arXiv:2302.11619v1 [cs.DS] 22 Feb 2023

有序图中的模式检测

Guillaume Ducoffe[1][0000−0003−2127−5989]、
Laurent Feuilloley[2][0000−0002−3994−0898]、
Michel Habib[3][0000−0002−8564−2314]、Fran、
cois Pitois[2], 4[0000−0002−5524−0138]

[1] 布加勒斯特大学，罗马尼亚；国家信息学研究与发展研究所，罗马尼亚[2] 里昂大学、法国国家科学研究中心、INSALyon、里昂第一大学、LIRIS实验室、UMR5205，法国维勒班 F-69622[3] IRIF研究所、法国国家科学研究中心与巴黎西岱大学´e Paris Cit´e，法国[4] LIB实验室、勃艮第-弗朗什-孔泰大学´e Bourgogne Franche-Comt´e，法国

摘要。定义或刻画图类的一种常见方法是通过禁子图或禁子式。这些刻画在图论中发挥着关键作用，但很少能导出用于识别这些图类的高效算法。相比之下，许多重要的图类可以借助如下形式的刻画得到高效识别：必须存在一种顶点排序，使得某个有序 $pat\text{-}tern$ 不会出现，其中模式本质上是一种有序子图。数十年来，人们一直在研究这类模式刻画，近期也开始尝试更系统地理解它们。本文聚焦于这一主题核心的一个简单问题：给定一个大小为 n的有序图，检测其中是否存在一个大小为 k 的固定模式，需要多快?

沿着图类识别领域的研究思路，我们首先寻找能够在线性时间内检测的模式。我们证明了若干结果，其中包括：几乎所有包含三个顶点的模式（它们涵盖许多有趣的图类，如区间图、弦图、分裂图、二分图和可比图）都属于这一范畴。接着，从更细粒度的复杂度视角出发，我们证明了该问题的条件下界。特别是，我们表明，对于一大类包含四个顶点的模式，不太可能存在次二次算法。最后，我们为模式定义了一个参数——$merge\text{-}width$，并证明，对于合并宽度为 t 的模式，可以在 $O(n^{ct})$ 时间内解决该问题，其中 c 为某个常数。作为推论，我们得到，检测外平面模式及其他一些模式类所需的时间可以与模式的大小无关。

关键词：遗传图类、禁结构、模式刻画、图算法、细粒度复杂性、参数化算法、子图检测、合并宽度。

1 引言

1.1 动机

禁结构的刻画与识别。定义图类最常见的方法是禁止某些子结构，通常是（诱导）子图或图子式。这种方法带来了大量关于这些图的结构以及如何利用这些结构（例如用于组合优化）的研究成果。它也是一个灵活多用的框架，因为许多有用的图类都可以用禁子图或图子式来描述。例如，[15] 建立了一组禁诱导子图，用以刻画区间图这一著名图类，而该图类最初的定义并未涉及任何禁结构。

然而，在高效识别图类时，禁用子图就不那么实用了。首先，禁结构列表可能很长，甚至是无限的。例如，区间图的禁结构列表包含三个无限族。其次，高效检测诱导子图可能颇具挑战。

高效识别与禁用模式。识别区间图或弦图等图类的经典高效方法，不是使用禁用子图，而是采用某些特定的图遍历来揭示这些图的内在结构。例如，弦图可以通过两步法识别：首先执行一种称为 LexBFS 的特定广度优先搜索（BFS），然后检查顶点访问顺序的逆序是否具有这样的性质：对于任意顶点，其后继所构成的邻域是一个团。这些识别算法高效性的关键原则在于，许多图类都可以通过其顶点的一种排序来刻画，而且该排序具有可高效检验的性质。例如，一个图是区间图，当且仅当存在其顶点的一种排序，使得对于任意三个（不一定连续的）顶点 $u < v < w$，如果 (u, w) 是一条边，那么 (u, v) 也必定是一条边。对这种禁结构进行紧凑编码的一种方法，是使用所谓的模式 [9], ，见图 1 及其说明。

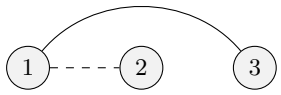


图 1：这幅图展示了与区间图相关的模式。如果存在三个（不一定连续的）顶点 $v_1 < v_2 < v_3$，使得 $(v_1, v_3) \in E$ 且 $(v_1, v_2) \notin E$，那么该模式就能在有序图中实现。换句话说，要实现这一模式，实线边（称为必选边）必须存在，虚线边（称为禁用边）必须不存在，而未绘出的边（称为未定对）则不受限制。当且仅当存在一种顶点排序，使得该模式无法实现时，图才是区间图。

许多图类都已有通过禁用模式进行刻画的结果，例如弦图、区间图、余可比图、平凡完美图等。在 [9] 中，作者系统地研究了所有这类

可由三个顶点上的禁用模式刻画的图类，并证明它们基本上都对应于众所周知的图类，而且几乎都能在线性时间内识别。

有序图中的模式检测。基于禁用模式刻画的典型识别算法通常包含两个组成部分：首先构造一个候选序，然后检查该模式是否未出现。乍看之下，第一部分似乎更具挑战性，也应该决定复杂度。事实上，对于一个有 n 个顶点的图，暴力生成所有顶点序需要 $O(n!)$，而对于一个有 k 个顶点的模式，暴力检查至多需要 $O(n^k)$。但是，正如前文所暗示的，对于许多图类，存在一种高效的方法，可以通过一组遍历计算出唯一的候选序，从而得到多项式时间算法。接下来，对于识别问题，我们需要优化多项式的指数，而目前尚不清楚构造排序和验证排序哪一部分的代价更高。

本文聚焦于验证部分：我们研究有序图中检测模式的细粒度复杂性。我们希望回答的问题包括：给定一个有 k 个顶点的模式，能否用远低于 n^k 的时间检测到它？是否独立于 k？能否在线性时间内完成？模式的某些参数是否决定了复杂度？哪些模式难以检测？

尽管我们最初的动机是更好地理解识别算法和模式刻画，但我们的研究也与一个研究如何高效检测（无序）图中子图的领域密切相关，即子图同构。近年来，人们对这个问题的细粒度复杂度进行了广泛研究（参见例如 [6] 及其中的参考文献），我们希望在此引入的新问题和新技术也能为这个更成熟的领域提供启发。反过来，这种交流已经颇有成效：我们借鉴了他们的一些技术和见解。

最后，这项工作也延续了认证算法这一研究方向 [17]。在这类算法中，算法不仅计算出答案，还会给出一份证明证书，以便快速验证解是否正确。其动机在于，代码或算法的执行过程可能存在错误。典型的例子包括通过二着色和一个平面嵌入，分别认证二分图性和平面性。在模式刻画中，避免该模式的排序就是一份证明证书，而本文关注的是对其进行认证的复杂度。

1.2 论文结构与结果概述

由于篇幅限制以及为了提高可读性，本文主要概述我们的结果和技术。大部分精确的形式化陈述和证明将推迟至附录。概述分为三节，分别对应所获得的三类结果。在第3节中，按照图类识别文献中的传统，我们首先证明，可以在线性于有序图边数的时间内检测出若干有趣类型的模式：三个顶点上的大多数模式、基本上属于森林的模式、一些源自几何的模式等。

