建一对相邻顶点 o^ℓ 和 $\bar{o}^\ell$。令 $\ell' \in [0, m-1]$ 为 $(\ell - 1)$ 模 m 的余数；对于所有 $i \in [t]$，$\mathbf{h} \in \kappa(\{0,1\}^\beta)$，使得 $\kappa^{-1}(\mathbf{h})$ 是一个为变量组 i 赋值且满足子句 $C_{\ell'}$ 的真值赋值，我们添加边 $\{o^\ell, y^{i,\ell}\mathbf{h}\}$。解码和子句构件的示意图见图 5。

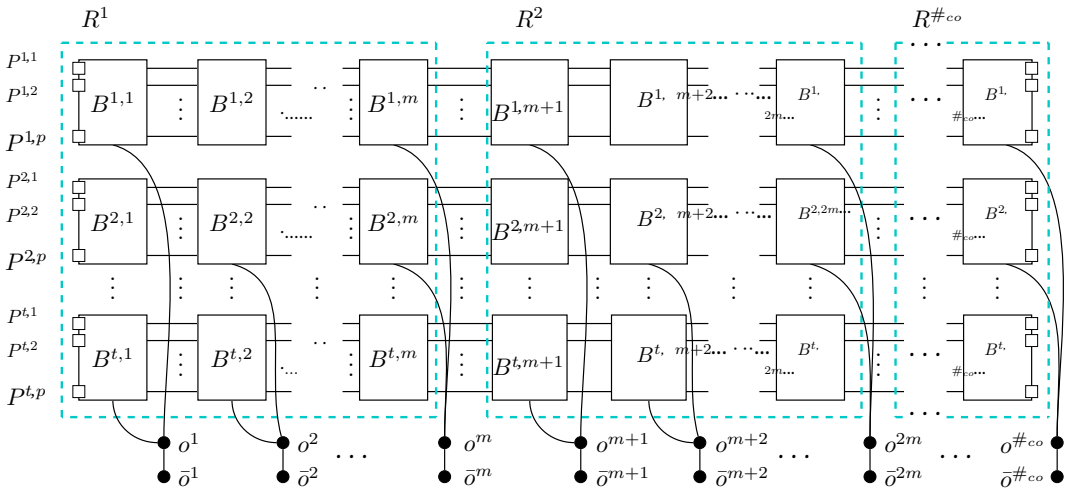

图 6 连通顶点覆盖的高层构造。两个块 $B^{i,\ell}$ 和 $B^{i,\ell+1}$ 之间的每条边都表示一次连接。区域 R^{γ} 对引理 5.6 的证明至关重要。

▶ **引理 5.5**。 如果 σ 可满足，那么存在 $G = G(\sigma, \beta)$ 的一个大小为 $|X| \leq (21tp + (6^p + 2)t + 1)m(5tp + 1) + 1 = b$ 的连通顶点覆盖 X。

证明。 令 τ 为 σ 的一个满足真值赋值，并令 τ^i 表示 τ 在每个 i- $i \in [t]$ 变量组上的限制，并令 $\kappa(\tau^i) = \mathbf{h}^i = (h_1^i, \ldots\ldots, h_p^i) \in [6]^p$ 为相应的序列。连通顶点覆盖为

$$X = \{\hat{r}\} \cup \bigcup_{\ell \in [m(5tp+1)]} \left(\{o^\ell\} \cup \bigcup_{i \in [t]} \left(\{y^{i,\ell}_{\mathbf{h}^i}, z^{i,\ell}\} \cup \bigcup_{\mathbf{h} \in [6]^p} \{x^{i,\ell}_{\mathbf{h}}\} \cup \bigcup_{j \in [p]} X^{h^i_j}_{P^{i,j,\ell}} \right) \right),$$

其中 $X^{h^i_j}_{P^{i,j,\ell}}$ 指的是引理 5.3 给出的集合。

显然，$|X| = \bar{b}$，因此还需证明 X 是一个连通顶点覆盖。根据引理 5.3 和引理 5.4 的第二部分，由路径构件诱导的所有边都被 X 覆盖，并且路径构件上属于 X 的所有顶点都与根连通，除了端点处的顶点可能不是如此，即 $\bigcup_{i \in [t]} \bigcup_{j \in [p]} \{u^i_{j,1}, u^i_{j,2}, u^i_{j,3}, u^i_{j,4}, \ldots, u^i_{j,m(5tp+1)}, u^i_{j,m(5tp+1)}\}$，但根据 G 的构造，这些顶点都包含在 $\hat{r}$ 的邻域中。

固定 $i \in [t]$, $\ell \in [m(5tp+1)]$, ，并考虑对应的解码构件。由于 $z^{i,\ell} \in X$ 以及 $x^{i,\ell}_{\mathbf{h}} \in X$ 对所有 $\mathbf{h} \in [6]^p$ 都成立，因此解码构件诱导的所有边，以及解码构件与路径构件之间的所有边，均由 X 覆盖。此外，由于 $o^\ell \in X$，子句构件内部的所有边，以及子句构件与解码构件之间的所有边，均由 X 覆盖。因此，X 必须是 G 的顶点覆盖。

剩下要证明的是，解码构件和子句构件中属于 X 的顶点也与根连通。同样，固定 $i \in [t]$, $\ell \in [m(5tp+1)]$, ，并令 $\mathbf{h} = (h_1, \ldots, h_p) \in [6]^p \setminus \{\mathbf{h}^i\}$。由于 $\mathbf{h} \neq \mathbf{h}^i$，存在某个 $j \in [p]$，使得 $h_j \neq h^i_j$，因此根据引理 5.3，$v^{i,j,\ell}_{h_j} \in X$，该引理将 $x^{i,\ell}_{\mathbf{h}}$ 连至根 $\hat{r}$。顶点 $x^{i,\ell}_{\mathbf{h}^i}$ 和 $z^{i,\ell}$ 通过 $y^{i,\ell}_{\mathbf{h}_i} \in X$ 与根连通。

最后，我们证明对于所有 $\ell \in [m(5tp+1)]$，o^ℓ 都与根连通。由于 τ 是 σ 的一个满足真值赋值，因此存在某个变量组 $i \in [t]$，使得 τ^i 已经满足子句 $C_{\ell'}$，其中 ℓ' 是 $(\ell - 1)$ 模 m 的余数。根据 G 的构造

以及 X，顶点 $y^{i,\ell}_{\mathbf{h}_i} \in X$ 与 o^ℓ相邻，因为 $\kappa(\tau^i) = \mathbf{h}^i$，并将 o^ℓ 与根 $\hat{r}$连通。这表明 X 的所有顶点都与根连通，因此 $G[X]$ 必定连通。◀

▶ **引理 5.6**。如果存在一个大小为 $|X| \leq (21tp + (6^p + 2)t + 1)m(5tp + 1) + 1 = b$ 的图 $G = G(\sigma, \beta)$ 的连通顶点覆盖 X，那么 σ 可满足。

证明。我们首先论证 X 必须满足 $|X| = b$。首先，必须有 $\hat{r} \in X$，因为 $\hat{r}$ 有一个度为 1 的邻居。根据引理 5.1，对于所有 $i \in [t]$, $j \in [p]$, $\ell \in [m(5tp + 1)]$，我们有 $|X \cap P^{i,j,\ell}| \geq 21$。在每个解码构件中，也就是说，对每个 $i \in [t]$ 和 $\ell \in [m(5tp + 1)]$，各有一个，集合 $\{z^{i,\ell}\} \cup \bigcup_{\mathbf{h} \in [6]^p} x^{i,\ell}_{\mathbf{h}}$ 必须包含在 X 中，因为该集合中的每个顶点都有一个度为 1 的邻居。此外，为了将 $z^{i,\ell}$ 与 $\hat{r}$ 连通，顶点 $y^{i,\ell}_{\mathbf{h}}, \mathbf{h} \in [6]^p$ 中至少有一个必须包含在 X 中。因此，X 每个解码构件至少必须包含 $6^p + 2$ 个顶点。最后，对于所有 $\ell \in [m(5tp + 1)]$，有 $o^\ell \in X$，因为 o^ℓ 有一个度为 1 的邻居。由于我们只考虑了互不相交的顶点集，这表明 $|X| = b$ 以及所有先前的不等式都必须取等。

根据引理 5.2，我们知道 X 在每个 $P^{i,j,\ell}$ 上都处于六种可能状态之一。固定某个 $P^{i,j} = \bigcup_{\ell \in [m(5tp+1)]} P^{i,j,\ell}$，并注意到根据引理 5.4，沿着 $P^{i,j}$ 状态最多改变五次。这种状态改变称为作弊。令 $\gamma \in [0, 5tp]$，并定义第 γ 个区域 $R^\gamma = \bigcup_{i \in [t]} \bigcup_{j \in [p]} \bigcup_{\ell=\gamma m+1}^{(\gamma+1)m} P^{i,j,\ell}$。由于有 $5tp + 1$ 个区域，至少存在一个区域 R^γ，其中不会发生作弊。证明余下部分中，我们将考察这个区域，并从该区域读出一个满足真值赋值。

对于 $i \in [t]$，定义 $\mathbf{h}^i = (h^i_1, \ldots, h^i_p) \in [6]^p$，使得 $v^{i,j,\gamma m+1}_{h^i_j} \notin X$ 对所有 $j \in [p]$ 都成立；由引理 5.2 可知这一定义良好。由于 R^γ 不包含任何作弊，$\mathbf{h}^i$ 的定义与我们考察哪一列 $\ell \in [\gamma m + 1, (\gamma + 1)m]$ 无关。对于每个 $i \in [t]$ 和 $\ell \in [\gamma m + 1, (\gamma + 1)m]$，我们断言 $y^{i,\ell}_{\mathbf{h}} \in X$ 当且仅当 $\mathbf{h} = \mathbf{h}^i$。我们已经证明，对于每个 i 和 ℓ，恰好存在一个 $\mathbf{h}$，使得 $y^{i,\ell}_{\mathbf{h}} \in X$。考虑顶点 $x^{i,\ell}_{\mathbf{h}_i} \in X$，它在 G 中的邻居为 $v^{i,1,\ell}_{h^i_1}, v^{i,2,\ell}_{h^i_2}, \ldots, v^{i,p,\ell}_{h^i_p}, \bar{x}^{i,\ell}_{\mathbf{h}^i}$，以及 $y^{i,\ell}_{\mathbf{h}^i}$。根据 $\mathbf{h}^i$ 的构造以及预算的紧密分配，我们有 $(N_G(x^{i,\ell}_{\mathbf{h}^i}) \setminus \{y^{i,\ell}_{\mathbf{h}^i}\}) \cap X = \emptyset$。因此，$X$ 必须包含 $y^{i,\ell}_{\mathbf{h}^i}$，才能将 $x^{i,\ell}_{\mathbf{h}^i}$ 连到根 $\hat{r}$。这证明了该断言。

对于 $i \in [t]$，我们通过以下方式定义变量组 i 的真值赋值 τ^i：当 $\mathbf{h}^i \notin \kappa(\{0,1\}^\beta)$ 时，任取一个真值赋值；否则，将 $\tau^i = \kappa^{-1}$设置为 $(\mathbf{h}^i)$。通过设置 $\tau = \bigcup_{i \in [t]} \tau^i$，我们得到所有变量的一个真值赋值，并声称 τ 满足 σ。考虑某个子句 $C_{\ell'}, \ell' \in [0, m-1]$，并令 $\ell = \gamma m + \ell' + 1$。我们已经论证了 $o^\ell \in X$；为了将 o^ℓ 连接到根 $\hat{r}$，必须存在某个 $y^{i,\ell}_{\mathbf{h}} \in N_G(o^\ell) \cap X$。根据前述断言，$\mathbf{h} = \mathbf{h}^i$，因此 τ^i，并且由于 G的构造，τ也满足子句 $C_{\ell'}$。由于 $C_{\ell'}$ 的选择是任意的，τ 必然是 σ的满足赋值。◀

▶ **引理 5.7.** 构造的图 $G = G(\sigma, \beta)$ 的线性团宽$(G) \leq tp + 3 \cdot 6^p + O(1)$，且可以在多项式时间内构造一个宽度为此值的线性团表达式。

证明。我们将描述如何为 G 构造一个宽度为 $tp + 3 \cdot 6^p + \mathcal{O}(1)$ 的线性团表达式。该团表达式将为每条长路径 $P^{i,j}, i \in [t], j \in [p]$使用一个路径标签（$i, j$），为解码构件的每个顶点使用临时解码标签，即 $3 \cdot 6^p + 2$ 个；为路径构件的每个顶点使用临时构件标签，即 38 个；为子句构件使用两个临时子句标签；为根顶点使用一个标签，以及一个垃圾标签。

团表达式的构造见算法 1，其核心思路是按列逐列处理，并在每一列中按组逐组处理。通过重复使用临时标签，我们将标签总数控制在较小范围内。同时使用的标签数量在第 7 行达到最大，为 $tp + (3 \cdot 6^p + 2) + 38 + 2 + 1 + 1$。证明至此完成。

◀

算法 1 为 G 构造线性团表达式。

```
引入带根标签的 r̂ 和带垃圾标签的 r̂′，并添加边 {r̂, r̂′};
对 ℓ ∈ [m（5tp + 1）] 执行
    引入 o^ℓ 和 ō^ℓ 以及边 {o^ℓ, ō^ℓ}，并为其添加子句标签;
    对 i ∈ [t] 执行
        使用解码标签构建 D^{i,ℓ}，并向 o^ℓ 添加边;
        对 j ∈ [p] 执行
            使用构件标签构建 P^{i,j,ℓ}，并向 D^{i,j} 添加边;
            若 ℓ > 1，则将 u^{i,j,ℓ,1} 和 u^{i,j,ℓ,2} 连接至路径标签（i, j），否则连接至根标签;
            将路径标签（i, j）重新标记为垃圾标签;
            将 u^{i,j,ℓ,3} 和 u^{i,j,ℓ,4} 重新标记为路径标签（i, j）;
            将所有其他构件标签重新标记为垃圾标签;
        将所有解码标签重新标记为垃圾标签;
    将所有子句标签重新标记为垃圾标签;
```

▶定理 5.8。 </style}除非 CNF-SETH 不成立，否则不存在算法能在时间 O(($(6-\varepsilon)^k$) 内，针对给定的线性 k-表达式求解连通顶点覆盖问题，其中 $\varepsilon > 0$。

证明。 假设存在一个算法$\mathcal{A}$，能够在$\mathcal{O}^*$(($6-\varepsilon$)^k)时间内求解连通顶点覆盖，其中 $\varepsilon > 0$ 对应一个给定的线性 k-表达式。给定 β，我们定义 $\delta_1 < 1$使得($6-\varepsilon$)$^{\log_6(2)} = 2^{\delta_1}$，并定义 δ_2 使得($6-\varepsilon$)$^{1/\beta} = 2^{\delta_2}$。通过将 β 取足够大，我们可以确保 $\delta = \delta_1 + \delta_2 < 1$。我们将展示如何使用$\mathcal{A}$ 在$\mathcal{O}^*$($2^{\delta n}$)时间内求解可满足性，其中 n 为变量数，从而与 CNF-SETH 矛盾。