在有序图的边中。随后，在第 4 节中，我们证明该问题的难度至多与在有序图中检测团相当；接着利用细粒度复杂性技术，我们证明了下界；特别是，我们证明，对于四个顶点上的一个大型模式族，人们不太可能找到优于平方时间的算法。最后，在第 5 节中，我们确定了一个称为合并宽度的参数，使得对于合并宽度为 t 且与 t 无关的常数 c，识别模式的复杂度为 $O(n^{ct})$。我们证明，对于与外平面图相关的多种模式类型，该参数是有界的。

2 模型、定义和基本性质

我们首先形式化定义模式及相关概念。假设读者熟悉基本的图论概念。

定义 1. 有序图（G, τ）是一个无向、有限、简单且无环的图，记作一对（$V(G)$ $E(G)$），其中 $V(G)$ 是顶点集，并按照 τ 完全有序； $E(G)$ 是边集。顶点数记为 n，边数记为 m。

作为输入给出的有序图采用邻接表进行编码。

定义 2. 模式 P 是一个四元组（$V(P)$ $M(P)$ $F(P)$ $U(P)$），其中 $V(P)$ 是一个顶点的全序集，并且每一对无序顶点均属于以下三个互不相交的集合之一： $M(P)$、 $F(P)$ 和 $U(P)$，分别称为强制边、禁用边和未决定对。

在图中（例如图 1），我们用实线表示必选边，用虚线表示禁用边，不画任何线表示未决定对。此外，在我们的图示中，模式按照从左到右的顺序排列。模式的补是在相同顶点集上，将 $M(P)$ 与 $F(P)$ 互换后得到的模式。模式的镜像是将相同模式按逆序排列后得到的模式。如果模式没有未决定对，则称其为完全指定；如果没有禁用边，则称其为正模式。

请注意，文献中“模式”有多种定义，但在本文中，它始终指定义 2，遵循 [9]。

定义 3. 有序图 G 是模式 P 的一个实现，当且仅当 $V(G)$ $=V(P)$ 且 $E(G)$ $= M(P) \cup U'$，其中 $U' \subset U(P)$。如果存在顶点集 X，使得由 X 诱导的有序子图是 P的一个实现，则称有序图 G包含模式 P。如果有序图 G 不包含模式 P，则称其避开该模式。

也可以考虑一种仅包含两种边的更简单形式体系，但在处理 classicgraph

类时，模式形式体系似乎更为合适。（另请参阅 [9] 中的讨论。）模式也可以描述为包含两种弧（必选边和禁用边）的无环符号图。

我们关注以下算法问题。

问题：Detection of pattern P（其中 P 有 k 个顶点）。输入：一个有 n 个顶点和 m 条边的图 G，以其邻接表表示；以及其顶点的一个全序 τ。输出：(G, τ) 是否包含 P?

为简洁起见，我们有时将该问题称为 P-Detection。通常，我们不会明确提及 τ，而是直接假设顶点按照 τ中的排序，从 1 到 n 依次命名。我们还会提到如下推广：检测一个模式集合；在这种情况下，我们要判断有序图是否至少包含该集合中的一个模式。

请注意，检测一个模式或其镜像是两个复杂度相同的问题。此外，如果允许平方时间，那么检测一个模式或其补图也是等价的，因为我们可以对有序图取补图。

3 线性时间检测（概述）

我们研究模式检测的第一个动机源于图类识别，尤其是线性时间识别。因此，我们首先关注的问题是：哪些模式可以在线性时间内识别？请注意，这里的线性时间是指时间复杂度为 $O(m)$，其中 m 是边数。（本文讨论更高的复杂度时，大多数复杂度都将表示为顶点数 n 的函数。）

聚焦于三个顶点上的模式。由于可以预期模式越大，检测问题就越复杂，我们从三个顶点上的模式入手。文献 [9] 对可由如下性质刻画的图类进行了系统研究：存在一种顶点排序，使得某个给定的三个顶点上的模式（或某个给定的模式族）不会出现。结果发现，这样的图类约有 20 种（以及它们的补图），而且几乎都是众所周知的图类（路径图、树、区间图、弦图、分裂图、置换图、可比图、无三角形图、二分图等）。此外，除由单个模式 Triangle、Comparability 及其补图定义的图类外，所有这些图类都可以在线性时间内识别 [9], 。（对于三个顶点上的模式，我们将采用 [9] 中的记法。具体而言，模式以小型大写字母命名，这些名称与禁用它们的图类相关。详情见附录 A。我们刚才提到的模式见图 2。）

识别与检测。明确区分两个算法问题非常重要。一方面，对于固定模式 P（或一组模式），识别问题的输入是一个图，问题是该图是否存在一种能够避开该模式（或这些模式）的排序。另一方面，对于固定模式 P（或一组模式），检测

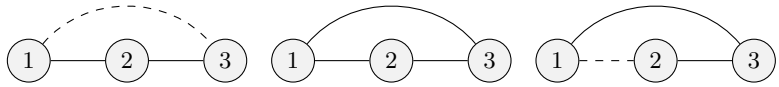


图 2：模式 Comparability、 Triangle 和 Chordal 的示意图

问题的输入是一个有序图，问题是该模式是否出现在这个有序图中。

这两个问题一般来说不可比较。识别问题似乎更难，因为在这个问题中，我们需要判断是否存在具有某种给定性质的顶点排序；而在检测问题中，给定的是排序。但在许多情况下，有非常高效的方法可以生成具有很强性质的排序，使得该排序要么避开该模式，要么不存在任何能避开该模式的排序。在这些情况下，检测问题可能更难，因为必须设计一种算法，能够在任意输入排序中检测该模式，而不只是针对具有特定结构的排序。

检测三个顶点上的一个模式。我们证明以下定理。

定理1。设 P 是一个包含三个顶点的模式。如果 $\mathcal{P}$ 既不是 Comparability，也不是 Triangle，且不是它们的补图之一，则可在线性时间内检测 P 。

请注意，困难情形与识别问题中的相同。实际上，对于这些模式，生成一个良好的候选序在某种程度上比检查该排序更容易。确实，除 Comparability 模式外，所有模式的复杂度都相同；对于该模式，生成一个良好的候选序可以在线性时间内完成 [18] 而人们相信其检测无法在次二次时间内完成。

使用邻域操作处理简单情形 定理 1 的证明中的所有情形都可以用类似方法处理，但 Chordal 除外（见图 2），我们稍后将讨论该情形。首先，对于每个顶点 i，我们构造 $N^-(i)$ 和 $N^+(i)$，它们分别是 i 的前驱和后继的有序邻接表。这些（有序）列表可以从无序邻接表中在 $O(m)$ 时间内计算得到，因为编号介于 1 和 n 之间。我们以 Interval 为例说明这一技术。显然，从该模式可以看出，当且仅当对于所有 i， $N^+(i)$ 为空，或其构成一个顶点区间时，该模式才不存在： $[i+1, j]$ 对于某个 j。由于检查这一性质时，每个顶点的邻域仅被扫描 $O(1)$ 次，因此复杂度为 $O(m)$。

Chordal的情况。模式检测的一般经验法则是：已确定的边越多，检测就越困难。在三个顶点上， Trian-gle 和 Comparability （以及它们的补图）先验而言是最难识别的，但最后一个完全指定模式 Chordal有所不同：我们可以在线性时间内检测出它。

已经在 [11,23] 中证明，当排序由一种称为 LexBFS 的图遍历生成时，我们可以在线性时间内检测 Chordal 。但我们希望得到一种不利用 LexBFS 所提供的良好结构的算法。请注意，检查一个顶点的前驱是否构成一个团这一自然算法的时间复杂度是二次的。相反，在我们的算法中，对于每个顶点 i，我们只检查该模式是否出现在某些位置（k、j、i）上，其中 j 是 i最右侧的前驱。我们证明，得益于该模式的结构，这种部分验证足以确保正确性，并且可以在线性时间内完成。

刻画其他类的模式扩展。我们证明以下扩展，涵盖所有可由三个顶点上的模式集合刻画的类。

定理 2。考虑由三个顶点上的禁用模式集合刻画的图类 $\mathcal{C}$，该图类既不是（余）可比图类，也不是（余）无三角形图类。存在一个禁用模式集合，可刻画 $\mathcal{C}$，且可在线性时间内检测。

因此，由于我们可以在线性时间内为（余）可比性构造一个候选排序 [18], ，我们得出，对于每个由三个顶点上的模式集合刻画的图类，找到排序比验证排序更容易。对于更大的模式则不然：含有 k 个顶点的路径模式对应于可用 k 种颜色着色的图，因此寻找合适的排序是 NP 困难的。

其他线性时间情形 我们还证明，其他类型的模式也可以在线性时间内识别。正式的定理和证明将在附录中给出，大致而言，我们的结果如下。首先，所有必选边构成无交叉森林的正模式，都可以在线性时间内识别（附录 D）。其次，所有必选边构成路径的四顶点正模式，都可以在线性时间内识别（附录 E）。最后，在文献中研究过的少数几个含有三个以上顶点的模式中，交图理论中研究过的两个模式可以在线性时间内识别（附录 F）。所用技术是针对三个顶点上的模式所用技术的推广和变体，也是我们将在第 5 节介绍的技术的推广和变体。

4 归约与下界（概述）

回想一下，几乎所有由三个节点组成的模式都可以在线性时间内验证（定理 1）。接下来顺理成章的一步是研究四节点模式的情况。在积极的一面，我们证明了存在比 n^k 时间更快的算法，可以验证任意 $k \geq 4$ 节点模式。不幸的是，本节还提供了证据表明，许多此类模式只能在（至少）平方时间内检测到，即使对于 $k = 4$ 也是如此。

4.1 从团检测的一般归约

回想一下，任意 k 节点模式都可以在 n^k 时间内验证。

然而，存在一种更快的算法：其运行时间约为 $n^{\omega k/3}$ ，其中 ω 表示方阵乘法的指数，这得益于以下定理。

定理3. 存在一种归约，可将有序图 $G=(V,\ E,\ \tau)$ 上具有 k 个节点的模式 P 的 P-Detection 问题（其中该图有 n 个顶点），归约为在具有 kn 个顶点的图 G' 中检测一个 k-团。

注意，模式 P 无需完全指定。下面我们描述该归约；图3以 $k=3$ 为例进行了说明。 G' 的顶点集等于 $V\times\{1,2,\ldots,k\}$。对于每个 u、$v\in V$，以及每个满足 $1\leq i<j\leq k$ 的 i、j，当且仅当以下某组条件成立时，我们就在 G' 中的（u，i）与（v，j）之间添加一条边：

– P的 i^{th} 和 j^{th} 顶点之间有一条必选边， u 和 v在 G中相邻， $\tau(u)<\tau(v)$；– P 的 i^{th} 和 j^{th} 顶点之间有一条禁用边， u 和 v 在 G 中不相邻， $\tau(u)<\tau(v)$；– P的 i^{th}和 j^{th}顶点构成一个未决定对， $\tau(u)<\tau(v)$。

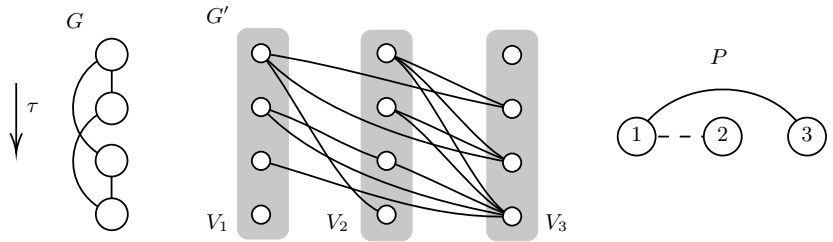


图 3：从 P-Detection 到团检测的归约示意图。左侧是从上到下排序的有序图（G,τ），右侧是我们要检测的模式。中间是通过归约构造的图 G'。这里可以在（G,τ）的最后三个顶点中找到该模式，这意味着我们可以在 G'中找到一个3-团：考虑 V_1的第二个顶点、V_2 的第三个顶点和 V_3的第四个顶点。

该归约可在 $\mathcal{O}((kn)^2)$ 时间内完成。容易验证，当且仅当 G' 中存在一个 k-团时， G 中存在一个能够实现 P 的有序 k-子图。

4.2 下界

我们证明了模式检测问题的条件下界。实际上，我们运用多种技术，并基于不同的复杂性假设，证明了若干结果。完整的结果列表请参见附录B。

在这篇概述中，我们将重点讨论大小为 4 的模式，并介绍我们的结果大致是什么。下面列出我们采用的复杂性假设，这些都是经典假设。关于这些假设的合理性及其在细粒度复杂性领域中的影响，请参见 [5,16,26,27] 。另请参见附录中的 B.1 小节。

假设 1 对于每个 $k \geq 3$，判定一个图是否包含一个有 k 个顶点的团（分别为独立集）需要 $n^{\omega(\lfloor k/3 \rfloor, \lceil k/3 \rceil, \lceil (k-1)/3 \rceil) - o(1)}$ 时间，其中 $\omega(p,q,r)$ 是乘法维度分别为 $n^p \times n^q$ 和 $n^q \times n^r$ 的两个矩阵时的指数。假设 2 对于检测图中的三角形，任何组合算法（不使用快速矩阵乘法或其他代数技术）都需要 $n^{3-o(1)}$ 时间。假设 3 判定一个 3-一致超图是否包含一个有四个节点的超团需要 $n^{4-o(1)}$ 时间。

以下定理综合了我们的大部分结果（并对其中一些结果作了弱化，使它们能够归入同一陈述）。

定理 4. 考虑一个定义在四个顶点上的完全指定模式 P。那么，假设假设 1、2 和 3 成立，检测 P 无法通过组合方式在次二次时间内完成。对于大多数此类模式 P，该下界对任何算法（不一定是组合算法）也成立。

本节余下部分将概述定理 4 的证明，并补充一些讨论。需要强调的是，这只是非正式的证明概述；所有证明都在附录 B 中以详细且正式的方式给出。

归约到子图检测的概率归约。由于上述列出的三个假设即使对于随机化算法也被认为成立，我们可以使用正确概率虽小但为常数的随机化归约。

粗略地说，我们将验证完全指定模式的复杂度与一个相关（无向、无序）图H的结构和算法性质联系起来。具体而言，如果移除排序和禁用边后得到 H，那么完全指定模式就是基于 H的。假设我们能够高效地检测基于 H的模式 P_H。那么我们证明，可以在相同的复杂度（再加上线性加法项）下，以常数概率判定图 G是否包含 H 作为诱导子图。为此，给定图 G，我们生成其顶点的一个随机排列（使用 Fisher-Yates 洗牌算法[7,10]），在这个有序图上运行 P_H检测算法，并输出相同的判定结果。如果诱导子图 H 存在，那么算法至少以$1/(k!)$的概率回答是；如果它不存在，该算法总是回答否。（参见附录中的命题 9。）