给定一个可满足性实例 σ，我们在多项式时间内构造 $G = G(\sigma, \beta)$以及引理 5.7 中的线性团表达式。注意， β =$\mathcal{O}$(1)，因此$p = \mathcal{O}$(1)；回顾一下， $p = \lceil \log_6(2^\beta) \rceil$。然后，我们在 G 上运行 $\mathcal{A}$ 并返回其答案。根据引理 5.5 和引理 5.6，该答案是正确的。由引理 5.7，运行时间为

$$
\begin{aligned}
& \mathcal{O}^*\left((6-\varepsilon)^{tp+3\cdot 6^p+\mathcal{O}(1)}\right) \leq \mathcal{O}^*\left((6-\varepsilon)^{tp}\right) && \leq \mathcal{O}^*\left((6-\varepsilon)^{\lceil \frac{n}{\beta} \rceil p}\right) \\
\leq\ & \mathcal{O}^*\left((6-\varepsilon)^{\frac{n}{\beta}p}\right) && \leq \mathcal{O}^*\left((6-\varepsilon)^{\frac{n}{\beta}\lceil \log_6(2^\beta) \rceil}\right) && \leq \mathcal{O}^*\left((6-\varepsilon)^{\frac{n}{\beta}\log_6(2^\beta)}(6-\varepsilon)^{\frac{n}{\beta}}\right) \\
\leq\ & \mathcal{O}^*\left(2^{\delta_1 \beta \frac{n}{\beta}} 2^{\delta_2 n}\right) && \leq \mathcal{O}^*\left(2^{(\delta_1+\delta_2)n}\right) && \leq \mathcal{O}^*\left(2^{\delta n}\right),
\end{aligned}
$$

从而完成证明。 ◀

5.2 连通支配集

本小节旨在证明，除非 CNF-SETH 不成立，否则连通支配集问题（单位代价）无法在时间 $\mathcal{O}^*((5-\varepsilon)^{\text{lin-cw}(G)})$内求解，其中 $\varepsilon > 0$。我们将简要阐述路径构件设计背后的直觉，其思路大体沿袭连通顶点覆盖问题的做法。随后，我们将给出路径构件的构造并对其进行分析，接着介绍完整构造及正确性证明。解码构件同样直接改编自 Cygan 等人给出的、以路径宽度为参数的连通顶点覆盖问题下界 [17]。

根。 我们创建一个特殊顶点 $\hat{r}$，称其为根，并通过向 $\hat{r}$ 连接一个度为 1 的顶点，确保每个连通支配集都必须包含 $\hat{r}$。

	$\{\mathbf{0}_1, \mathbf{0}_0\}$	$\{\mathbf{0}_1\}$	$\{\mathbf{1}_0, \mathbf{0}_0\}$	$\{\mathbf{1}_1, \mathbf{0}_0\}$	$\{\mathbf{1}_1\}$
$\{\mathbf{1}_1\}$	1	1	1	1	1
$\{\mathbf{0}_1\}$	0	1	0	0	1
$\{\mathbf{1}_1, \mathbf{0}_0\}$	0	0	1	1	1
$\{\mathbf{1}_0, \mathbf{0}_0\}$	0	0	0	1	1
$\{\mathbf{0}_1, \mathbf{0}_0\}$	0	0	0	0	1

表4 对连通支配集的兼容性矩阵重新排列行和列后得到的最大三角子矩阵。

5.2.1 路径构件

为了排除任何 $\varepsilon > 0$ 下运行时间为 $\mathcal{O}^*((5-\varepsilon)^{\text{lin-cw}(G)})$ 的算法，我们必须构造一个路径构件，使其具有 5 种不同的状态，并逐步缩减到单个标签，从而使构造中的每一行都贡献一个线性团宽的单位元。我们首先分析部分解在一个标签上的可能行为。

首先，我们考虑单个顶点 v 相对于部分解X的可能状态。与连通顶点覆盖相比，这里多出一种状态，因为状态 $\mathbf{0}$ 被拆分为状态 $\mathbf{0}_1$ 和 $\mathbf{0}_0$，，分别表示 $v \notin X$ 以及 v 是否被 X 支配；而状态 $\mathbf{1}_1$ 和 $\mathbf{1}_0$则予以保留。因此，标签的状态可以表示为 $\{\mathbf{0}_0, \mathbf{0}_1, \mathbf{1}_0, \mathbf{1}_1\}$ 的一个子集。

与之前类似，我们研究这些标签状态在一次连接中的兼容性。结果是一个大小为 15×15 的矩阵，因为可以排除空集。不过，许多状态会产生相同的兼容模式，例如，对于任意子集 $\emptyset \neq \mathbf{S} \subseteq \{\mathbf{0}_0, \mathbf{0}_1, \mathbf{1}_0, \mathbf{1}_1\}$，状态 $\mathbf{S}$ 和 $\mathbf{S} \cup \{\mathbf{0}_1\}$ 会产生相同的兼容模式，因为顶点状态 $\mathbf{0}_1$不会增加任何额外约束。事实证明，兼容性矩阵仅包含五个不同的行，重排后可得到大小为 5×5 的三角子矩阵表4。令人惊讶的是，冗余如此之多，以至于尽管连通支配集有四种顶点状态，而连通顶点覆盖只有三种，我们最终得到的标签状态数却比连通顶点覆盖更少。

同样，只需将大小为二的独立集作为与连接相邻的标签。相关的标签状态将由以下顶点状态的有序对表示：$(\mathbf{0}_1, \mathbf{0}_0)$、$(\mathbf{0}_1, \mathbf{0}_1)$、$(\mathbf{1}_0, \mathbf{0}_0)$、$(\mathbf{1}_1, \mathbf{0}_0)$、$(\mathbf{1}_1, \mathbf{1}_1)$。随后，将这些状态沿三角子矩阵的对角线配对，即可按照转移顺序得到路径构件所需的状态。

状态的形式化定义。 我们定义四种原子状态 **原子组件** $= \{\mathbf{0}_0, \mathbf{0}_1, \mathbf{1}_0, \mathbf{1}_1\}$，并定义三个谓词 $\texttt{sol}$，$\texttt{conn}$，$\texttt{dom}$：**原子组件** $\to \{0, 1\}$，具体定义如下：$\texttt{sol}(\mathbf{a}) = [\mathbf{a} \in \{\mathbf{1}_0, \mathbf{1}_1\}]$,$\texttt{conn}(\mathbf{a}) = [\mathbf{a} = \mathbf{1}_1]$，以及 $\texttt{dom}(\mathbf{a}) = [\mathbf{a} = \mathbf{0}_1]$。原子状态 $\mathbf{0}_1$ 和 $\mathbf{0}_0$ 表示某个顶点不在部分解中，下标表示该顶点是否被部分解支配；$\mathbf{1}_1$ 和 $\mathbf{1}_0$ 表示某个顶点在部分解中，下标表示该顶点是否与根连通。基于这些原子状态，我们定义五种状态，每种状态由四个原子状态组成：

$$\mathbf{s}^1 = (\mathbf{0}_1, \mathbf{0}_0, \mathbf{1}_1, \mathbf{0}_1)$$
$$\mathbf{s}^2 = (\mathbf{0}_1, \mathbf{0}_1, \mathbf{0}_1, \mathbf{0}_1)$$
$$\mathbf{s}^3 = (\mathbf{1}_0, \mathbf{0}_0, \mathbf{1}_1, \mathbf{0}_0)$$
$$\mathbf{s}^4 = (\mathbf{1}_1, \mathbf{0}_0, \mathbf{1}_0, \mathbf{0}_0)$$
$$\mathbf{s}^5 = (\mathbf{1}_1, \mathbf{0}_1, \mathbf{0}_1, \mathbf{0}_0)$$

我们将这些状态汇集为集合 **状态** $=\{\mathbf{s}^1, \ldots\ldots, \mathbf{s}^5\}$，并使用记号 $\mathbf{s}_i^\ell \in$ **原子组件**,$\ell \in [5]$, $i \in [4]$ 表示状态 $\mathbf{s}^\ell$ 的第 i 个原子分量。注意，$\mathbf{s}^5$ 可通过交换前两个分量与后两个分量，从 $\mathbf{s}^1$ 得到；同理，$\mathbf{s}^4$ 可由 $\mathbf{s}^3$得到。

给定一个部分解 $Y \subseteq V(G)$，我们通过映射将每个顶点关联到其在 Y 中的状态 **状态**$_Y$: $V(G) \setminus \{\hat{r}\} \to$ **原子组件**，其定义如下:

$$\mathbf{state}_Y(v) = \begin{cases} \mathbf{0}_0 & \text{if } v \notin N[Y \cup \{\hat{r}\}], \\ \mathbf{0}_1 & \text{if } v \in N[Y \cup \{\hat{r}\}] \setminus Y, \\ \mathbf{1}_0 & \text{if } v \in Y \text{ and } v \text{ is not root-connected in } Y \cup \{\hat{r}\}, \\ \mathbf{1}_1 & \text{if } v \in Y \text{ and } v \text{ is root-connected in } Y \cup \{\hat{r}\}. \end{cases}$$

细分边。在图构造中，我们经常需要细分边。给定两个顶点 u 和 v，在 u 和 v 之间添加一条细分边，意味着添加一个新顶点$w_{\{u,v\}}$ 以及边$\{u, w_{\{u,v\}}\}$ 和$\{w_{\{u,v\}}, v\}$。 u 和 v 之间的细分边具有一个关键性质：任何连通支配集 X 都必须包含 u和 v中的至少一个，因为在整个构造过程中， u 和 v 始终是$w_{\{u,v\}}$ 仅有的邻居。从这个意义上说，对于细分边，连通支配集的行为类似于顶点覆盖对于普通边的行为，因此我们可以将连通顶点覆盖的构造中相当一部分改编用于连通支配集。

形式化构造。我们接下来描述如何构造路径构件 P。我们创建 4 个连接顶点 u_1,1,u_1,2,u_2,1, u_2,2、6 个辅助顶点 a_1,1, a_1,2, a_1,3, a_2,1, a_2,2, a_2,3、4 个解指示顶点 b_1,0, b_1,1,b_2,0, b_2,1、4 个连通性指示顶点 c_1,0, c_1,1, c_2,0, c_2,1、4 个支配指示顶点 d_1,0, d_1,1, d_2,0, d_2,1，以及 5 个团顶点 $v_1, \ldots, v_5$。对于每个 $i \in [2]$，我们添加边 {u_i,1,a_i,1}、{a_i,1, a_i,2}、{a_i,1, b_i,0}、{a_i,1, c_i,1} 和 {u_i,2, d_i,1}。此外，对于每个 $i \in [2]$，我们添加从 u_i,1 到 a_i,3 和 b_i,0 的细分边；从 a_i,3 到 b_i,1 的细分边；从 b_i,0 到 b_i,1 的细分边；从 c_i,0 到 c_i,1 的细分边；以及从 d_i,0 到 d_i,1 的细分边。

此外，对于所有 $i \in [2]$、所有解指示顶点、所有连通性指示顶点、所有支配指示顶点，以及所有与根顶点 $\hat{r}$ 相邻的团顶点，我们令 $a_{i,\ 3}$。我们在团顶点 v_ℓ, $\ell \in [5]$ 之间添加所有可能的细分边，使它们诱导出一个大小为 5 的团，其中每条边都经过细分。

最后，我们说明如何将指示顶点连接到团顶点。团顶点 v_ℓ 对应于在连接顶点 ($u_{1,\ 1}$, $u_{1,\ 2}$, $u_{2,\ 1}$, $u_{2,\ 2}$) 上选择状态 $\mathbf{s}^\ell$。部分解 X 关于 $P + \hat{r}$ 的预期行为如下:

- b_i,1 $\in X \iff u_i$,1 $\in X$,c_i,1 $\in X \iff u_i$,1 $\in X$ 和 u_i,1 在 $X \cup \{\hat{r}\}$,d_i,1 $\in X \iff u_i$,2 $\in N[X] \setminus X$ 中与根连通，即 u_1,2 被 X 支配。因此，对于所有 $i \in [2]$ 和 $\ell \in [5]$，我们添加从 v_ℓ 到 $b_{i,\mathtt{sol}(\mathbf{s}^\ell_{2i-1})}$、$c_{i,\mathtt{conn}(\mathbf{s}^\ell_{2i-1})}$ 和 $d_{i,\mathtt{dom}(\mathbf{s}^\ell_{2i})}$ 的细分边。至此， P 的构造完成。

单路径构件的行为。对于接下来的引理，我们假设 G 是一个图，包含 $P + \hat{r}$ 作为诱导子图，并且只有连接顶点 $u_{i,1}$,$u_{i,2}$, $i \in [2]$以及团顶点 $v_\ell, \ell \in [5]$在该副本 $P + \hat{r}$之外有邻居。此外，设 X是 G 的一个连通支配集，且有 $\hat{r} \in X$。我们研究这类连通支配集在 P上的行为；为避免繁琐，我们将用 $X \cap P$代替 $X \cap V(P)$。关于 P如何连接到其余图的假设意味着，任何状态为**状态**$_{X\cap P}(v) = \mathbf{1}_0$ 的顶点 $v \in V(P)$，都必须通过一条离开 $P + \hat{r}$的路径与 X 中的根连通

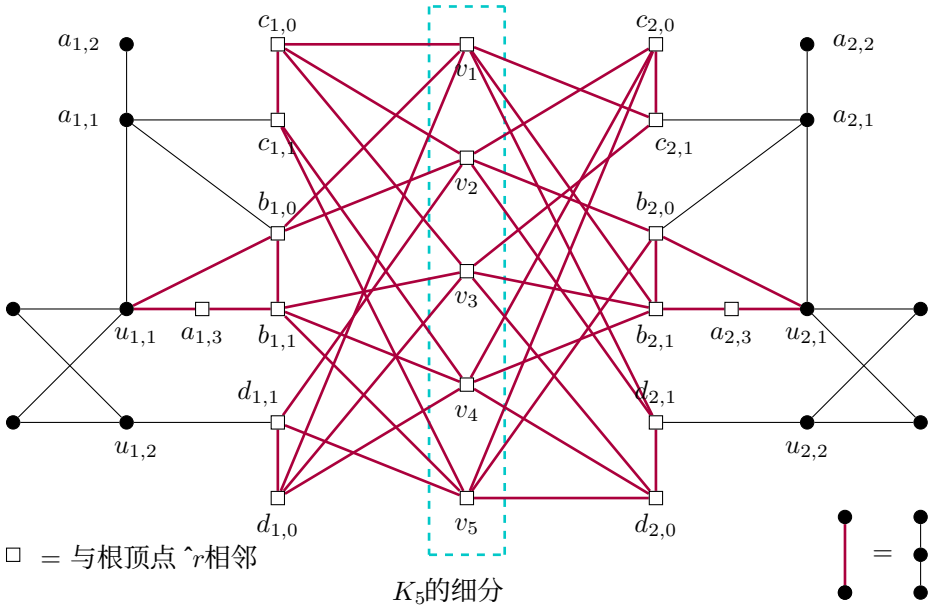

图 7 以线性团宽为参数的连通支配集问题的路径构件。方块表示的顶点与根顶点$\hat{r}$ 相邻，粗红色边表示细分边。由团顶点 v_ℓ, $\ell \in [5]$构成的细分 5-团的边未在图中绘出。除与$\hat{r}$的连接外，只有连接顶点 $u_{i,1}$，$u_{i,2}$, $i \in [2]$以及团顶点 v_ℓ, $\ell \in [5]$与此路径构件之外的部分相连。

该路径经过连接顶点 $u_{i,j}$, $i,j \in [2]$之一。注意，任何经由团顶点 v_ℓ, $\ell \in [5]$离开 $P + \hat{r}$ 的路径，都会立即导出一条在 $P + \hat{r}$ 中通往 $\hat{r}$ 的路径，因为$\{v_\ell : \ell \in [5]\} \subseteq N(\hat{r})$。