现在，几乎整个证明都在讨论如何调整无序子图检测的所有已知结果，使其能够通过上述归约，从而得到我们想要的结果。这比乍看之下更具挑战性，因为我们并不完全理解

无序子图检测的复杂度。因此，有时我们可以以黑箱方式复用已有结果，但通常需要调整或补充已知证明，并且在某些情况下还要从头构造临时归约。

图的术语表。证明定理 4 时，我们将考虑所有大小为四的模式，但大多数时候，我们可以一次处理所有基于同一张图的模式。这类四顶点图的完整列表见图 4，图中标出了我们将使用的名称。

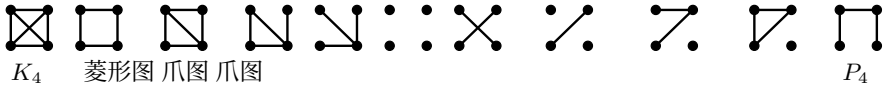


图 4：四个顶点上的 11 种（无序）图： K_4、 C_4、菱形图、爪图、爪图、它们各自的补图，以及其补图就是自身的 P_4 。

团、独立集和 C_4。团的子图检测问题最容易理解。实际上，完全模式（所有边都是必需的）这一情形可直接由假设 1 得出，无需概率归约，因为排序无关紧要。现在，若一个子图难以检测，那么包含它的任何更大子图也必定难以检测，这似乎是自然而然的结论。但一般来说，这一点尚未得到证明。只有团的情形已得到证明，见 [5]。因此，对于任何包含三角形的四顶点模式，根据假设 1，其所需时间至少为 $n^{\omega-\varepsilon}$ （其中 $\omega=\omega(1,1,1)\in[2,2.3727]$ [25]）。

至此，我们已经证明了定理 4 对所有以 H 为基础且 H 为 K_4、菱形图、爪图或 K_3+K_1 的模式都成立。现在，借助补图，我们也得到了 $4K_1$ 的下界。但对于大小为三的独立集，我们尚未发现可以直接使用的结果（也就是 [5] 的类似结果）。因此，对于爪图、 K_2+2K_1 和 P_2+K_1 的情形，我们设计了从检测 $3K_1$ 归约到检测这些图的直接归约。参见附录中的引理 1。

以 C_4 为基础的模式这一情形比较特殊，因为这是唯一需要使用假设 3、且只能得到 $n^{2-\varepsilon}$ 下界的情形。这是通过直接使用 [6] 最近的结果实现的。

新工具的必要性、有向路径与完成证明。至此，我们还剩下 $H=P_4$ 或 $2K_2$。为了证明这些情形的下界，我们需要付出最多努力。对于这些情形，我们无法使用迄今为止采用的概率归约，因为在子图情形下，这些问题尚无已知的非平凡下界。更糟的是，在图中检测 P_4 实际上并不难，因为不含 P_4 的图就是余图，而众所周知，余图可以在线性时间内识别 [4]。

我们基于有向路径的概念，设计了一个新的通用结果。模式中的有向路径是指模式中的一条路径，其顶点遵循模式的顺序。我们证明，如果模式 P 包含一条包含 t 个顶点的唯一有向路径，那么就存在一个 $O(n^2)$ 时间归约，可将 t-团检测归约为 P-检测。详见附录中的命题 10。除了改进

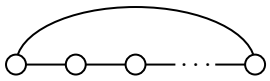
基于菱形图、爪图或 C_4 的某些模式外，这还使我们能够为组合算法得到 $n^{3-o(1)}$ 下界；其前提是，对所有基于 P_4 的模式，假设 2 均成立，其中路径包含三个（不一定连续的）顶点u、v、w，且它们在模式中按此顺序出现。

借助有向路径技术，我们解决了各种基于 P_4 的模式，也通过补图处理了一些基于 $2K_2$ 的模式。最后，我们通过若干临时归约解决其余情况，特别是从三角形检测出发的归约。参见附录B.4。

四个以上的顶点。我们的主要目标是彻底理解四个顶点的完全指定模式这一情形。在此过程中，我们得到的若干结果实际上更为一般（从诱导子图检测出发的概率归约、包含团的模式情形，以及有向路径技术）。我们再证明一个这样的结果：如果 P 是较大模式 Q 的前缀，那么检测 Q 至少与检测 P 一样困难（在随机化归约下）。参见附录B.5。

5 参数化算法（概述）

本节将介绍一种针对模式的参数——合并宽度，并设计一种关于 n 多项式时间的算法，其复杂度指数仅取决于该参数。首先，我们通过下面这个模式来直观说明我们的方法，我们称之为长度为 k 的平坦环。



为了在有序图 G中检测此模式，我们采用动态规划方法。首先，对于每个 $x \in [2,k]$，我们计算所有满足 $1 \le i < j \le n$的对（i，j），使得在 G中存在一条长度为 x 的路径，其顶点索引递增，最左侧顶点为 i，最右侧顶点为 j。对于 $x=2$，可以直接利用 G的邻接矩阵完成此操作。对于任何 $x \ge 3$，我们可以通过考察为 $x-1$计算出的区间 $[i,j']$，并检查 $E(G)$中是否存在边（j'，j），$j' < j$，轻松完成此操作。现在，对于每个对应于一条经过 k 个顶点的路径的对（i，j），我们检查（i，j）是否是 G 的一条边。该算法的复杂度至多为 $O(n^3)$。

请注意，复杂度并不取决于 k。相反，复杂度源于我们操作了成对的位置，它们对应于“子模式”的端点。如果该模式只有路径而没有反馈边，那么我们只需存储区间的右端点：只要记住存在一条长度为 x 、终止于某个给定顶点的子路径即可。但由于需要添加反馈边，因此必须存储两个端点。换言之，我们需要将路径的两个端点作为锚点，以便与反馈边合并。

合并宽度的直观理解及其用法。我们为模式定义了一个参数，称为合并宽度。我们考虑几种用于构造模式的操作：创建顶点和边，以及合并模式。在两个模式的合并过程中，我们会将第一个模式中的某些 r 顶点与第二个模式中的 r 个顶点对应起来（顺序保持一致）。在构造的任何一步中，任意顶点都可以承担一种特殊角色，我们称之为锚点。例如，在合并中被对应起来的顶点必须是锚点。合并宽度是在所有可能的模式构造方式中，取所有中间模式的锚点数量最大值后所得的最小值。

我们的算法将采用与示例类似的方式：我们将自底向上遍历模式的构造，并在每一步记录锚点所有可能的位置集合。随后，对于一次合并，我们将比较这些列表，找出在合并中涉及的所有锚点上都匹配的列表，并保留这些锚点列表的合并。算法的复杂度主要取决于我们所处理的位置集合的大小，而这由合并宽度衡量。因此，我们的算法复杂度为 n^{ct}，其中 c 是一个常数，t 是合并宽度。

难点：锚点的定义。确定哪些顶点应当作为锚点并非易事。仅保留某个时刻需要合并的顶点并不足够，因为在目标模式中，有些顶点必须按特定顺序出现，而只保留“待合并顶点”无法控制这一点。见图 5。

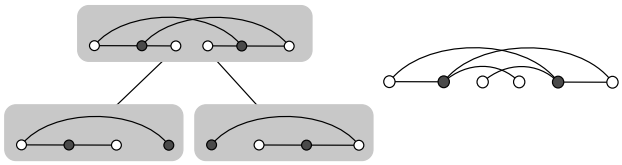


图 5：考虑左侧的合并，其中锚点是黑色顶点。如果没有更多约束，也可以得到右侧的模式。在这种情况下，算法可能会输出“是”，而正确答案是“否”。

为避免这种情况，我们定义了一组约束，规定模式的构造在锚点放置方面应满足这些约束，并证明这足以确保算法的正确性。参见附录中的 C.1 小节。

通过矩阵乘法加速 我们利用矩阵乘法加速合并情况下锚点位置的计算。例如，在这个示例中，矩阵乘法可以得到 $O(n^{\omega})$，而不是 $O(n^3)$；这一方法也可推广到包含两个以上锚点的合并。

应用。我们证明，可以在不发生边交叉的情况下绘制的模式（外平面模式）的合并宽度为 2；如果交叉数量有上界 r，则合并宽度至多为 $2r+2$。我们认为，对于书厚度有界的模式，以及某种“模式树宽度”概念，合并宽度也应当有界，但这有待进一步研究。

6 开放问题

我们提出了有序图的模式检测问题。关于这一新课题，还有许多有趣的方向有待探索。下面列出了一些尚未解决的问题。

– 我们能否在下界方面更进一步？利用子图检测的结果可能难以带来更多帮助，因为我们已经触及当前知识的极限（例如，我们所使用的关于 C_4 的结果非常新）。但或许可以构建更多专门的证明？– 合并宽度是刻画该问题难度的恰当参数吗？我们能否证明，可以快速检测的模式具有较小的合并宽度？– 合并宽度是否可与某个已知参数相比较？书厚度可能是这里的一个候选参数。– 正平面森林是否是可以在线性时间内识别的最大自然模式类？– 如果我们研究的不是有序图，而是偏序图，情况会有很大不同吗？– 关于“几何模式”，模式 P_{ac} 能否在线性时间内测试，因为它是两个可在线性时间检测的模式 P_a 和 P_c 的某种组合？

参考文献

1. Frank Pok Man Chu. 一种用于识别平凡完美图及其补图的简单线性时间认证型基于 LBFS 的算法。 *Inf. Process. Lett.*, 107 1 7 12, 2008 ( ): – 。2. Derek G. Corneil、J´er´emie Dusart、Michel Habib 和 Ekkehard K¨ohler。关于图搜索在余可比图中的能力。 *SIAM J. Discrete Math.*, 30 1 569 591, 2016 ( ): – 。3. Derek G. Corneil 和 Richard Krueger。图搜索的统一视角。 *SIAM J. Discrete Math.* 22 4 1259 1276, 2008 , ( ): – 。

4. Derek G. Corneil、Yehoshua Perl 和 Lorna K. Stewart。余图的线性识别算法。 *SIAM J. Comput.*, 14(4):926–934, 1985。5. Mina Dalirrooyfard、Thuy Duong Vuong 和 Virginia Vassilevska Williams。图模式检测：所有诱导模式的困难性与更快的非诱导环。 *Proceedings of the 31st Annual ACM-SIAM Symposium on Theory of Computing* 载于 1167 1178, 2019 , 页码 – 。6. Mina Dalirrooyfard 和 Virginia Vassilevska Williams。诱导环和路径比你想象的更难。载于 *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)* 531 542 2022 , 页码 – 。IEEE, 。7. Richard Durstenfeld。算法 235 *Communications of* : 随机排列。 *the ACM* 7 7 420, 1964 , ( ): 。8. Laurent Feuilloley 和 Michel Habib。通过有序禁模式对基于线段的交图进行分类，12 月，http://arxiv.org/abs/2112.00629ø，2021。9. Laurent Feuilloley 和 Michel Habib。图类与三个顶点上的禁用模式。 *SIAM Journal on Discrete Mathematics (SIDMA)* 35 1 55 90, , ( ): – 。2021
10. Ronald Aylmer Fisher 和 Frank Yates。 *Statistical tables for biological agricultural and medical research* , 。爱丁堡：Oliver and Boyd, 1963 。11. Philippe Galinier、Michel Habib 和 Christophe Paul。弦图及其团图。载于 *Graph-Theoretic Concepts in Computer Science, 21st International Workshop, WG '95, Aachen, Germany, June 20-22*, 1995, *Proceedings*, 页码 358–371, 1995。12. Tibor Gallai。可传递定向图。 *Acta Math. Hungarica*, 18 66, 1967 1 25( ): – 。13. Michel Habib、Ross M. McConnell、Christophe Paul 和 Laurent Viennot。Lex-bfs 与划分细化，及其在传递定向、区间图识别和连续 1 测试中的应用。 *Theor. Comput. Sci.*, 234(1-2):59– 84, 2000。14. Xiaohan Huang 和 Victor Y Pan。快速矩形矩阵乘法及其应用。 *Journal of complexity*, 14(2):257–299, 1998。15. C Lekkeikerker 和 J Boland。用实直线上的一组区间表示有限图。 *Fundamenta Mathematicae*, 51(1):45–64, 1962。16. Andrea Lincoln、Virginia Vassilevska Williams 和 R. Ryan Williams。稀疏图中最短环和路径的紧下界。载于 *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms* 1236 1252 , 页码 – 。SIAM, 2018。17. Ross M. McConnell、Kurt Mehlhorn、Stefan N¨aher 和 Pascal Schweitzer。认证型算法。 *Comput. Sci. Rev.*, 5(2):119–161, 2011。18. Ross M. McConnell 和 Jeremy P. Spinrad。模分解与传递定向。 *Discrete Mathematics*, 201(1-3):189–241, 1999。

19. Stephan Olariu。无爪图。 *Information Processing Letters* 28 1 53 54, , ( ): – 。20. 1988
Donald J. Rose、R. Endre Tarjan 和 George S. Lueker。图上顶点消去的算法方面。 *SIAM Journal on Computing*, 5(2):266–283, 1976。21. Marcus Schaefer。图交叉数及其变体：综述。*electronic journal of combinatorics* 21 2012 , 第 *The* DS –Apr 页, 。22. Robert E. Tarjan 和 Mihalis Yannakakis。补遗：用于检验图的弦性、超图的无环性以及选择性约简无环超图的简单线性时间算法。 *SIAM Journal on Computing*, 14(1):254–255, 1985。23. Robert Endre Tarjan 和 Mihalis Yannakakis。用于检验图的弦性、超图的无环性以及选择性约简无环超图的简单线性时间算法。 *SIAM J. Comput.*, 13(3):566–579, 1984。24. Marc Tedder、Derek Corneil、Michel Habib 和 Cristophe Paul。通过递归因式分解排列实现更简单的线性时间模分解。载于 *ICALP*, 第 634–645 页。施普林格, 2008。25. Virginia Vassilevska Williams。比 Coppersmith–Winograd 更快地进行矩阵乘法。载于 Howard J. Karloff 和 Toniann Pitassi 编， *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY USA 2012* 887 898 2012 , , *May 19 - 22*, , 第 – 页。美国计算机协会, 。26. Virginia Vassilevska Williams。关于算法与复杂度中的一些细粒度问题。载于 *Proceedings of the International Congress of Mathematicians: Rio de Janeiro 2018* 3447 3487 2018 , 第 – 页。世界科技出版公司, 。27. Virginia Vassilevska Williams 和 R. Ryan Williams。路径、矩阵和三角形问题之间的次立方等价。 *J. ACM*, 65(5):27:1–27:38, 2018。28. J.-H Yan、J.-J Chen 和 G.J. Chang。准阈值图。 *Discrete Applied Math.* 69 3 1996 , ( ), 。