我们首先通过对子图进行顶点不相交打包，为 $|X \cap P|$给出一个下界。

▶ 引理 *5.9*。任何具有 $\hat{r} \in X$ 的连通支配集 X 都满足 $|X \cap P| \geq 14 = 2 \cdot 5 + 4$，更具体地说，对于所有 $i \in [2]$，都有 $a_{i,1} \in X$, $|X \cap \{b_{i,0}, b_{i,1}, a_{i,3}, u_{i,1}\}| \geq 2$, $X \cap \{c_{i,0}, c_{i,1}\} \neq \emptyset$, $X \cap \{d_{i,0}, d_{i,1}\} \neq \emptyset$ 以及 $|X \cap \{v_1, \ldots, v_5\}| \geq 4$。

证明。首先，注意以下每个顶点的闭邻域两两不相交：$a_{i,2}$、$w_{\{b_{i,0},b_{i,1}\}}$、$w_{\{a_{i,3},u_{i,1}\}}$、$w_{\{c_{i,0},c_{i,1}\}}$、$w_{\{d_{i,0},d_{i,1}\}}$ 对所有 $i \in [2]$均成立。由于 X 是一个支配集，X 在这些邻域中每个至少包含一个顶点。由于 X是连通的，因此可得$a_{i,1} \in X$，并且对于每条细分边，其一个端点必须包含在 X中。

接下来，我们转向细分的 5-团。同样，X 必须包含该团中每条细分边的至少一个端点。若存在两个团顶点 v_i，$v_j \notin X$、$i \neq j$，那么它们之间的细分边无法由 X解决。因此，X 至多可以不包含一个团顶点。证明至此完成。◀

对于任意部分解 $Y \subseteq V(P)$，我们用 4 元组 **状态**$(Y) =$ (**状态**$_Y(u_{1,1})$，**状态**$_Y(u_{1,2})$，**状态**$_Y(u_{2,1})$，**状态**$_Y(u_{2,2})$ 来形式化描述 P 的边界处的状态。下面的引理表明，传递到边界的状态以所需的方式取决于中心团的状态。

▶ 引理 *5.10*. 任意满足$a_{i,1} \in X$, $X \cap \{b_{i,0}, b_{i,1}, a_{i,3}, u_{i,1}\} \in \{\{b_{i,0}, a_{i,3}\}, \{b_{i,1}, u_{i,1}\}\}$, $|X \cap \{c_{i,0}, c_{i,1}\}| = 1$, $|X \cap \{d_{i,0}, d_{i,1}\}| = 1$ 的连通支配集 X *with* $\hat{r} \in X$ *and* $|X \cap P| \leq 14$，对所有 $i \in [2]$ 均成立，并且存在唯一的 $\ell \in [5]$ 使得 $v_\ell \notin X$。此外，我们有 *state*$(X \cap P) = s^\ell$。

证明. 在此情况下，引理 5.9 的所有不等式都必须取等号，这证明了第一部分的全部内容，唯独关于$\{b_i,0,b_i,1,a_i,3,u_i,1\}$的部分除外。我们知道 X 恰好包含这些顶点中的两个；但如果 X 包含由一条细分边相连的两个顶点，那么另外两个顶点之间的细分边就没有被解决。因此，$X \cap \{b_i,0, b_i,1, a_i,3, u_i,1\} \in \{\{b_i,0, a_i,3\}, \{b_i,1, u_i,1\}\}$。

还需证明关于连接顶点状态的性质。由于 $v_\ell \notin X$，关联细分边的另一端点必须包含在 X 中。根据构造，对于 $i \in [2]$，这些顶点分别是 $b_{i,\mathtt{sol}(\mathbf{s}^\ell_{2i-1})}$、$c_{i,\mathtt{conn}(\mathbf{s}^\ell_{2i-1})}$ 和 $d_{i,\mathtt{dom}(\mathbf{s}^\ell_{2i})}$。固定 $i \in [2]$，注意到根据 P 内部的预算分配，可知 $u_i,2 \notin X$。因此，$u_i,2$ 只能在 P 内由 $d_i,1$ 支配，并且状态$_{X\cap P}(u_i,2) = \mathbf{0}_1 \iff d_i,1 \in X \iff \mathtt{dom}(\mathbf{s}^\ell_{2i}) = 1 \iff \mathbf{s}^\ell_{2i} = \mathbf{0}_1$。我们有 $u_i,1 \in X \iff b_i,1 \in X \iff \mathtt{sol}(\mathbf{s}^\ell_{2i-1}) = 1 \iff \mathbf{s}^\ell_{2i-1} \in \{\mathbf{1}_0, \mathbf{1}_1\}$，并且 $u_i,1$ 始终被 $a_i,1$ 支配，因此在所有情况下，状态$_{X\cap P}(u_i,1) \neq \mathbf{0}_0$。如果 $u_i,1 \in X$，那么其在 $X \cap P$ 内的唯一邻居是 $a_i,1$；当且仅当$c_i,1 \in X$时，该邻居使根连通性达到 $u_i,1$。因此，状态$_{X\cap P}(u_i,1) = \mathbf{1}_1 \iff c_i,1 \in X \iff \mathtt{conn}(\mathbf{s}^\ell_{2i-1}) = 1 \iff \mathbf{s}^\ell_{2i-1} = \mathbf{1}_1$。证明完毕。◀

接下来，我们证明对于任意 $\mathbf{s}^\ell \in$状态，实际上存在一个达到 $\mathbf{s}^\ell$ 的部分解，并且这些部分解与根连通，且支配 P 内的所有元素，只要这一性质在连接顶点 $u_{i,j}, i, j \in [2]$处成立。

▶ **引理 5.11.** 对于每个 $\ell \in [5]$，都存在一个由 P 构成的集合 X^ℓ_P，使得 $|X^\ell_P| = 14$，$X^\ell_P \cap \{v_1, \ldots, v_5\} = \{v_1, \ldots, v_5\} \setminus \{v_\ell\}$，状态$(X^\ell_P) = \mathbf{s}^\ell$，并且

$$\mathbf{state}_{X^\ell_P}(V(P) \setminus \{a_{1,1}, a_{2,1}, u_{1,1}, u_{1,2}, u_{2,1}, u_{2,2}\}) \subseteq \{\mathbf{0}_1, \mathbf{1}_1\}.$$

若 X 是 G 的一个顶点子集，满足 $\hat{r} \in X$ 和 $X \cap P = X^\ell_P$，且对于每个 $i \in [2]$，都有 状态$_X(\{u_{i,1}, u_{1,2}\}) \subseteq \{\mathbf{0}_1, \mathbf{1}_1\}$，那么 状态$_X(V(P)) \subseteq \{\mathbf{0}_1, \mathbf{1}_1\}$。

证明。 我们定义

$$\begin{aligned} X^\ell_P = {} & (\{v_1, \ldots, v_5\} \setminus \{v_\ell\}) \cup \{a_{i,1} : i \in [2]\} \\ & \cup \{u_{i,1} : i \in [2] \text{ and } \mathtt{sol}(\mathbf{s}^\ell_{2i-1}) = 1\} \cup \{a_{i,3} : i \in [2] \text{ and } \mathtt{sol}(\mathbf{s}^\ell_{2i-1}) = 0\} \\ & \cup \{b_{i,\mathtt{sol}(\mathbf{s}^\ell_{2i-1})} : i \in [2]\} \cup \{c_{i,\mathtt{conn}(\mathbf{s}^\ell_{2i-1})} : i \in [2]\} \cup \{d_{i,\mathtt{dom}(\mathbf{s}^\ell_{2i})} : i \in [2]\} \end{aligned}$$

并声称 X^ℓ_P 具有所需的性质。显然，$|X^\ell_P| = 14$ 和 $v_\ell \notin X^\ell_P$。下面证明，除$a_{1,1}$，$a_{2,1}$，$u_{1,1}$，$u_{1,2}$，$u_{2,1}$，$u_{2,2}$这 6 种情况外，P中的每个顶点都被 X^ℓ_P 支配，或者在 $X^\ell_P + \hat{r}$中与根连通。

首先，注意对于任意顶点 $v \in N(\hat{r}) \cap V(P)$，都有状态$^{X^\ell_P}(v) \in \{\mathbf{0}_1, \mathbf{1}_1\}$。由于$a_i,1 \in X^\ell_P$和$a_i,2 \notin X^\ell_P$，对于每个 $i \in [2]$，都有状态$^{X^\ell_P}(a_i,2) = \mathbf{0}_1$。剩下需要处理的是细分顶点$^{w_{\{x,y\}}}$。由于 X^ℓ_P 包含每对指示顶点中的一个顶点，因此这对顶点之间的每条细分边都已解决。根据构造，对于每个 $i \in [2]$，X^ℓ_P要么包含顶点对$\{b_i,0,a_i,3\}$，要么包含顶点对$\{b_i,1, u_i,1\}$；无论哪种情况，与$a_i,3$和$u_i,1$关联的细分边都已解决。团顶点之间的细分边也已解决，因为 X^ℓ_P 包含 5 个团顶点中的 4 个。最后，v_ℓ 与指示顶点之间的细分边也已解决，因为根据P的构造，与 v_ℓ 关联的细分边分别通向$b_{i,\mathtt{sol}(\mathbf{s}^\ell_{2i-1})}$、$c_{i,\mathtt{conn}(\mathbf{s}^\ell_{2i-1})}$和$d_{i,\mathtt{dom}(\mathbf{s}^\ell_{2i})}$，其中所有 $i \in [2]$ 恰好是 X^ℓ_P中包含的指示顶点。

该状态$(X^\ell_P) = \mathbf{s}^\ell$ 可通过与引理 5.10 的证明中非常相似的论证得到。下面处理第二部分。根据第一部分，剩下需要处理的是顶点$a_{1,1}$，$a_{2,1}$，$u_{1,1}$，$u_{1,2}$，$u_{2,1}$，$u_{2,2}$。根据假设，不包含在 X 中的连接顶点

均被支配，而包含在 X 中的那些顶点均与根连通。根据 X_P^ℓ的定义，我们看到 $a_{1,\,1}$，$a_{2,\,1} \in X$，因此剩下需要确立 $a_{1,\,1}$ 和 $a_{2,\,1}$ 的根连通性。

我们通过考察 state$^{X_P^\ell}(u_i,1)$ 的不同情形，证明 $a_i,1$ 与根连通。注意，由于 $a_i,1 \in X_P^\ell$，state$^{X_P^\ell}(u_i,1) \neq \mathbf{0}_0$。如果 state$^{X_P^\ell}(u_i,1) = \mathbf{0}_1$，那么 $a_i,1$ 通过 X 中的路径 $a_i,1,b_i,0,\hat{}r$ 与根连通。如果 state$^{X_P^\ell}(u_i,1) = \mathbf{1}_0$，那么 $u_i,1$ 在X 中通过某条离开 $P+\hat{}r$ 的路径与根连通，我们可以将该路径延伸至 $a_i,1$。最后，如果 state$^{X_P^\ell}(u_i,1) = \mathbf{1}_1$，则 $G[X]$ 中存在路径 $a_i,1,c_i,1,\hat{}r$。证明完毕。◀

状态转移。在下界构造中，我们再次通过反复连接路径构件 P 来构造长路径。为了研究状态如何在两个连续的路径构件之间发生变化，假设我们有 P 的两个副本 P^1 和 P^2，其中 P^1 中的顶点 $u_2,1$ 和 $u_2,2$ 分别与 P^2 中的顶点 $u_1,1$ 和 $u_1,2$ 相连。我们用上标 1 标记 P^1 的顶点，用上标 2 标记 P^2 的顶点，例如，$u^1_2,1$ 指的是 P^1 的顶点$u_2,1$。同样，假设 P^1 和 P^2作为诱导子图嵌入更大的图 G 中，该图具有根顶点 $\hat{}r$，且只有 $P^1 + P^2 + \hat{}r$ 外部的顶点 $u^1_1,1,u^1_1,2,u^2_2,1,u^2_2,2$ 以及团顶点 v^1_ℓ, v^2_ℓ, $\ell \in [5]$ 拥有邻居。此外，X表示 G 的一个包含 $\hat{}r \in X$ 的连通支配集。

前面的引理现在将使我们能够得出结论：构件状态确实只能按照转移顺序进行转移，并且状态可以保持稳定。

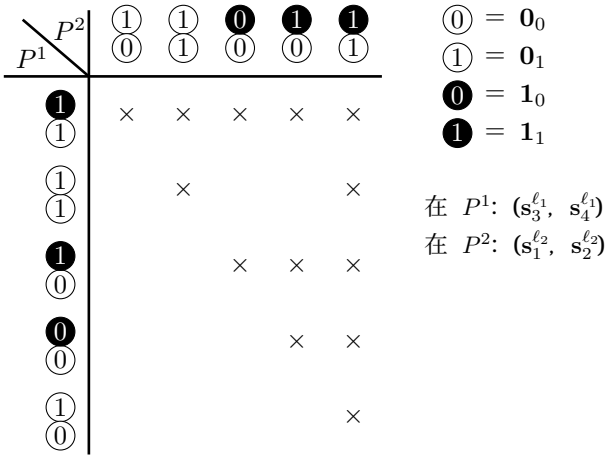

图 8 描述两个连续路径构件之间可能的状态转移的矩阵。行标记为 $(\mathbf{s}_3^{\ell_1}, \mathbf{s}_4^{\ell_1})$、$\ell_1 \in [5]$，列标记为 $(\mathbf{s}_1^{\ell_2}, \mathbf{s}_2^{\ell_2})$、$\ell_2 \in [5]$。× 表示连通支配集中的可能状态转移。

▶ **引理 5.12。 </style}如果 满足 和，则 状态**$(X \cap P^1) = \mathbf{s}^{\ell_1}$，并且 **状态**$(X \cap P^2) = \mathbf{s}^{\ell_2}$，其中 $\ell_1 \leq \ell_2$。

此外，对于每个 $\ell \in [5]$，集合 $X^\ell = X^\ell_{P^1} \cup X^\ell_{P^2}$ 支配或通过根连接所有内部连接顶点，即 **状态**$_{X^\ell}$（$\{u^1_{2,\,1}, u^1_{2,\,2}, u^2_{1,\,1}, u^2_{1,\,2}\}$）$\subseteq \{\mathbf{0}_1, \mathbf{1}_1\}$。

证明。根据引理 5.10，存在 $\ell_1, \ell_2 \in [5]$，使得 state$(X \cap P^1) = \mathbf{s}^{\ell_1}$，且 state$(X \cap P^2) = \mathbf{s}^{\ell_2}$。接下来只需验证 $\ell_1 \leq \ell_2$。主要思路是考察内部连接顶点 $u^1_2,1,u^1_2,2$ 和 $u^2_1,1, u^2_1,2$ 处的状态，并注意到只要 $\ell_1 > \ell_2$，这四个顶点中至少有一个未被支配或不与根连通。回想一下，这些内部连接顶点仅与 $P^1 + P^2$ 和 $N(\{u^1_2,1, u^1_2,2\}) \cap V(P^2) = \{u^2_1,1, u^2_1,2\}$ 以及 $N(\{u^2_1,1, u^2_1,2\}) \cap V(P^1) = \{u^1_2,1, u^1_2,2\}$ 内部的顶点相邻。图 8 展示了可能的状态转移。

首先，若 $\ell_1 \in \{3, 4, 5\}$ 且 $\ell_2 \in [2]$，则 **状态**$_{X \cap P^1}(u^1_{2,2}) = \mathbf{0}_0$，且 **状态**$_{X \cap P^2}(\{u^2_{1,1}, u^2_{1,2}\}) \subseteq \{\mathbf{0}_0, \mathbf{0}_1\}$。因此，在这种情况下，顶点 $u^1_{2,2}$ 不能被 X 支配。

其次，若 $\ell_1 \in \{4,5\}$ 且 $\ell_2 = 3$，则 $\mathbf{1}_1 \notin$ **状态**$_{X \cap P^1}(\{u^1_{2,1}, u^1_{2,2}\})$ 和 **状态**$_{X \cap P^2}(u^2_{1,1}) = \mathbf{1}_0$。因此，在这种情况下，顶点 $u^2_{1,1}$ 无法与 X 中的根连通。

最后，若 $(\ell_1, \ell_2) \in \{(2,1), (5,4)\}$，则有 **状态**$_{X \cap P^1}(\{u^1_{2,1}, u^1_{2,2}\}) \subseteq \{\mathbf{0}_0, \mathbf{0}_1\}$ 且 **状态**$_{X \cap P^2}(u^2_{1,2}) = \mathbf{0}_0$。因此，在这种情况下，顶点 $u^2_{1,2}$ 无法被 X 支配。

对于引理的第二部分，固定某个 $\ell \in [5]$。根据引理 5.11，我们有 **状态**$(X^\ell \cap P^1) =$ **状态**$(X^\ell \cap P^2) = \mathbf{s}^\ell$。我们根据 $\ell \in [5]$ 进行区分；另见图 9：

- $\ell = 1$：$u^2_{1,2}$ 被 $u^1_{2,1}$ 支配。$\ell = 2$：根据引理 5.11，全部四个顶点都已被支配。$\ell = 3$：$u^1_{2,1}$ 与 $u^2_{1,1}$ 根连通，并支配 $u^2_{1,2}$；$u^2_{1,1}$ 支配 $u^1_{2,2}$。$\ell = 4$：$\ell = 3$ 的反向情况。$\ell = 5$：$\ell = 1$ 的反向情况。至此，第二部分的证明完成。

◀

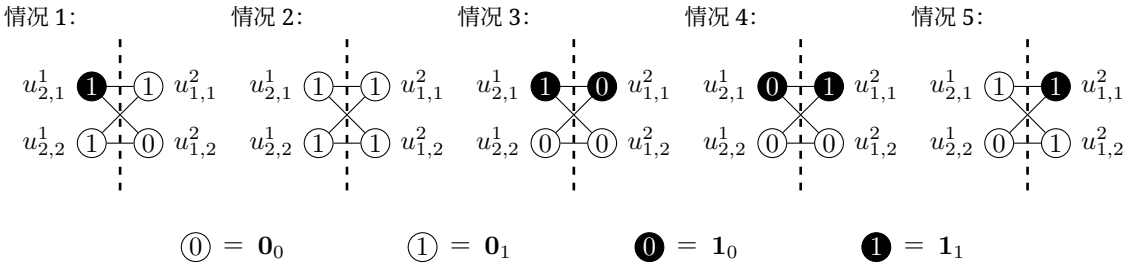

图 9 用于证明 **状态**$_{X^\ell}(\{u^1_{2,1}, u^1_{2,2}, u^2_{1,1}, u^2_{1,2}\}) \subseteq \{\mathbf{0}_1, \mathbf{1}_1\}$ 的分类讨论。

当 $\ell_1 < \ell_2$ 时，我们称发生了作弊。构造任意长的路径构件串接时，引理 5.12 告诉我们，这样的路径上至多发生 $|$**状态**$| - 1 = 4 = \mathcal{O}(1)$ 次作弊，因此我们可以像之前概述的那样找到一个无作弊区域。

5.2.2 完整构造

准备工作。 假设连通支配集可以在时间 $\mathcal{O}^*((5-\varepsilon)^k)$ 内求解，其中 $\varepsilon > 0$ 为某个参数，输入为一个线性 k-表达式。给定一个包含 n 个变量和 m 个子句的可满足性实例 σ，我们构造一个等价的连通支配集实例，其线性团宽约为 $n \log_5(2)$，从而若存在这样的连通支配集算法，则强指数时间假设不成立。

我们选取一个仅取决于 ε 的整数 β；β 的具体取值将在稍后讨论。将 σ 的变量划分为若干组，每组至多包含 β 个变量，从而得到 $t = \lceil n/\beta \rceil$ 组。此外，我们选取满足 $5^p \geq 2^\beta$ 的最小整数 p，即 $p = \lceil \log_5(2^\beta) \rceil$。现在开始构造连通支配集实例 $(G = G(\sigma, \beta), b)$。

我们创建根顶点 $\hat{r}$，并连接一个叶节点 $\hat{r}'$，以此迫使 $\hat{r}$ 属于任何连通支配集。对于每个组 $i \in [t]$，我们创建 p 个长类路径构件 $P^{i,j}$，$j \in [p]$，其中每个 $P^{i,j}$ 都由 $m(4tp+1)$ 个路径构件 P 的副本 $P^{i,j,\ell}$，$\ell \in [m(4tp+1)]$，组成，相邻副本通过连接相连。更具体地说，某个 $P^{i,j,\ell}$ 中的顶点沿用通用路径构件 P 中的名称，并带有 $P^{i,j,\ell}$ 的上标；对于每个 $i \in [t]$，$j \in [p]$，$\ell \in [m(4tp+1) - 1]$，顶点 $\{u^{i,j,\ell}_{2,1}, u^{i,j,\ell}_{2,2}\}$

与顶点 $\{u^{i,}_{,}{}_{j\,\ell+1}{}_{,}{}_{,}{}_{,}\ u^{i,}_{,}{}_{j\,\ell+1}{}_{,}{}_{,}\}$ 相连。每条路径 $P^{i,}_{,}{}_{j}$ 的两端，即顶点 $u^{i,}_{,}{}_{j\,1}{}_{,}{}_{,}{}_{,}\ u^{i,}_{,}{}_{j\,1}{}_{,}{}_{,}{}_{,}\ u^{i,}_{,}{}_{j\,m(4tp+1)}{}_{,}{}_{,}{}_{,}\ u^{i,}_{,}{}_{j\,m(4tp+1)}{}_{,}{}_{,}{}_{,}$ 均与根 $\hat{r}$ 相邻。

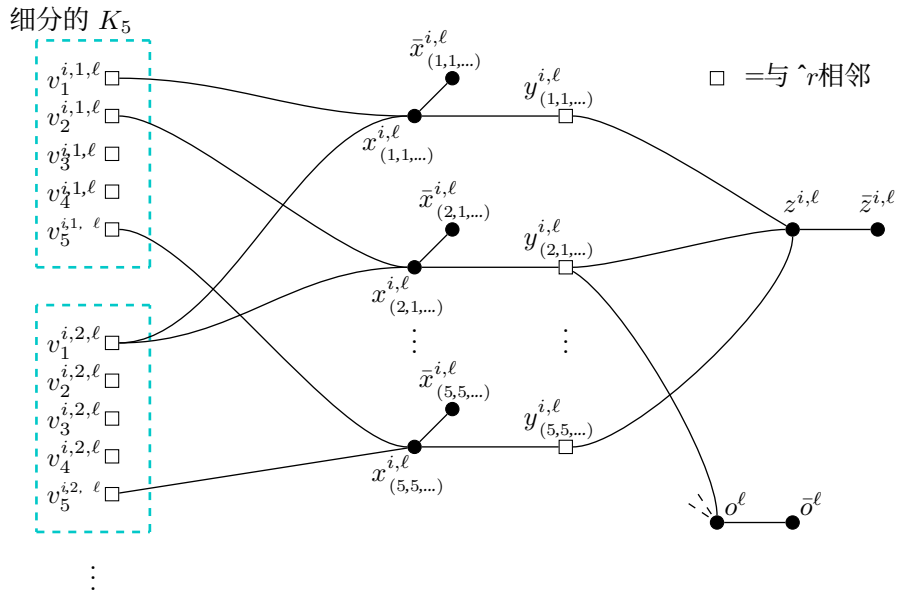

图10 连通支配集的解码构件和子句构件。

对于每个组 $i \in [t]$ 和列 $\ell \in [m\ (4tp+1)\ ],$，我们创建一个解码构件 $D^{i,}_{,}{}_{\ell}$，其样式与 Cygan 等人 [17] 针对以路径宽度为参数的连通顶点覆盖所采用的样式相同。每个变量组 i 至多有 2^β 种可能的真值赋值，并且根据 p 的选择，我们有 $5^p \geq 2^\beta$，因此可以找到一个单射 κ：$\{0,1\}^\beta \to [5]^p$，它将每个真值赋值 $\tau \in \{0,1\}^\beta$ 分配给一个序列 $\kappa\ (\tau)\ \in [5]^p$。对于每个序列 $\mathbf{h}\ = (h_1,\ \ldots,\ h_p)\ \in [5]^p$，我们创建顶点 $x^{i,}_{,}{}_{\ell}\mathbf{h}$、$\bar{x}^{i,}_{,}{}_{\ell}\mathbf{h}$、$y^{i,}_{,}{}_{\ell}\mathbf{h}$，以及边 $\{x^{i,}_{,}{}_{\ell}\mathbf{h}, \bar{x}^{i,}_{,}{}_{\ell}\mathbf{h}\}$、$\{x^{i,}_{,}{}_{\ell}\mathbf{h}, y^{i,}_{,}{}_{\ell}\mathbf{h}\}$、$\{y^{i,}_{,}{}_{\ell}\mathbf{h}, \hat{r}\}$。此外，对于所有 $\mathbf{h}\ = (h_1,\ \ldots,\ h_p)\ \in [5]^p$ 和 $j \in [p]$，我们添加边 $\{x^{i,}_{,}{}_{\ell}\mathbf{h}, v^{i,}_{,}{}_{j\,\ell}\}$。最后，我们创建两个相邻的顶点 $z^{i,}_{,}{}_{\ell}$ 和 $\bar{z}^{i,}_{,}{}_{\ell}$，并添加边 $\{z^{i,}_{,}{}_{\ell}, y^{i,}_{,}{}_{\ell}\mathbf{h}\}$（对所有 $\mathbf{h}^{h_j} \in [5]^p$）。每个解码构件 $D^{i,}_{,}{}_{\ell}$ 与相邻的路径构件 $P^{i,}_{,}{}_{j,}{}_{\ell,}\ j \in [p]$ 共同构成块 $B^{i,}_{,}{}_{\ell}$；不同块之间连接关系的高层示意图见图11。

最后，我们构造子句构件。我们将 σ 的子句编号为 $C_0, \ldots, C_{m-1}$。对于每一列 $\ell \in [m\ (4tp+1)],$，我们创建一对相邻的顶点 o^ℓ 和 $\bar{o}^\ell$。令 $\ell' \in [0, m-1]$ 为 $(\ell-1)$ 模 m 的余数；对于所有满足以下条件的 $i \in [t]$，$\mathbf{h} \in \kappa(\{0,1\}^\beta)$：$\kappa^{-1}(\mathbf{h})$ 是变量组 i 的一个真值赋值，且满足子句 $C_{\ell'}$，我们添加边 $\{o^\ell, y^{i,}_{,}{}_{\ell}\mathbf{h}\}$。解码过程和子句构件的示意图见图 10。

▶ **引理 5.13**。如果 σ 可满足，那么存在一个连通支配集 X of $G = G(\sigma,\beta)$ 大小为 $|X| \leq (14tp + (5^p+2)t + 1)m(4tp+1)\ + 1 = b$。

证明。令 τ 为 σ 的一个满足真值赋值，并令 τ^i 表示 τ 在 i-th 个变量组上的限制，对每个 $i \in [t]$，并令 $\kappa(\tau^i)\ = \mathbf{h}^i = (h_1^i, \ldots, h_p^i) \in [5]^p$ 为相应的序列。连通支配集由下式给出：

$$X = \{\hat{r}\} \cup \bigcup_{\ell \in [m(4tp+1)]} \left(\{o^\ell\} \cup \bigcup_{i \in [t]} \left(\{y^{i,\ell}_{\mathbf{h}^i}, z^{i,\ell}\} \cup \bigcup_{\mathbf{h} \in [5]^p} \{x^{i,\ell}_{\mathbf{h}}\} \cup \bigcup_{j \in [p]} X^{h_j^i}_{P^{i,j,\ell}} \right) \right),$$

其中 $X^{h_j^i}_{P^{i,\ j,\ \ell}}$ 表示引理 5.11 给出的集合。

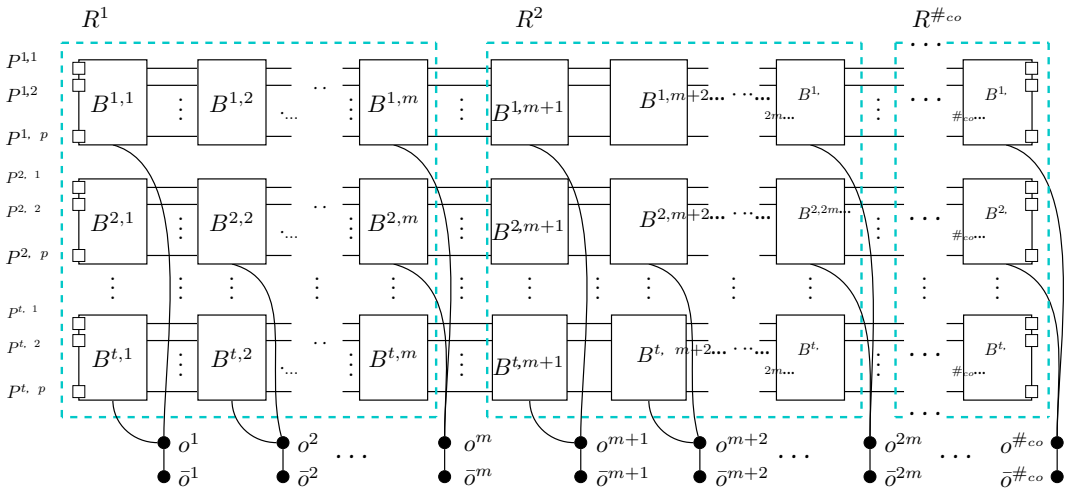

图11 连通支配集的高层次构造。两个块 $B^{i,\ell}$ 和 $B^{i,\ell+1}$ 之间的每条边都表示一次连接。区域 R^γ 对于引理 5.14 的证明十分重要。

显然，对于所有 i，j，ℓ，$|X| = b$ 如 $|X \cap P^{i,j,\ell}_{h^i_j}| = 14$，因此只需证明 X 是一个连通支配集。我们先考虑路径构件。首先，注意到我们有

$$\bigcup_{i\in[t]} \bigcup_{j\in[p]} \{u^{i,j,1}_{1,1}, u^{i,j,1}_{1,2}, u^{i,j,m(4tp+1)}_{2,1}, u^{i,j,m(4tp+1)}_{2,2}\} \subseteq N(\hat{r}),$$

因此，路径 $P^{i,j}$ 两端的顶点在 X中被支配或与根连通。接下来，我们应用引理 5.11 和引理 5.12 的第二部分，得出路径构件上的所有顶点在 X 中均与根连通或被支配。