附录目录

– 附录 A：三个顶点上的模式检测。– 附录 B：具有 $k \geq 4$ 的模式的条件下界。– 附录 C：参数化算法与合并宽度。– 附录 D：正平面森林的线性时间检测。– 附录 E：每个正 P_4 的线性时间检测。– 附录 F：几何中产生的模式的线性时间检测。

A 三个顶点上的模式检测

三个顶点上的模式与 classicgraph 图类的研究密切相关。事实上，许多图类都可以通过某种顶点排序来刻画，即该排序不包含一个或多个这样的模式。在本节中，我们将详细研究三个顶点上的模式检测问题。

我们证明两个定理。第一个定理针对单个模式的检测。

定理1. 设 P 是一个三个顶点上的模式。如果 $\mathcal{P}$ 既不是 Comparability，也不是 Triangle，也不是它们的补图之一，那么可以在线性时间内检测 P 。

对此的一种自然推广，是考虑所有由三个顶点上的模式组成的集合。由于这需要进行冗长的分类讨论，本文这一版本采用了一种不那么系统、但仍有意义的做法。由此类模式集合定义的图类列表见 [9]。对于每个这样的类，我们考察定义它的其中一个模式集合。

定理 2. 设图类 $\mathcal{C}$ 由三个顶点上的一组禁用模式刻画，且它既不是（余）可比图类，也不是（余）无三角形图类。则存在一组刻画 $\mathcal{C}$ 的禁用模式，可以在线性时间内检测。

本节其余部分将用于证明这些定理。定理 1 分两部分证明：A.2 和 A.3 小节（在后者中，我们研究模式为 Chordal 或余-Chordal 的情形）。定理 2 在 A.4 小节中证明。

A.1 名称约定与观察

在本节中，为便于阅读，我们将使用 [9] 中指定的名称来称呼由三个顶点构成的模式。这些名称列于图 6 中。它们与禁止这些模式出现的类相关，并以小型大写字母书写。

为减少需要考虑的这 3 个节点模式的情形数量，我们可以依据以下观察结果。

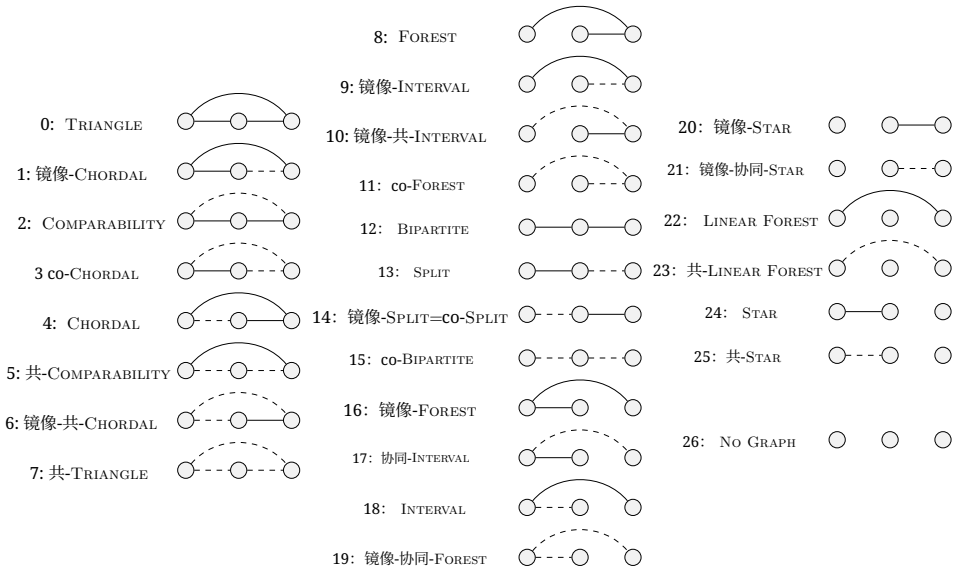


图 6：三个节点上的 27 种模式。按照惯例，由于镜像-Split=共-Split，我们将忽略模式镜像-Split。

注 1. 设 $f(n,m)$ 表示判定图 G 的顶点全序是否避免模式 P 的复杂度，则判定一个排序是否避免镜像模式 $mirror(P)$ 的复杂度 $mirror\text{-}f(n,m)$ 满足 $mirror\text{-}f(n,m) \leq max\{f(n,m)\ g(n,m)\}$，其中 $g(n,m)$ 表示改变数据以将 τ 转换为 τ^d 的复杂度。

证明。如果我们有一个算法 $\mathcal{A}$，能够判定图 G 的顶点全序排序 τ 是否不含模式 P，只需将其应用于图 G 和逆序排序 τ^d，即可证明 τ^d 不含模式 P，因此 τ 不含模式 $mirror\text{-}P$。□

在大多数情况下，$g(n,m)$ 属于 $O(n+m)$，因为只需根据 τ^d G 的邻接表进行排序即可。因此，如果能在线性时间内判定模式 P，那么也能在线性时间内判定其镜像模式 $mirror\text{-}P$。

注 2. 设 P_1 和 P_2 是两个模式，$f_1(n,m)$（分别为 $f_2(n,m)$）表示判定图 G（有 n 个顶点和 m 条边）的顶点排序 τ 不产生模式 P_1（分别为 P_2）的复杂度，那么判定图 G（有 n 个顶点和 m 条边）的顶点排序 τ 不产生模式 P_1 和 P_2 的复杂度$f_{12}(n,m)$ 满足

$f_{12}(n,m) \leq max\{f_1(n,m)\ f_2(n,m)\}$.

因此，一组可在线性时间内判定的模式也可以在线性时间内判定。

注释 3. 设 $f(n,m)$ 表示判定图 G 的顶点排序是否避免了刻画图类$\mathcal{G}$的一组模式 $\mathcal{P}$ 的复杂度。那么，补图类可以在 $O(max(\{f(n,m)\ g(n,m)\})$ 内判定，其中 $g(n,m)$ 表示从 G 构建 $\overline{G}$ 的复杂度。

证明。从一个图 G 出发，我们首先构造其补图 $\overline{G}$。如果我们能够通过一种避免模式 $\mathcal{P}$的排序来刻画 $\overline{G}$，那么只需在 O（f、 m）中判定它们即可。□

因此，为了刻画和判定一个补图类，我们可以使用由补模式构成的模式集合（交换必选边和禁用边）。此外，在某些示例中无需计算补图，原因将在下文中说明。

最终列表。如果移除作为另一个模式镜像的模式，我们将得到以下包含18种模式的列表（见图7）： Triangle、co-Triangle、 Comparability、co-Comparability、 Chordal、co-Chordal、 For-est、co-Forest、 Interval、co-Interval、 Bipartite、co-Bipartite、 Split、Star、co-Star、 Linear Forest、co-Linear Forest、 No Graph。