固定 $i \in [t]$, $\ell \in [m(4tp+1)]$, , 并考虑相应的解码构件。由于$z^{i,\ell} \in X$ 以及 $x^{i,\ell}\,\mathbf{h} \in X$ 对所有$\mathbf{h} \in [5]^p$均成立，解码构件中的所有顶点都被X支配。此外，由于 $o^\ell \in X$，子句构件中的所有顶点都被 X支配。因此， X 必须是 G的一个支配集。

还需证明，属于 X的解码构件和子句构件中的顶点也与根连通。同样，固定 $i \in [t]$, $\ell \in [m(4tp+1)]$, , 并令 $\mathbf{h} = (h_1, \ldots, h_p) \in [5]^p \setminus \{\mathbf{h}^i\}$。由于 $\mathbf{h} \neq \mathbf{h}^i$，存在某个 $j \in [p]$ 使得 $h_j \neq h^i_j$，因此，根据引理 5.11， $v^{i,\ell}_{h_j} \in X$ 将 $x^{i,\ell}\,\mathbf{h}$ 与根 $\hat{r}$连通。顶点 $x^{i,\ell}\,\mathbf{h}_i$ 和 $z^{i,\ell}$ 通过 $y^{i,\ell}\,\mathbf{h}_i \in X$与根连通。

最后，我们证明对于所有 $\ell \in [m(4tp+1)]$， o^ℓ 均与根连通。由于 τ 是 σ的满足真值赋值，存在某个变量组 $i \in [t]$ ，使得 τ^i 已满足子句 $C_{\ell'}$，其中 ℓ' 是 $(\ell-1)$ 模 m的余数。根据 G和 X的构造，顶点 $y^{i,\ell}\,\mathbf{h}_i \in X$ 与 o^ℓ相邻，因为 $\kappa(\tau^i) = \mathbf{h}^i$，并将 o^ℓ 与根 $\hat{r}$连通。这表明 X 的所有顶点都与根连通，因此 $G[X]$ 必定连通。◀

▶ **引理 5.14.** 若存在 $G = G(\sigma, \beta)$ 的一个大小为$|X| \leq (14tp + (5^p+2)t + 1)m(4tp+1) + 1 = b$的连通支配集 X ，则 σ 可满足。

证明。 我们首先说明， X 必须满足 $|X| = b$。首先，必有$\hat{r} \in X$，因为$\hat{r}$ 有一个度为 1 的邻居。由引理 5.9 可知，对于所有$i \in [t]$、 $j \in [p]$、 $\ell \in [m(4tp+1)]$，都有 $|X \cap P^{i,j,\ell}| \geq 14$。在每个解码构件中，即每个 $i \in [t]$ 和$\ell \in [m(4tp+1)]$, 对应的构件中，集合$\{z^{i,\ell}\} \cup \bigcup_{\mathbf{h}\in[5]^p} x^{i,\ell}\,\mathbf{h}$ 必须包含在 X中，因为其中每个顶点

都与一个度为 1 的顶点相邻。此外，为了将 $z^{i,\ell}$ 连接到$\hat{}r$，顶点 $y^{i,\ell}{}_{\mathbf{h}}$、$\mathbf{h} \in [5]^p$中至少有一个必须包含在 X中。因此，每个解码构件中， X 至少必须包含 $5^p + 2$个顶点。最后，对于所有 $\ell \in [m(4tp + 1)]$,，都有 $o^\ell \in X$，因为 o^ℓ 有一个度为 1 的邻居。由于我们只考虑了互不相交的顶点集，这表明 $|X| = b$ 以及此前所有不等式都必须取等。

根据引理 5.10，我们知道 X 在每个 $P^{i,\ j,\ \ell}$ 上都取五种可能状态之一。固定某个 $P^{i,\ j} = \bigcup_{\ell\in[m(4tp+1)]} P^{i,\ j,\ \ell}$，并注意到，由于引理 5.12，沿着 $P^{i,\ j}$，状态至多改变四次。这种状态改变称为作弊。令 $\gamma \in [0, 4tp]$，并定义第 γ 个区域 $R^\gamma = \bigcup_{i\in[t]} \bigcup_{j\in[p]} \bigcup_{\ell=\gamma m+1}^{(\gamma+1)m} P^{i,\ j,\ \ell}$。由于共有 $4tp + 1$ 个区域，至少存在一个区域 R^γ，使得其中的 R^γ 不会发生作弊。在证明余下部分中，我们将考察这个区域，并从中读出一个满足真值赋值。

对于 $i \in [t]$，定义 $\mathbf{h}^i = (h^i_1, \ldots, h^i_p) \in [5]^p$，使得对于所有 $j \in [p]$， $v^{i,\ j\ \gamma m+1}_{h^i_j} \notin X$；根据引理 5.10，这一定义是良定义的。由于 R^γ 不包含任何作弊，$\mathbf{h}^i$ 的定义与我们考察哪一列 $\ell \in [\gamma m + 1,(\gamma + 1)m]$ 无关。对于每个 $i \in [t]$ 和 $\ell \in [\gamma m + 1, (\gamma + 1)m]$，我们断言 $y^{i,\ell}\mathbf{h} \in X$ 当且仅当 $\mathbf{h} = \mathbf{h}^i$。我们已经证明，对于每个 i 和 ℓ，恰好存在一个 $\mathbf{h}$，使得 $y^{i,\ell}\mathbf{h} \in X$。考虑顶点 $x^{i,\ell}\mathbf{h}_i \in X$，它在 G 中的邻居为 $v^{i,\ 1\ \ell}_{h^i_1}, v^{i,\ 2\ \ell}_{h^i_2}, \ldots, v^{i,\ p\ \ell}_{h^i_p}, \bar{x}^{i,\ell}\mathbf{h}^i$，以及 $y^{i,\ell}\mathbf{h}^i$。根据 $\mathbf{h}^i$ 的构造以及预算的紧分配，我们有 $(N_G(x^{i,\ell}\mathbf{h}^i) \setminus \{y^{i,\ell}\mathbf{h}^i\}) \cap X = \emptyset$。因此，$X$ 必须包含 $y^{i,\ell}\mathbf{h}^i$，才能将 $x^{i,\ell}\mathbf{h}^i$ 连接到根 $\hat{}r$。这就证明了该断言。

对于 $i \in [t]$，我们通过以下方式定义变量组 i 的真值赋值 τ^i：若 $\mathbf{h}^i \notin \kappa(\{0, 1\}^\beta)$，则任取一个真值赋值；否则令 $\tau^i = \kappa^{-1}(\mathbf{h}^i)$。通过令 $\tau = \bigcup_{i\in[t]} \tau^i$，我们得到所有变量的一个真值赋值，并声称 τ 满足 σ。考虑某个子句 $C_{\ell'}$， $\ell' \in [0, m - 1]$，并令 $\ell = \gamma m + \ell' + 1$。我们已经论证了 $o^\ell \in X$，而要将 o^ℓ 连接到根 $\hat{}r$，就必须存在某个 $y^{i,\ell}\mathbf{h} \in N_G(o^\ell) \cap X$。根据前面的论断，$\mathbf{h} = \mathbf{h}^i$ 对某个 $i \in [t]$ 成立，因此 τ^i，以及 τ，都满足子句 $C_{\ell'}$，这是由 G的构造所致。由于 $C_{\ell'}$ 的选择是任意的， τ 必定是 σ 的一个满足赋值。◀

▶ **引理 5.15**。 所构造的图 $G = G(\sigma, \beta)$ 具有 线性团宽$(G) \leq tp + 3 \cdot 5^p + O(1)$，并且可以在多项式时间内构造出一个具有该宽度的线性团表达式。

证明。 我们将描述如何为宽度为 $tp + 3 \cdot 5^p + \mathcal{O}(1)$ 的 G 构造一个线性团表达式。对于每条长路径 $P^{i,j}, i \in [t], j \in [p]$，团表达式将使用一个路径标签（$i, j$）；对于解码构件中的每个顶点，使用临时解码标签，即 $3 \cdot 5^p + 2$ 个；对于路径构件中的每个顶点，使用临时构件标签；对于子句构件，使用两个临时子句标签；此外还使用一个根顶点标签和一个垃圾标签。

团表达式的构造在算法 2 中进行了描述，其基本思路是在每一列中逐列、逐组进行处理。通过重复使用临时标签，我们使标签总数保持较小。同时使用的标签的最大数量出现在第 7 行，为 $tp + (3 \cdot 5^p + 2) + \mathcal{O}(1)$。证明完毕。◀

▶**定理 5.16**。 除非 *CNF-SETH* 不成立，否则不存在一种算法，能够在时间 $O^*((5 - \varepsilon)^k)$ 内，给定一个线性 k-表达式来求解连通支配集问题，其中 $\varepsilon > 0$。

证明。 假设存在一个$\mathcal{A}$ 算法，能够针对给定的线性 k-表达式，在$\mathcal{O}^*((5 - \varepsilon)^k)$时间内解决连通支配集问题，其中 $\varepsilon > 0$。给定 β，我们定义 $\delta_1 < 1$，使得 $(5 - \varepsilon)^{\log_5(2)} = 2^{\delta_1}$；并定义 δ_2，使得 $(5 - \varepsilon)^{1/\beta} = 2^{\delta_2}$。通过选取足够大的 β，我们可以确保 $\delta = \delta_1 + \delta_2 < 1$。我们将展示如何使用$\mathcal{A}$ 在$\mathcal{O}^*(2^{\delta n})$时间内解决可满足性问题，其中 n 为变量数，从而与 CNF-SETH 矛盾。

算法 2 为 G 构造线性团表达式。

```
引入带有根标签的 r̂ 和带有垃圾标签的 r̂′，并添加边 {r̂, r̂′};
对于 ℓ ∈ [m(4tp + 1)]，执行
    引入 o^ℓ 和 ō^ℓ，并添加带有子句标签的边 {o^ℓ, ō^ℓ};
    对于 i ∈ [t]，执行
        使用解码标签构建 D^{i,ℓ}，并向 o^ℓ 添加边;
        对 j ∈ [p] 执行
            使用装置标签构建 P^{i,j,ℓ}，并向 D^{i,j} 添加边;
            连接 u^{i,j,ℓ}_{1,1} 和 u^{i,j,ℓ}_{1,2} 如果 ℓ > 1，则标记为路径标签 (i, j)，否则标记为根标签;
            将路径标签 (i, j) 重新标记为垃圾标签;
            重新标记 u^{i,j,ℓ}_{2,1} 和 u^{i,j,ℓ}_{2,2} 为路径标签 (i, j);
            将所有其他构件标签重新标记为垃圾标签;
        将所有解码标签重新标记为垃圾标签;
    将所有子句标签重新标记为垃圾标签;
```

给定一个可满足性实例 σ，我们在多项式时间内构造 $G = G(\sigma, \beta)$ 以及引理 5.15 中的线性团-表达式。注意，我们有 $\beta = \mathcal{O}(1)$，因此$p = \mathcal{O}(1)$；回忆一下，$p = \lceil \log_5(2^\beta) \rceil$。然后，我们在 G 上运行$\mathcal{A}$ 并返回其答案。根据引理 5.13 和引理 5.14，该算法是正确的。根据引理 5.15，运行时间为

$$\begin{aligned}
& \mathcal{O}^*\left((5-\varepsilon)^{tp+3\cdot 5^p+\mathcal{O}(1)}\right) \leq \mathcal{O}^*\left((5-\varepsilon)^{tp}\right) && \leq \mathcal{O}^*\left((5-\varepsilon)^{\lceil \frac{n}{\beta} \rceil p}\right) \\
\leq\ & \mathcal{O}^*\left((5-\varepsilon)^{\frac{n}{\beta} p}\right) && \leq \mathcal{O}^*\left((5-\varepsilon)^{\frac{n}{\beta}\lceil \log_5(2^\beta) \rceil}\right) \leq \mathcal{O}^*\left((5-\varepsilon)^{\frac{n}{\beta}\log_5(2^\beta)}(5-\varepsilon)^{\frac{n}{\beta}}\right) \\
\leq\ & \mathcal{O}^*\left(2^{\delta_1 \beta \frac{n}{\beta}} 2^{\delta_2 n}\right) && \leq \mathcal{O}^*\left(2^{(\delta_1+\delta_2)n}\right) && \leq \mathcal{O}^*\left(2^{\delta n}\right),
\end{aligned}$$

由此完成证明。 ◀

6 结论与开放问题

我们首次在强指数时间假设下，为以团宽为参数的连通性问题（即连通顶点覆盖和连通支配集）得到了紧确结果。对于斯坦纳树、连通奇环横截集和反馈顶点集等若干重要的基准问题，凭借当前技术，我们尚无法得到紧确结果。对于斯坦纳树，我们的算法技术很容易得到一个运行时间为$\mathcal{O}^*(4^{\mathrm{cw}})$的算法，但下界所用的兼容性矩阵只包含一个大小为 3×3的三角子矩阵，因此我们无法证明比树宽情形更大的下界。类似地，对于连通奇环横截集，连通顶点覆盖所用的技术可以得到一个运行时间为$\mathcal{O}^*(14^{\mathrm{cw}})$的算法；通过改造连通顶点覆盖的构件并添加一个用于检测连接顶点处所用颜色的构件，可以证明更大的下界，但仍然没有足够大的三角子矩阵，无法证明$\mathcal{O}^*(14^{\mathrm{cw}})$是最优的。对于反馈顶点集这一带有无环性形式的负连通性约束的问题，常用的割计数方法需要统计由部分解诱导出的边数，但正如Bergougnoux和Kanté [4, 5]已指出的那样，这会立即导致一个以团宽为参数的XP算法。因此，要获得合理的运行时间并证明紧确结果，就需要采用不同的方法。

除了基准问题之外，要理解低团宽对问题复杂度的影响，研究大型问题类别也至关重要。目前，唯一此类精细化结果是Ganian 等人 [24]对同态问题的研究。最近，Focke 等人 [20] 证明了以树宽为参数的 (σ, ρ)- 支配集问题中一个大子类的紧确复杂度结果。根据 Bui-Xuan 等人 [11],的结果，我们知道这些问题也可以在以团宽为参数时用单指数时间求解。然而，除独立集和支配集等少数例外，以团宽为参数时的最优运行时间尚不明确。因此，若能获得类似于Focke 等人 [20] 的、以团宽为参数化对象的结果，将会很有意义。

应用以团宽为参数的算法时，一个重要的限制是，我们缺少计算团表达式的优良算法。目前最好的算法依赖于通过秩宽近似团宽，参见 Oum 和 Seymour [42] 提出的首个此类算法，以及 Fomin 和 Korhonen [21] 提出的最新算法。然而，通过秩宽进行近似会引入指数级误差，因此，除非我们通过其他方式获得团表达式，否则所有以团宽为参数的单指数算法都会变成双指数算法。改进团宽近似算法的第一步，或许是设计一种带有亚指数误差的固定参数可解算法。

参考文献

1 Josh Alman 和 Virginia Vassilevska Williams。改进的激光法与更快的矩阵乘法。载于 Dániel Marx（编），《2021年 ACM-SIAM 离散算法研讨会论文集》，SODA 2021，线上会议，2021年1月10日至13日，第522–539页。SIAM，2021。 doi:10.1137/1.9781611976465.32。2 Benjamin Bergougnoux。矩阵分解及其在（超）图中的算法应用。博士论文，克莱蒙奥弗涅大学，法国克莱蒙费朗，2019。URL： https://tel.archives-ouvertes.fr/tel-02388683。3 Benjamin Bergougnoux、Jan Dreier 和 Lars Jaffke。针对 mim-宽的基于逻辑的算法元定理，第3282–3304页。URL： https://epubs.siam.org/doi/abs/10.1137/1.9781611977554.ch125, arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9781611977554.ch125, doi:10.1137/1.9781611977554.ch125。4 Benjamin Bergougnoux 和 Mamadou Moustapha Kanté。若干由团宽参数化的连通性问题的快速精确算法。《理论计算机科学》，782：30–53，2019。doi:10.1016/j.tcs.2019.02.030。5 Benjamin Bergougnoux 和 Mamadou Moustapha Kanté。d-邻居等价的更多应用：无环性与连通性约束。《SIAM 离散数学杂志》，35(3)：1881–1926，2021。 doi:10.1137/20M1350571。6 Andreas Björklund、Thore Husfeldt、Petteri Kaski 和 Mikko Koivisto。针对有界团宽图的哈密顿回路最优 XP 算法。Algorithmica，82(6)：1654–1674，2020。 doi:10.1007/s00453-019-00663-9。7 Andreas Björklund、Thore Husfeldt、Petteri Kaski 和 Mikko Koivisto。截断莫比乌斯反演与有界度图。《理论计算机系统》，47(3)：637–654，2010。 doi:10.1007/s00224-009-9185-7。8 Andreas Björklund、Thore Husfeldt、Petteri Kaski、Mikko Koivisto、Jesper Nederlof 和 Pekka Parviainen。针对不可约元较少的格的快速泽塔变换。《美国计算机协会算法汇刊》，12(1)：4:1–4:19，2016。 doi:10.1145/2629429。9 Hans L. Bodlaender、Marek Cygan、Stefan Kratsch 和 Jesper Nederlof。针对由树宽参数化的连通性问题的确定性单指数时间算法。《信息与计算》，243：86–111，2015。 doi:10.1016/j.ic.2014.12.008。

48 以团宽为参数的连通性问题的紧致算法

10 Narek Bojikian、Vera Chekan、Falko Hegerfeld 和 Stefan Kratsch。针对由割宽参数化的连通性问题的紧确界。载于第40届国际计算机科学理论方面研讨会 STACS 2023，德国汉堡，2023年3月7日至9日。即将发表。11 Binh-Minh Bui-Xuan、Jan Arne Telle 和 Martin Vatshelle。针对局部可检验的顶点子集问题和顶点划分问题的快速动态规划。《理论计算机科学》，511：66–76，2013。 doi:10.1016/j.tcs.2013.01.009。12 Chris Calabro、Russell Impagliazzo 和 Ramamohan Paturi。小深度电路可满足性的复杂度。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，修订精选论文，计算机科学讲义第5917卷，第75–85页。施普林格，2009。 doi:10.1007/978-3-642-11269-0_6。13 Bruno Courcelle 和 Stephan Olariu。图的团宽上界。《离散应用数学》，101(1-3)：77–114，2000。 doi:10.1016/S0166-218X(99)00184-5。14 Radu Curticapean、Nathan Lindzey 和 Jesper Nederlof。通过矩阵秩计算哈密顿回路的紧确下界。载于 Artur Czumaj（编），《第二十九届 ACM-SIAM 离散算法研讨会论文集》，SODA 2018，美国路易斯安那州新奥尔良，2018年1月7日至10日，第1080–1099页。SIAM，2018。 doi:10.1137/1.9781611975031.70。15 Marek Cygan、Fedor V. Fomin、Lukasz Kowalik、Daniel Lokshtanov、Dániel Marx、Marcin Pilipczuk、Michał Pilipczuk 和 Saket Saurabh。《参数化算法》。施普林格，2015。doi:10.1007/978-3-319-21275-3。16 Marek Cygan、Stefan Kratsch 和 Jesper Nederlof。通过完美匹配基快速判定哈密顿性。《美国计算机协会会刊》，65(3)：12:1–12:46，2018。 doi:10.1145/3148227。17 Marek Cygan、Jesper Nederlof、Marcin Pilipczuk、Michał Pilipczuk、Johan M. M. van Rooij 和 Jakub Onufry Wojtaszczyk。在线性指数时间内求解由树宽参数化的连通性问题。载于 Rafail Ostrovsky（编），IEEE 第52届年度计算机科学基础研讨会，FOCS 2011，美国加利福尼亚州棕榈泉，2011年10月22日至25日，第150–159页。IEEE计算机学会，2011。 doi:10.1109/FOCS.2011.23。18 Marek Cygan、Jesper Nederlof、Marcin Pilipczuk、Michal Pilipczuk、Johan M. M. van Rooij 和 Jakub Onufry Wojtaszczyk。在线性指数时间内求解由树宽参数化的连通性问题。CoRR，abs/1103.0534，2011。URL： http://arxiv.org/abs/1103.0534，arXiv:1103.0534。19 Marek Cygan、Jesper Nederlof、Marcin Pilipczuk、Michal Pilipczuk、Johan M. M. van Rooij 和 Jakub Onufry Wojtaszczyk。在线性指数时间内求解由树宽参数化的连通性问题。《美国计算机协会算法汇刊》，18(2)：17:1–17:31，2022。 doi:10.1145/3506707。20 Jacob Focke、Dániel Marx、Fionn Mc Inerney、Daniel Neuen、Govind S Sankar、Philipp Schepper 和 Philip Wellnitz。有界树宽图中广义支配集计数的精细复杂度界。载于《2023年 ACM-SIAM 离散算法研讨会论文集》（SODA），第3664–3683页。SIAM，2023。21 Fedor V. Fomin 和 Tuukka Korhonen。分支宽的快速 FPT 近似。载于 Stefano Leonardi 和 Anupam Gupta（编），STOC '22：第54届年度 ACM SIGACT计算理论研讨会，意大利罗马，2022年6月20日至24日，第886–899页。美国计算机协会，2022。doi:10.1145/3519935.3519996。22 Fedor V. Fomin、Daniel Lokshtanov、Fahad Panolan 和 Saket Saurabh。代表族的高效计算及其在参数化算法和精确算法中的应用。《美国计算机协会会刊》，63(4)：29:1–29:60，2016。 doi:10.1145/2886094。23 Fedor V. Fomin、Daniel Lokshtanov、Fahad Panolan 和 Saket Saurabh。乘积族的代表族。《美国计算机协会算法汇刊》，13(3)：36:1–36:29，2017。 doi:10.1145/3039243。24 Robert Ganian、Thekla Hamm、Viktoriia Korchemna、Karolina Okrasa 和 Kirill Simonov。由团宽参数化的图同态的精细复杂度。载于 Mikolaj Bojanczyk、Emanuela Merelli 和 David P. Woodruff（编），第49届自动机、语言与编程国际研讨会，ICALP 2022，2022年7月4日至8日，法国巴黎，

LIPIcs第229卷，第66:1–66:20页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2022。 doi:10.4230/LIPIcs.ICALP.2022.66。25 Carla Groenland、Isja Mannens、Jesper Nederlof和Krisztina Szilágyi。按割宽参数化的计数着色与连通边集问题的紧致界。载于Petra Berenbrink和Benjamin Monmege（编），第39届国际计算机科学理论方面研讨会（STACS 2022），2022年3月15日至18日，法国马赛（线上会议），LIPIcs第219卷，第36:1–36:20页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2022。 doi:10.4230/LIPIcs.STACS.2022.36。26 Falko Hegerfeld和Stefan Kratsch。以单指数时间和多项式空间求解按树深度参数化的连通性问题。载于Christophe Paul和Markus Bläser（编），第37届国际计算机科学理论方面研讨会（STACS 2020），2020年3月10日至13日，法国蒙彼利埃，LIPIcs第154卷，第29:1–29:16页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2020。 doi:10.4230/LIPIcs.STACS.2020.29。27 Falko Hegerfeld和Stefan Kratsch。迈向参数化复杂性的精确结构阈值。载于Holger Dell和Jesper Nederlof（编），第17届参数化与精确计算国际研讨会（IPEC 2022），2022年9月7日至9日，德国波茨坦，LIPIcs第249卷，第17:1–17:20页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2022。 doi:10.4230/LIPIcs.IPEC.2022.17。28 Falko Hegerfeld和Stefan Kratsch。按模树宽参数化的连通性问题的紧致算法。预印本，即将发表，2023。29 Russell Impagliazzo和Ramamohan Paturi。论k-SAT的复杂性。《计算机与系统科学杂志》，62(2):367–375，2001。 doi:10.1006/jcss.2000.1727。30 Russell Impagliazzo、Ramamohan Paturi和Francis Zane。哪些问题具有强指数复杂度？《计算机与系统科学杂志》，63(4):512–530，2001。 doi:10.1006/jcss.2001.1774。31 Yoichi Iwata和Yuichi Yoshida。有界宽度问题之间的等价性。载于Nikhil Bansal和Irene Finocchi（编），算法——ESA 2015——第23届欧洲年度研讨会，希腊帕特雷，2015年9月14日至16日，论文集，计算机科学讲义第9294卷，第754–765页。施普林格，2015。 doi:10.1007/978-3-662-48350-3_63。32 Hugo Jacob、Thomas Bellitto、Oscar Defrain和Marcin Pilipczuk。近亲（反馈顶点集），再探。载于Petr A. Golovach和Meirav Zehavi（编），第16届参数化与精确计算国际研讨会（IPEC 2021），2021年9月8日至10日，葡萄牙里斯本，LIPIcs第214卷，第21:1–21:15页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2021。 doi:10.4230/LIPIcs.IPEC.2021.21。33 Bart M. P. Jansen和Jesper Nederlof。通过图分解和矩阵秩计算色数。《理论计算机科学》，795:520–539，2019。 doi:10.1016/j.tcs.2019.08.006。34 Ioannis Katsikarelis、Michael Lampis和Vangelis Th. Paschos。(k, r)-中心的结构参数、紧致界与近似。《离散应用数学》，264:90–117，2019。doi:10.1016/j.dam.2018.11.002。35 Ton Kloks。《树宽、计算与近似》，计算机科学讲义第842卷。施普林格，1994。 doi:10.1007/BFb0045375。36 Michael Lampis。团宽上着色问题的更精细紧致界。《SIAM离散数学杂志》，34(3):1538–1558，2020。 doi:10.1137/19M1280326。37 Daniel Lokshtanov、Dániel Marx和Saket Saurabh。有界树宽图上的已知算法很可能已是最优的。《美国计算机协会算法汇刊》，14(2):13:1–13:30，2018。doi:10.1145/3170442。38 Ketan Mulmuley、Umesh V. Vazirani和Vijay V. Vazirani。匹配问题与矩阵求逆一样容易。Combinatorica，7(1):105–113，1987。 doi:10.1007/BF02579206。39 Jesper Nederlof。通过矩阵的秩相关参数求解NP难问题的算法。载于Fedor V. Fomin、Stefan Kratsch和Erik Jan van Leeuwen（编），《树宽、核与算法——献给Hans L. Bodlaender的论文集，纪念其60

岁生日》，计算机科学讲义第12160卷，第145–164页。施普林格，2020。 doi:10.1007/978-3-030-42071-0_11。40 Jesper Nederlof、Michal Pilipczuk、Céline M. F. Swennenhuis和Karol Wegrzycki。按树深度参数化的哈密顿环问题可在单指数时间和多项式空间内求解。载于Isolde Adler和Haiko Müller（编），《计算机科学中的图论概念——第46届国际研讨会》，WG 2020，英国利兹，2020年6月24日至26日，修订精选论文，计算机科学讲义第12301卷，第27–39页。施普林格，2020。doi:10.1007/978-3-030-60440-0_3。41 Jesper Nederlof、Johan M. M. van Rooij和Thomas C. van Dijk。容斥原理与测度与征服相结合。Algorithmica，69(3):685–740，2014。 doi:10.1007/s00453-013-9759-2。42 Sang-il Oum和Paul D. Seymour。近似团宽和分支宽度。组合理论杂志B辑，96(4):514–528，2006。 doi:10.1016/j.jctb.2005.10.006。43 Michal Pilipczuk和Marcin Wrochna。处理图结构分解的算法的空间效率。《ACM计算理论汇刊》，9(4):18:1–18:36，2018。 doi:10.1145/3154856。44 Willem J. A. Pino、Hans L. Bodlaender和Johan M. M. van Rooij。分支分解上的切割与计数及代表集。载于Jiong Guo和Danny Hermelin（编），第11届参数化与精确计算国际研讨会（IPEC 2016），2016年8月24日至26日，丹麦奥胡斯，LIPIcs第63卷，第27:1–27:12页。达格施图尔城堡—莱布尼茨信息学中心，2016。 doi:10.4230/LIPIcs.IPEC.2016.27。45 Johan M. M. van Rooij、Hans L. Bodlaender和Peter Rossmanith。使用广义快速子集卷积在树分解上进行动态规划。载于Amos Fiat和Peter Sanders（编），算法——ESA 2009，第17届欧洲年度研讨会，丹麦哥本哈根，2009年9月7日至9日，论文集，计算机科学讲义第5757卷，第566–577页。施普林格，2009。 doi:10.1007/978-3-642-04128-0_51。

A 快速卷积算法

A.1 修剪子集卷积

本节将介绍快速计算连通顶点覆盖算法中并集节点处出现的覆盖乘积 所需的细节。连通顶点覆盖算法的状态为**状态** $= \mathcal{P}(\Omega) \setminus \{\emptyset, \Omega\}$，其中 $\Omega = \{\mathbf{0}, \mathbf{1}_L, \mathbf{1}_R\}$。本质上，给定两个表 A, B: **状态**$^{[k]} \to \mathbb{Z}_2$，我们希望在时间$\mathcal{O}^*(6^k) = \mathcal{O}^*(|\textbf{状态}|^k)$内计算以下覆盖乘积 $A \otimes_c B$: **状态**$^{[k]} \to \mathbb{Z}_2$：

$$(A \otimes_c B)(f) = \sum_{\substack{f_1, f_2 \in \mathbf{States}^{[k]}: \\ f_1 \cup f_2 = f}} A(f_1) B(f_2),$$

其中并集是逐分量计算的。若不排除 $\emptyset$ 和 Ω，直接应用标准的快速 Zeta 变换和莫比乌斯变换 [15] 便足够了。然而，这样得到的运行时间仅为$\mathcal{O}^*(8^k)$。为获得更优的运行时间，我们对上述快速 Zeta 变换和莫比乌斯变换的计算进行上下修剪。按照 Ω 的子集大小排序，并依照此顺序执行计算，我们便可以仅从相关子集开始，即排除 $\emptyset$，并在到达 Ω 本身之前停止计算。Björklund 等人 [7], 采用了所有这些思路，但他们的表述不适用于我们的情形，因此我们给出一种适用且自成一体的表述。

我们首先改用一种更标准的记法，并考虑定义在集合族 $\mathcal{F} \subseteq \mathcal{P}(U)$ 和有限全集 U 上的表，而不是定义在从 $[k]$ 到 **状态** 的函数上。给定函数 $f\colon [k] \to$ **状态**，将其转换为集合 $S_f = \{(i, \mathbf{s}) : i \in [k], \mathbf{s} \in f(i)\} = \bigcup_{i \in [k]} \{i\} \times f(i) \subseteq \bigcup_{i \in [k]} \{i\} \times \Omega =: U$，由此得到的集合族为

$\mathcal{F} = \{S \subseteq U : 1 \leq |S \cap (\{i\} \times \Omega)| \leq 2 \; \forall i \in [k]\}$。我们将看到，这个集合族可以写成一种特殊形式，从而能够为 Zeta 变换和莫比乌斯变换提供经过修剪的算法。