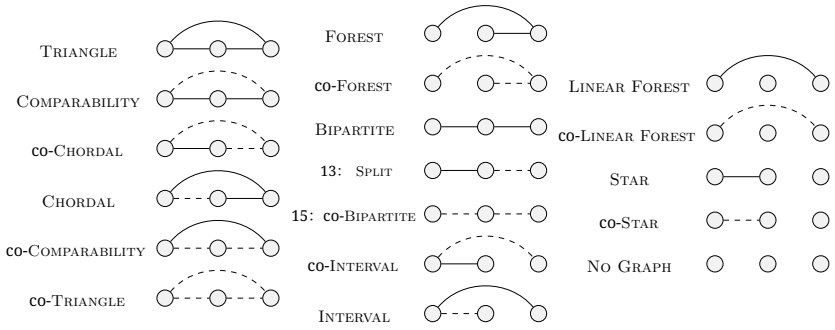


图7：移除镜像后，三个节点上的18种模式。

A.2 单一模式，简单情况

对于大多数三节点模式，检测它基本上归结为比较顶点的邻域。请记住， $N^{-}(i)$ 和 $N^{+}(i)$ 分别表示按排序排列的顶点前驱邻接表和后继邻接表。

命题 1. 以下 12 种模式（及其镜像）的检测均可在线性时间内完成：
No Graph, Split, Star, co-Star, Bipar-tite, co-Bipartite, Forest, co-Forest, Linear Forest, co-Linear For-est, Interval, co-Interval。

证明。对于所引述的每种模式，我们都建立一个性质，使得：如果该性质在每个顶点 i 处都成立，那么该模式不会出现；否则，该模式就会出现。（这种否定式表述可能看起来有悖直觉，但实际上它让条件更容易表述，也符合图类识别领域的文献惯例。）这些性质涉及列表 $N^-(i)$ 和 $N^+(i)$。

– No Graph：$|V(G)| \leq 2$。– Split：若对于某个 i，$N^-(i) \neq \emptyset$，则 $|N^+(i)| = n - i$。– Star：若对于某个 i，$N^-(i) \neq \emptyset$，则 $i = n$。– co-Star：若对于某个 i，$|N^-(i)| < i - 1$，则 $i = n$。– Bipartite：若对于某个 i，$N^-(i) \neq \emptyset$，则 $N^+(i) = \emptyset$。– co-Bipartite：若对于某个 i，$N^-(i) \neq \emptyset$，则 $N^+(i) = \emptyset$。– Forest：$\forall i$，$|N^-(i)| \leq 1$。– co-Forest 对于所有 i，$|N^-(i)| \geq i - 1$。– Linear Forest：$\forall ij \in E(G)$，其中 $i < j$，则 $i = j - 1$。– co-Linear Forest：$\forall i$，要么 $|N^+(i)| = n - i$，要么（$|N^+(i)| = n - i - 1$ 且 i，$i + 1 \notin E(G)$ – Interval：$\forall i$，$N^+(i)$ 若非空，则是一个唯一的 $[i + 1, j]$ 类型区间。– co-Interval：$\forall i$，i 右侧的非邻居（若存在）构成一个唯一的 $[i + 1, j]$ 类型区间。

由于只需扫描列表 $N^-(i)$ 和 $N^+(i)$ 一次，即可检查所有 i 的这些性质，因此复杂度为 $O(m)$。□

不过，用于检测这些模式的条件绝非唯一，也可能存在其他正确的条件。

A.3 一种模式：弦图和余弦图的特殊情形

模式 Chordal 和余-Chordal 在本文中具有特殊地位。事实上，它们是完全指定的模式示例，因此与下界部分研究的模式以及模式 Triangle 和 Comparability相似；但与这些模式不同，它们可以在线性时间内检测。

然而，这里的证明不如前几个情形那么直接。事实上，朴素算法会检查每个顶点的左邻域是否构成一个团，其复杂度为二次。

Chordal 检测。在此之前，已有一些用于检测模式 Chordal 的算法 [11,22], ，但只有当图排序来自一种称为 LexBFS 的图遍历时，这些算法才能做到这一点。这些排序具有我们无法利用的强性质，例如，第一个顶点在图中具有某种特殊作用。我们提出以下算法（算法1）。

输入： G 由邻接表给出的图，以及 τ 顶点的全序 输出：(G,τ) 是否包含 Chordal? 不失一般性，假设 $\tau = 1, 2, \ldots, n$预先计算每个顶点 i 的 $N^-(i)$。对于 $i = 3$ 到 n，执行：如果 $N^-(i) \neq \emptyset$：令 $j \in N^-(i)$ 为索引最大的顶点。如果非 $N^-(i) \setminus j \subseteq N^-(j)$，则返回“是，找到 Chordal ”。结束。返回“否，未找到 Chordal ”。算法1：模式 Chordal的检测算法。

命题2. 算法1可在线性时间内检测模式 Chordal 。

证明. 我们先证明正确性，然后证明复杂度

正确性. 如果对于某个 i，其索引最大的邻居 j不满足 $N^-(i)\setminus j \subseteq N^-(j)$，则可得到某个顶点 k，使得 $ki, jk \in E(G)$ 且 $kj \notin E(G)$，因此出现了 Chordal 。反过来，如果算法以答案 No 终止，我们声称该模式不存在。为了导出矛盾，假设该模式存在，但算法没有输出 yes。令 i 为使模式以最右侧顶点 i出现的最小整数。索引 i 必须严格大于 3，否则算法就会检测到该模式。现在令 $z < t$ 为该模式实现中的其他顶点。我们有 $z, t \in N^-(i)$，使得 zi, $ti \in E(G)$ 且 $zt \notin E(G)$。顶点 t 不可能是 i中索引最大的顶点，否则算法就会检测到该模式。因此，存在 j，使得 $ji \in E$，且 $z < t < j < i$。由于算法没有输出 yes，我们必有 $N^-(i)\setminus j \subseteq N^-(j)$。于是必然有 $zj, tj \in E(G)$，并由此在 z, t, j上得到一个禁构型。这与 i的最小性矛盾。

复杂度。每个顶点在 for 循环中以及评估测试 $N^-(i) \setminus j \subseteq N^-(j)$ 时，最多会被考虑一次。由于顶点 j 可能多次成为右侧最近的邻居，因此我们必须谨慎组织此测试，以确保其为线性时间。一种实现方法是每次都将列表 $N^-(i) \setminus j$ to$j$关联起来。然后，在第二次扫描序列 τ 时，我们将这些关联列表与列表 $N^-(j)$进行比较。这样，$N^-(j)$的所有列表总共只需扫描一次。因此，整个过程是线性的。□

请注意，此检测算法可以嵌入已知的识别算法中，并使用 LexBFS、LexDFS 或 MCS 等特定搜索方法[3]。

该算法采用了一种范式，我们可以称之为“依赖最右侧的左邻居”。在接下来的几节中，我们将推广这一思想，以在线性时间内认证更复杂的模式。

co-CHORDAL 检测。请注意，要在线性时间内检测模式 P 的补，不能通过对图取补后再检测 P 来实现，因为这样需要平方时间。在某些情况下，可以隐式地处理补图，例如使用划分细化时。由于我们上面的算法并非如此，我们需要为 co-CHORDAL 的情况定义一个算法。我们提出以下算法（算法 2）。