现在，我们回到一般情形。设 U 是某个固定的全集，并设 $\mathrm{F} \subseteq \mathrm{P}(U)$ 是 U 上的某个集合族。F 的上闭包 $\uparrow(\mathrm{F}) \subseteq \mathrm{P}(U)$ 定义为 $\uparrow(\mathrm{F}) = \{S \subseteq U : \text{存在一个 } T \in \mathrm{F}, \text{使得 } T \subseteq S\}$；当且仅当 $\mathrm{F} = \uparrow(\mathrm{F})$ 时，集合族 F 对超集封闭。若存在集合族 F_+ 和 F_-，使得 $\mathrm{F} = \uparrow(\mathrm{F}_+) \setminus \uparrow(\mathrm{F}_-)$，则称集合族 F 为闭包差。注意，$\uparrow(\emptyset) = \emptyset$，因此 $\uparrow(\mathrm{F}_+)$ 也是闭包差。闭包差恰好是不含空洞的集合族，其含义见下述引理。该引理对于证明各种变换的性质至关重要。

▶ **引理 A.1.** 集合族 $F \subseteq P(U)$ 是闭包差，当且仅当它满足以下区间性质：对于所有满足 $W \subseteq T \subseteq S$ 的集合 W、T、$S \subseteq U$，若 W、$S \in F$，则还必须有 $T \in F$。

证明。我们首先说明，当 F 是一个闭包差时，区间性质成立。因此，设 $\mathrm{F} = \uparrow(\mathrm{F}_+) \setminus \uparrow(\mathrm{F}_-)$，且 $W \subseteq T \subseteq S$ 满足 $W \in \mathrm{F}$ 和 $S \in \mathrm{F}$。存在一个集合 $X \in \mathrm{F}_+$ 满足 $X \subseteq W$，因为 $W \in \uparrow(\mathrm{F}_+)$；并且对于所有集合 $Y \in \mathrm{F}_-$，都有 $Y \not\subseteq S$，因为 $S \notin \uparrow(\mathrm{F}_-)$。因此，由于 $X \subseteq W \subseteq T$，我们有 $T \in \uparrow(\mathrm{F}_+)$，而不存在满足 $Y \subseteq T$ 的 $Y \in \mathrm{F}_-$，否则我们也会有 $Y \subseteq S$。因此，$T \in \uparrow(\mathrm{F}_+) \setminus \uparrow(\mathrm{F}_-) = \mathrm{F}$。

对于反方向，我们首先说明 $\uparrow(\mathrm{F}) \setminus \mathrm{F}$ 是一个上闭包，即对取超集封闭。反设 $T \in \uparrow(\mathrm{F}) \setminus \mathrm{F}$ 且 $T \subseteq S$，但 $S \notin \uparrow(\mathrm{F}) \setminus \mathrm{F}$。由于 $T \in \uparrow(\mathrm{F})$，存在某个 $W \in \mathrm{F}$ 满足 $W \subseteq T \subseteq S$，因此也有 $S \in \uparrow(\mathrm{F})$。由 $S \notin \uparrow(\mathrm{F}) \setminus \mathrm{F}$ 可知，进而有 $S \in \mathrm{F}$。最后，由于 F 满足区间性质，我们必有 $T \in \mathrm{F}$，从而与 $T \in \uparrow(\mathrm{F}) \setminus \mathrm{F}$ 矛盾。因此，$\uparrow(\mathrm{F}) \setminus \mathrm{F} = \uparrow(\mathrm{F}_-)$，其中某个族 F_-，于是得到 $\mathrm{F} = \uparrow(\mathrm{F}) \setminus (\uparrow(\mathrm{F}) \setminus \mathrm{F}) = \uparrow(\mathrm{F}) \setminus \uparrow(\mathrm{F}_-)$。◀

给定一个表 $A\colon \mathcal{F} \to R$，其中 R 是某个含幺交换环，*Zeta* 变换 $\zeta A\colon \mathcal{F} \to R$ 定义为

$$(\zeta A)(S) = \sum_{\substack{T \in \mathcal{F}: \\ T \subseteq S}} A(T),$$

莫比乌斯变换 $\mu A\colon \mathcal{F} \to R$ 定义为

$$(\mu A)(S) = \sum_{\substack{T \in \mathcal{F}: \\ T \subseteq S}} (-1)^{|S \setminus T|} A(T),$$

以及 奇反演变换 $\sigma A\colon \mathcal{F} \to R$ 定义为

$$(\sigma A)(S) = (-1)^{|S|} A(S).$$

所有这些变换都可以视为从 $\mathcal{F}$ 到 R 的函数空间上的算子。给定两个表 A，$B\colon \mathcal{F} \to R$，它们的覆盖乘积 $A \otimes_c B\colon \mathcal{F} \to R$ 定义为

$$(A \otimes_c B)(S) = \sum_{\substack{T_1, T_2 \in \mathcal{F}: \\ T_1 \cup T_2 = S}} A(T_1) B(T_2).$$

此外，我们用 $A \cdot B\colon \mathcal{F} \to R$ 表示两个表 A 和 B 的逐点乘法，即 $(A \cdot B)(S) = A(S)B(S)$。现在，我们将证明这些变换的若干性质，以便设计一种快速算法来计算覆盖乘积。

▶**引理 A.2.** 设 F 为一个闭包差，且 R 为含幺交换环。以下陈述成立：

1. $\sigma\zeta\sigma = \mu$， $\sigma\mu\sigma = \zeta$，2. $\mu\zeta = \zeta\mu = \mathrm{id}$，其中 id 是恒等变换，
3. $\zeta(A \otimes_c B) = (\zeta A) \cdot (\zeta B)$，对于任意两个表 A， B：
$\mathcal{F} \to R$。

证明。 我们调整 Cygan 等人 [15] 的证明以适用于我们的情形，并指出证明中 f
我们使用 $\mathcal{F}$ 是闭包差的步骤。
1. 对于每个表，我们计算 A： $\mathcal{F} \to R$ 并令 $S \in \mathcal{F}$:

$$\begin{aligned}(\sigma\zeta\sigma A)(S) &= (-1)^{|S|}\sum_{\substack{T\in\mathcal{F}:\\ T\subseteq S}}(-1)^{|T|}A(T) = \sum_{\substack{T\in\mathcal{F}:\\ T\subseteq S}}(-1)^{|S|+|T|}A(T) = \sum_{\substack{T\in\mathcal{F}:\\ T\subseteq S}}(-1)^{|S\setminus T|}A(T)\\ &= (\mu A)(S),\end{aligned}$$

其中第三个等式由 $|S|+|T| = 2|S\cap T| + |S\setminus T|$ 得出。由于 $\sigma\sigma = \mathrm{id}$，因此也可得 $\sigma\mu\sigma = \sigma\sigma\zeta\sigma\sigma = \zeta$。
2. 由前述结论可知 $\mu\zeta = \sigma\zeta\sigma\zeta$。同样，对于每个表 A，我们计算：$\mathcal{F} \to R$ 并令 $S \in \mathcal{F}$:

$$\begin{aligned}(\sigma\zeta\sigma\zeta A)(S) &= (-1)^{|S|}\sum_{\substack{T\in\mathcal{F}:\\ T\subseteq S}}(\sigma\zeta A)(T) = (-1)^{|S|}\sum_{\substack{T\in\mathcal{F}:\\ T\subseteq S}}(-1)^{|T|}\sum_{\substack{W\in\mathcal{F}:\\ W\subseteq T}}A(W)\\ &= (-1)^{|S|}\sum_{\substack{W\in\mathcal{F}:\\ W\subseteq S}}A(W)\sum_{\substack{T\in\mathcal{F}:\\ W\subseteq T\subseteq S}}(-1)^{|T|} = A(S) + (-1)^{|S|}\sum_{\substack{W\in\mathcal{F}:\\ W\subsetneq S}}A(W)\sum_{\substack{T\in\mathcal{F}:\\ W\subseteq T\subseteq S}}(-1)^{|T|}\\ &= A(S),\end{aligned}$$

其中，最后一个等式由以下事实得出：$\sum_{T\in\mathcal{F}:\ W\subseteq T\subseteq S}(-1)^{|T|} = 0$ 当 $W \subsetneq S$ 时成立，因为我们可以选取某个 $x \in S\setminus W$，并将每个 T 与 $T \triangle \{x\}$ 配对，从而得到一个无不动点的符号反转对合。注意，这一步依赖于引理 A.1（应用于 $W \subseteq T \triangle\{x\} \subseteq S$）。对于任意集合族 $\mathcal{F}'$, , $T \triangle\{x\} \in \mathcal{F}'$不一定成立。

其余等式由 $\zeta\mu = \zeta\sigma\zeta\sigma = \sigma\mu\sigma\sigma\zeta\sigma = \sigma\mu\zeta\sigma = \sigma\sigma = \mathrm{id}$ 得出。
3. 对于任意两个表 A、 B，我们计算：$\mathcal{F} \to R$ 并令 $S \in \mathcal{F}$:

$$\begin{aligned}(\zeta(A\otimes_c B))(S) &= \sum_{\substack{W\in\mathcal{F}:\\ W\subseteq S}}\ \sum_{\substack{T_1,T_2\in\mathcal{F}:\\ T_1\cup T_2 = W}} A(T_1)B(T_2) = \sum_{\substack{T_1,T_2\in\mathcal{F}:\\ T_1\cup T_2\subseteq S}} A(T_1)B(T_2)\\ &= \sum_{\substack{T_1\in\mathcal{F}:\\ T_1\subseteq S}}\ \sum_{\substack{T_2\in\mathcal{F}:\\ T_2\subseteq S}} A(T_1)B(T_2) = \left(\sum_{\substack{T\in\mathcal{F}:\\ T\subseteq S}} A(T)\right)\left(\sum_{\substack{T\in\mathcal{F}:\\ T\subseteq S}} B(T)\right)\\ &= ((\zeta A)\cdot(\zeta B))(S),\end{aligned}$$

其中，第二个等式再次依赖于引理 A.1（应用于 $T_1 \subseteq T_1 \cup T_2 \subseteq S$）。对于任意集合族 $\mathcal{F}'$, ，我们可能有 $T_1 \cup T_2 \notin \mathcal{F}'$ ，因此左侧不会对其求和。◀

借助引理 A.2 中各种变换之间的关系，我们现在可以着手算法部分。不过，我们需要集合族 $\mathcal{F}$ 满足一些算法要求。我们称$\mathcal{F} \subseteq \mathcal{P}(U)$为可高效列举的 ，如果存在一种算法，能够在$\mathcal{O}^*(|\mathcal{F}|) = |\mathcal{F}||U|^{\mathcal{O}(1)}$时间内输出$\mathcal{F}$ 的所有成员。$\mathcal{F}$中的每个成员

都表示为一个位串，并且我们可以假设枚举算法的输出按排序顺序排列。因此，列举$\mathcal{F}$之后，我们可以通过二分查找，在$\mathcal{O}(\log |\mathcal{F}|) = \mathcal{O}(|U|)$时间内判断集合 $S \subseteq U$ 是否属于 $S \in \mathcal{F}$。表 $A\colon \mathcal{F} \to R$由按与$\mathcal{F}$的列举相同顺序列出的值$A(S)$，$S \in \mathcal{F}$给出。

在以下运行时间分析中，我们将环运算的影响与算法的其余部分分开，也就是说，我们只统计执行的环运算次数，并将算法其余部分所耗费的时间计入上界。

▶**推论 A.3.** 设 F 是一个可高效列举的闭包差，R 是一个含幺交换环。如果表 $A\colon F \to R$ 的 *Zeta* 变换 ζA 和莫比乌斯变换 μA 可以在 $O(|F||U|)$ 次环运算及额外的 $O^*(|F|)$时间内计算出来，那么覆盖乘积 $A \otimes_c B$ 可以在 $O(|F||U|)$ 次环运算及额外的 $O^*(|F|)$时间内计算出来。

证明。我们使用引理 A.2。给定表 A 和 B，我们首先计算 ζA 和 ζB，然后计算逐点乘法 $(\zeta A) \cdot (\zeta B)$。根据定义，逐点乘法可以在 $O(|F|)$ 次环运算及额外的 $O^*(|F|)$ 时间内完成，后者用于列举 F 的所有元素。最后，我们计算逐点乘法的莫比乌斯变换。根据引理 A.2，有 $\mu((\zeta A) \cdot (\zeta B)) = \mu\zeta(A \otimes_c B) = A \otimes_c B$。根据假设，每一步都需要 $O(|F||U|)$ 次环运算及额外的 $O^*(|F|)$ 时间，因此结论成立。◀

▶ **定理 A.4.** 设 $F = \uparrow(F_+) \mid \uparrow(F_-)$ 是可高效列举的闭包差，且 R是含幺交换环。给定一个表 $A\colon F \to R$，*Zeta* 变换 ζA 和莫比乌斯变换 μA 可通过 $O(|F||U|)$ 次环运算以及额外的 $O^*(|F|)$ 时间计算得出。

证明。 我们对 Björklund 等人的算法 [7, 算法 Z] 进行修改，使其适用于我们的情形。设 $n = |U|$ 为全集的大小，并且不失一般性地假设 $U = [n]$。该算法维护 $n+1$ 个集合族 $\mathcal{L}_0, \ldots, \mathcal{L}_n \subseteq \mathcal{F}$，其中每个 $\mathcal{L}_i$ 仅包含大小为 i的集合。对于所有 $j \in [0, n]$ 和 $S \in \mathcal{F}$，我们将计算辅助值

$$A_j(S) = \sum_{\substack{T \in \mathcal{F}\colon T \cap [j] \subseteq S \cap [j], \\ T \cap [j+1,n] = S \cap [j+1,n]}} A(T).$$

显然，$A_0(S) = A(S)$ 且 $A_n(S) = \zeta A(S)$。此外，可以验证递推式$A_j(S) = [(j \in S) \wedge (S \setminus \{j\} \in \mathrm{F})]A_{j-1}(S \setminus \{j\}) + A_{j-1}(S)$ 对所有 $j \in [n]$ 和 $S \in \mathrm{F}$ 均成立。我们特别指出 $j \in S$ 但 $S \setminus \{j\} \notin \mathrm{F}$ 的情形；此时，每个满足 $T \subseteq S$ 的 $T \in \mathrm{F}$ 都必须满足$j \in T$，否则根据引理 A.1，我们将得到 $S \setminus \{j\} \in \mathrm{F}$，因此 $A_j(S) = A_{j-1}(S)$。

算法 3 快速泽塔变换

```
1 对于 S ∈F，插入 S 到 L^|S|；
2 对于 r = 0, 1, ..., n，执行
3 当 L^r 非空时，执行
4 选择任意 S ∈L^r 并将其从 L^r中移除；
5 A_0(S) := A(S)；
6 对于 j = 1, ..., n，执行
7 A_j(S) := [(j ∈ S) ∧ (S \ {j} ∈ F)]A_{j-1}(S \ {j}) + A_{j-1}(S);
8 返回 A_n;
```

54 连通性问题的紧致算法（参数化于团宽）

显然，算法 3 恰好考虑每个集合 $S \in \mathcal{F}$ 一次，并且由于按基数排序，只会访问已计算出的值。根据前面的讨论，算法 3 能正确计算 Zeta 变换 $\zeta A = A_n$。环运算只在线 7 中执行，即每次执行线 7 最多进行两次。对于$\mathcal{F}$中的每个元素，线 7 恰好执行 n 次，因此算法 3 总共执行 $\mathcal{O}(|\mathcal{F}|n)$ 次环运算。额外时间由线 1 中列出 $\mathcal{F}$ 的操作以及线 7 引发的$\mathcal{O}(|\mathcal{F}|n)$ 次成员查询所支配，因此额外时间为 $\mathcal{O}^*(|\mathcal{F}|)$。

为计算莫比乌斯变换 μA，我们使用引理 A.2 中的 $\mu = \sigma\zeta\sigma$。因此，我们先计算 σA，再应用算法 3 对 σA 进行处理，以得到 $\zeta\sigma A$，最后再次应用 σ。由于当 k 为偶数时，$(-1)^k = 1$；当 k 为奇数时，$(-1)^k = -1$，我们可以在 $\mathcal{O}(|\mathcal{F}|)$ 次环运算内应用 σ，此外，由于需要列举，还需耗时 $\mathcal{O}^*(|\mathcal{F}|)$，因此结论得证。◀

逐分量并转化为集合并。固定一个自然数 k 和全集 U。我们定义投影 $\pi_U\colon [k] \times U \to U$，$(i, u) \mapsto u$；对于任意 $i \in [k]$和集合 $S \subseteq U$，定义$\pi^i_U(S) := \pi_U(S \cap (\{i\} \times U))$。给定集合族 $\mathbf{F} \subseteq \mathbf{P}(U)$，我们构造集合族$k\mathbf{F} := \{S \subseteq [k] \times U\colon \pi^i_U(S) \in \mathbf{F}$，对所有 $i \in [k]$均成立$\}$。给定函数 $f\colon [k] \to \mathbf{F}$，我们构造集合 $k\mathbf{F} \ni S_f := \bigcup_{i\in[k]} \{i\} \times f(i) = \{(i, x) \in [k] \times U\colon x \in f(i)\}$。这是一个双射，并且该构造将函数 $[k] \to \mathbf{F}$ 的逐分量并转化为集合 $k\mathbf{F}$ 中的集合并；也就是说，对于函数 $f, f_1, f_2\colon [k] \to \mathbf{F}$，当且仅当 $S_f = S_{f_1} \cup S_{f_2}$ 时，对所有 $i \in [k]$，均有 $f(i) = f_1(i) \cup f_2(i)$。因此，给定两个表 $A, B\colon \mathbf{F}^{[k]} \to R$，逐分量覆盖乘积$A \otimes_{comp} B\colon \mathbf{F}^{[k]} \to R$ 可化简为标准覆盖乘积，即

$$(A \otimes_{comp} B)(f) := \sum_{\substack{f_1, f_2 \in \mathcal{F}^{[k]}\colon \\ f_1(i) \cup f_2(i) = f(i)\ \forall i \in [k]}} A(f_1)B(f_2) = \sum_{\substack{S_{f_1}, S_{f_2} \in k\mathcal{F}\colon \\ S_{f_1} \cup S_{f_2} = S_f}} A(f_1)B(f_2) = (A' \otimes_c B')(S_f),$$

其中 $A', B'\colon\ k\mathcal{F} \to R$满足 $A'(S_f) := A(f)$ 且 $B'(S_f) := B(f)$，由于 $f \mapsto S_f$ 是双射，因此定义良好。此外，下面的引理表明，如果 $\mathcal{F}$ 是闭包差，我们就可以将快速卷积算法应用于 $k\mathcal{F}$。

▶ **引理 A.5.** 设 $F \subseteq P(U)$ 是一个集合族，且 k 是某个自然数。若 F 是闭包差，则集合族 $kF \subseteq P([k] \times U)$ 也是闭包差。

证明。我们首先论证，对于所有集合族 $\mathbf{F}$ 和自然数 k，都有 $k{\uparrow}(\mathbf{F}) = {\uparrow}(k\mathbf{F})$。当且仅当存在 $T_i \in \mathbf{F}$ 使得对于所有 $i \in [k]$，$\{i\} \times T_i \subseteq S \cap (\{i\} \times U)$ 时，$S \in k{\uparrow}(\mathbf{F})$ 成立。令 $T = \bigcup_{i\in[k]} \{i\} \times T_i \in k\mathbf{F}$，可知这等价于存在某个 $T \subseteq S$，且当且仅当 $S \in {\uparrow}(k\mathbf{F})$ 时该条件成立。

令 $\mathcal{F} = {\uparrow}(\mathcal{F}_+) \setminus {\uparrow}(\mathcal{F}_-)$ 为一个闭包差。我们计算

$k\mathbf{F} = \{S \subseteq [k] \times U\colon \pi^i_U(S) \in \mathbf{F}$ 对所有 $i \in [k]\} = \{S \subseteq [k] \times U\colon \pi^i_U(S)$
$\in ({\uparrow}(\mathbf{F}_+) \setminus {\uparrow}(\mathbf{F}_-))$ 对所有 $i \in [k]\} = k{\uparrow}(\mathbf{F}_+) \setminus \{S \subseteq [k] \times U$: 存在一个
$i \in [k]$ 使得 $\pi^i_U(S) \in {\uparrow}(\mathbf{F}_-)\} = {\uparrow}(k\mathbf{F}_+) \setminus \{S \subseteq [k] \times U$: 存在一个
$i \in [k]$ 使得 $\pi^i_U(S) \in {\uparrow}(\mathbf{F}_-)\}$

并注意到，最后一行中的第二个集合显然在取超集的运算下封闭，因此 $k\mathcal{F}$ 也同样是一个闭包差。◀

▶ **定理 A.6。** 设 $F \subseteq P(U)$ 是一个固定的可高效列举的闭包差，且 R 是一个含幺交换环。给定表 A、$B\colon\ F^{[k]} \to R$，其逐分量覆盖乘积$A \otimes_{comp} B$ 可以通过 $O(k|F|^k|U|)$ 次环运算以及额外的 $O^*(|F|^k)$时间计算得到。

证明。我们构造集合族 $k\mathbb{F} \subseteq \mathrm{P}([k] \times U)$，根据引理 A.5，它是一个闭包差。由于 F 是可高效列举的，因此 $k\mathbb{F}$ 也可以按字典序高效列举。我们应用定理 A.4 和推论 A.3，利用 $k\mathbb{F}$ 在 $\mathrm{O}(k|\mathbb{F}|^k|U|)$ 次环运算以及额外的 $\mathrm{O}^*(|\mathbb{F}|^k)$ 时间内计算 $k\mathbb{F}$ 上的覆盖乘积。根据前面的讨论，$k\mathbb{F}$ 上的覆盖乘积给出了 F 上的逐分量覆盖乘积 $A \otimes_{comp} B$。◀

A.2 基于格的卷积

偏序集是一个二元组（L，$\preceq$），由集合 L 和其上的二元关系 $\preceq$ 组成，该关系具有自反性、传递性和反对称性。格是一个偏序集（L，$\preceq$），其中任意一对 a、$b \in$ L 都有一个最大下界（交）$a \wedge b \in$ L 和一个最小上界（连接）$a \vee b \in$ L。任意有限格（L，$\preceq$）都包含一个 $\preceq$-最小元素 $\hat{0} \in$ L，它由对 L 中所有元素取交得到，并且对所有 $a \in$ L 都满足 $\hat{0} \vee a = a \vee \hat{0} = a$。

在连通支配集算法中，可能状态的集合构成一个格；而在团表达式的并节点处，我们必须计算一个卷积类乘积。为了得到高效算法，我们将考察一些格论性质。

我们感兴趣的乘积可以在格的框架下表述如下。给定一个格（$\mathcal{L}$ $\preceq$）和表 A，$B\colon \mathcal{L} \to \mathbb{F}$，其中 $\mathbb{F}$ 是某个域，则 $\vee$-乘积 $A \otimes_{\mathcal{L}} B\colon \mathcal{L} \to \mathbb{F}$ 定义为 $(A \otimes_{\mathcal{L}} B)(x) = \sum_{y,\ z \in \mathcal{L}\colon\ x = y \vee z} A(y)B(z)$，对每个 $x \in \mathcal{L}$ 均成立。

Björklund 等人 [8] 通过为 Zeta 变换和莫比乌斯变换设计规模较小的算术电路，为特定格上的 $\vee$-乘积开发了一种高效算法；此处不需要给出这两种变换的精确定义。相关概念如下：若格（$\mathcal{L}$ $\preceq$）中的一个元素 $x \in \mathcal{L}$ 满足：对所有 a，$b \in \mathcal{L}$，$x = a \vee b$ 蕴含 $x = a$ 或 $x = b$，则称其为并不可约；否则 x 称为并可约。我们用 $\mathcal{L}_\vee$ 表示 $\mathcal{L}$ 中并不可约元的集合。注意，$\hat{0}$ 总是并不可约的，否则 $\hat{0}$ 就不是 $\preceq$-最小元。我们甚至有更强的性质：$\hat{0} = a \vee b$ 蕴含 $a = b = \hat{0}$。

我们假设有限格（$\mathcal{L}$ $\preceq$）以<此处应为标签原文>方式以算法形式给出；我们得到集合 $\mathcal{L}$，其中 $\mathcal{L}$ 的元素表示为 $\mathcal{O}(\log|\mathcal{L}|)$ 位字符串、并不可约元集合 $\mathcal{L}_\vee \subseteq \mathcal{L}$，以及一个算法 $A_{\mathcal{L}}$，它在给定元素 $a \in \mathcal{L}$ 和并不可约元 $x \in \mathcal{L}_\vee$ 时计算连接 $a \vee x$。

▶ **定理 A.7** ([8])。设 $(L \preceq)$ 是以并表示给出的有限格，且 A、$B\colon L \to \mathbb{F}$ 是两个表，其中 $\mathbb{F}$ 是某个域。$\vee$-积 $A \otimes_{\mathcal{L}} B$ 可以通过 $O(|L||L_\vee|)$ 次域运算以及对算法 $A_{\mathcal{L}}$ 的调用计算，另需 $O(|L||L_\vee|^2)$ 时间。

接下来，我们分析连通支配集算法中出现的格，并推导该格中并不可约元数量的上界，以便应用定理 A.7。相关的格可以表示为一个较小格的幂，我们将给出这类格的一般上界。下面介绍相关定义。

给定有限多个格 $(\mathcal{L}_i, \preceq_i)$，$i \in I$，它们的直积 $(\prod_{i \in I} \mathcal{L}_i, \preceq)$，其中 $(a_i)_{i \in I} \preceq (b_i)_{i \in I}$ 当且仅当 $a_i \preceq_i b_i$ 对所有 $i \in I$ 成立，也是一个格；直积格中的连接运算和交运算，是通过对各个组成格中的相应运算逐分量进行定义的。给定一个格（$\mathcal{L}$ $\preceq$）和一个自然数 k，次幂 $(\mathcal{L}^k, \preceq^k)$ 是 $\mathcal{L}$ 的 k 个副本的直积。

▶ **引理 A.8.** 设 $(L \preceq)$ 是一个有限格，k 是一个自然数。在 L 的 k 次幂 $(L^k, \preceq^k)$ 中，元素 $x \in L^k$ 是并不可约元，当且仅当 $x = (\hat{0}, \ldots, \hat{0})$，或者恰有一个分量不等于 $\hat{0}$，且该分量在 L 中是并不可约元。特别地，L^k 中恰有 $1 + k(|L_\vee| - 1) \leq k|L|$ 个并不可约元。

证明。首先，我们证明 L^k 的每个并不可约元 x 都必定具有上述形式。反设 $x=(x_1,\ldots,x_k)$ 至少有两个不等于 $\hat{0}$ 的分量；不失一般性，设 $x_1\neq\hat{0}$ 且 $x_2\neq\hat{0}$。我们计算得 $x=(x_1,\ldots,x_k)=(\hat{0},x_2,x_3,\ldots,x_k)\vee(x_1,\hat{0},x_3,\ldots,x_k)$，并注意到根据假设，$x\neq(\hat{0},x_2,x_3,\ldots,x_k)$ 和 $x\neq(x_1,\hat{0},x_3,\ldots,x_k)$。因此，在这种情况下，$x$ 是并可约的。此外，若 x 含有一个并可约分量，例如 $x_1=a_1\vee b_1$，其中 $a_1\neq x_1\neq b_1$，则 x 是并可约的，因为 $x=(a_1,x_2,\ldots,x_k)\vee(b_1,x_2,\ldots,x_k)$。

对于另一个方向，我们有 $x=(\hat{0},\ldots,\hat{0})$，它是 $\mathcal{L}^k$ 中的 $\preceq^k$ 最小元素，因此是并不可约的。现在，设 $x=(x_1,\ldots,x_k)\in\mathcal{L}^k$ 恰有一个分量不为 $\hat{0}$，且该分量是并不可约的，记为 $x_1\in\mathcal{L}_\vee\setminus\{\hat{0}\}$，并且对所有 $i\in[2,k]$，$x_i=\hat{0}$。假设 $x=y\vee z$，其中 $y,z\in\mathcal{L}^k$，那么对所有$i\in[2,k]$，我们都有 $\hat{0}=x_i=y_i\vee z_i$，这意味着对所有 $i\in[2,k]$，$x_i=y_i=z_i=\hat{0}$。对于 $i=1$，我们看到 $x_1=y_1\vee z_1$ 意味着 $x_1=y_1$ 或 $x_1=z_1$，这是由 x_1 的不可约性得出的，因此 $x=y$ 或 $x=z$。◀

▶ **推论 A.9.** 设 $(L\preceq)$ 是一个以并表示给出的有限格，且 k 是一个自然数。存在一个算法 $A_{\mathcal{L}^k}$，它只需调用 $A_{\mathcal{L}}$ 一次，并额外花费 $O(k\log|L|)$的时间，就能计算连接 $a\vee x$，其中 $a\in L^k$ 且 $x\in(L^k)_\vee$。

证明。L 中的每个元素都由一个 $O(\log|L|)$ 位的字符串表示，因此 L^k 中的每个元素都可以由一个 $O(k\log|L|)$ 位的字符串表示。根据引理 A.8，我们知道，并不可约元 $x\in L^k$ 至多有一个分量不等于 $\hat{0}$。我们用 $O(k\log|L|)$ 的时间查找这个分量及其在 a 中的对应分量，随后调用 $A_\vee$ 求得该分量的结果。由于对所有 $i\in[k]$都有 $a_i\vee\hat{0}=a_i$，a 的所有其他分量保持不变。◀

▶ **推论 A.10.** 设 $(L\preceq)$ 是一个以并表示给出的有限格，且 k 是一个自然数。给定两个表 A,B: $L^k\to\mathbb{Z}_2$，可在 $O(k^2|L|^{k+2})$ 时间内计算 L^k 中的 $\vee$-积 $A\otimes_{\mathcal{L}^k}B$，并调用算法 $A_{\mathcal{L}}$ $O(k|L|^{k+1})$ 次。

证明。我们将定理 A.7 与引理 A.8 和推论 A.9 流水线式结合。首先，注意到 $\mathbb{Z}_2$ 中的域运算可在常数时间内完成。其次，每次调用 $A_{\mathcal{L}^k}$都可以通过一次调用 $A_{\mathcal{L}}$ 以及额外的 $O(k\log|L|)$ 时间来模拟，因此这些调用总共需要 $O(|L^k||(L^k)_\vee|(k\log|L|))=O(k^2|L|^{k+1}\log|L|)$ 的额外运行时间，这将被算法的其余部分所支配。最后，定理 A.7 中的算法还需要 $O(|L^k||(L^k)_\vee|^2)=O(|L|^k(k|L|)^2)=O(k^2|L|^{k+2})$ 的额外时间，这支配了所有其他计算。◀