输入： G 一个以邻接表给出的图，以及 τ 顶点的一个全序输出： (G, τ) 是否包含 co-CHORDAL? 不失一般性，假设 $\tau = 1, 2, ..., n$ 对每个顶点 i, 预先计算 $N^-(i)$。对于 $i = 3$ *to* n 执行如果 $N^-(i) \neq \emptyset$ 令 $j \in N^-(i)$ 为具有最大索引的项 如果 NOT($N^-(j) \subseteq N^-(i)$) 返回“找到 co-CHORDAL ”结束输出“未找到 co-CHORDAL 。” 算法 2：认证一个单纯消去排序

命题 3。算法 2 在线性时间内检测 co-CHORDAL 。

证明。证明与前一个算法基本相同。唯一需要检查的是，“令 $j \in N^-(i)$ 为具有最大索引的项”这一部分可以在时间 $O(|N^-(i)|)$ 内计算，这确实成立。□

至此，定理1的证明完成。事实上，这样的模式共有 18 种（忽略镜像），我们已在命题 1 中给出了其中 12 种模式的线性时间检测方法，而命题 2 和命题 3 又给出了另外两种。剩下的四种正是定理中列为例外的模式。

A.4 对应于图类的集合

现在我们证明定理 2。对于每个可由三个顶点上的一组禁用模式刻画的图类，我们都考虑这样一组禁用模式，并证明除可比性、余可比性、无三角形和余无三角形这些图类外，都可以在线性时间内检测。

根据 [9], 中的刻画定理 2，我们知道，除补图外，恰好存在 22 个可通过三个节点上的禁边集定义的非平凡类。（平凡类是有限类，可以在常数时间内处理。）

1. 森林 9. 无三角形 16. 增强团 2. 线性森林 10. 置换 17. 二星 3. 星 11. 二部置换 18. 二部链 4. 区间图 19. 毛毛虫 5. 分裂图 12. 团 20. 平凡完美 6. 二分图 13. 阈值 21. 无三角形 7. 弦图 14. 真区间 ∩ 余弦图 8. 可比性 15. 一分裂 22. 完全二部

所列出的类大多是经典类。例外包括一分裂、二星和增强团，它们分别是分裂图、星图和团的小变体。我们无需在此对它们进行严格定义，因此请参阅 [9]。

简单情形 本段所述内容均请参阅 [9] 。

对于若干类，由于它们可以通过唯一模式刻画，因此结论可由定理 1 得出。这些类包括：森林、线性森林、星、区间图、分裂图、二分图和弦图、团，以及它们各自的补图。对于其他一些类，存在一个能够刻画它们的模式集合，并且其中所有模式都可以在线性时间内检测。在这种情况下，我们同样可以得出结论，因为可以依次独立运行每个模式的检测算法。以下情况属于此类：

– 阈值图，其刻画模式为 Chordal 和余-Chordal；– 真区间图，其刻画模式为 Chordal 和镜像-Chordal；– 一分裂图，其刻画模式为 Split 和余-Split；– 增强团，其刻画模式为 Chordal 和 Split；– 二星，其刻画模式为余-Chordal 和 Forest。– 二部链图，其刻画模式为余-Chordal 和 Bipartite。

剩下的类中，每一种三顶点模式刻画都必须使用某个困难模式：Comparability、余-Comparability、Triangle 和余-Triangle。也就是说（去掉定理的例外情形，并忽略补图）：置换图、二部置换图、毛毛虫图、平凡完美图、无三角形图、 ∩ 余弦图以及完全二部图。

剩余情形 我们用五个命题证明剩余情形。

命题4。置换图和二部置换图可由一个能够在线性时间内检测的模式集合刻画。

证明。置换图可由一族两个模式来刻画： Com-parability 和 co-Comparability。而且该类在取补图后保持不变，因此我们不提及其补图。此外，只需考虑置换图的情形，因为对于二部置换图，只需添加二部模式即可，而它可以在线性时间内检测。

对于置换图，该命题已由 [18] 证明（尽管当时并未使用这些术语）；而在 [2], 中，同一结果被提出，作为

关于图搜索的一般框架的一项应用。其思路是处理给定的有序图，构建一个新图 G'，使得 $G = G'$ 当且仅当该排序同时避开 Comparability 和共-Comparability。更准确地说，深度优先搜索遍历（采用特定的平局处理规则）会产生一个关于 G 的新排序 θ。现在，$\tau\theta$ 定义了一个将 G 表示为置换图的方法；要得出结论，只需在线性时间内验证该表示是否对应于 G。□

命题 5. 完全二分图及其补图可由一个模式集合刻画，并且这些模式可以在线性时间内检测。

证明。可以用模式 co-Chordal、co-Comparability 和 Bipartite [9] 刻画这些图。模式 Bipartite 意味着该排序是两个顶点集 V_1、V_2 的串接，并且该图具有二分划分 (V_1, V_2)。随后，只需扫描边集，即可在线性时间内检查其是否为完全二分图。对补图类（两个不相交团）也可作类似论证。□

命题 6. 毛毛虫树可由一个模式集合刻画，并且这些模式可以在线性时间内检测。

证明。毛毛虫树可用模式 co-Comparability 和 Forest [9] 刻画。设 G 是一棵毛毛虫树，其脊柱是满足 x_1、x_k 不是悬挂顶点这一性质的极大路径 $[x_1, \ldots, x_k]$。众所周知，通过连续进行两次广度优先搜索即可找到这样的脊柱。设 τ 是 G 的一个不含这些模式的排序。考虑 $\forall i$、$1 \leq i \leq k-1$，不妨设 $x_i <_\tau x_{i+1}$；那么，根据 co-可比性模式，$[x_i, x_{i+1}]_\tau$ 中的所有顶点都必须与 x_i 或 x_{i+1} 相邻；但根据森林模式，它们只能与 x_i 相邻。因此，在这个区间内，x_i 可能有悬挂顶点（连接到 x_{i+1} 的悬挂顶点除外）。类似地，若 x_l 与 $l \neq i$、$i+1$ 属于该区间，则它必须与 x_i 和 $l = i-1$ 相邻，且不能有任何悬挂顶点，这就产生了矛盾。因此，τ 必然按从 x_1 到 x_k 的顺序排列脊柱；连接到 x_i 的悬挂顶点（$i > 1$）只能位于 x_i 和 x_{i+1} 之间。反过来，这样的排序不会出现模式 co-Comparability 和 Forest。我们称这种排序为非交错序，这一性质可以在线性时间内轻松检验。

对于补图类，我们无需计算补图，就能轻松检验这些性质（通过广度优先搜索寻找脊柱，以及检验交错性质）。

□

命题 7. 平凡完美图可由一组能在线性时间内检测的模式刻画。

证明。平凡完美图可以用模式 Triangle和 Chordal [9]来刻画。我们已经提出算法1，可在线性时间内检查 Chordal。对于模式 Triangle，利用平凡完美图的结构，我们只需检查：对于每个 x、y，使得 y 是相对于 τ 而言 x 的第一个邻居，$N'(x) = N'(y) \cup y$，其中 N' 表示 τ中 x 之后的邻域。这也可以在线性时间内完成。□

注意，平凡完美图也称为准阈值图 [28]，因为它们可以通过反复执行两种操作得到：不交并和添加一个全邻顶点。事实上，它们是树的可比图，也是余图和弦图的子类。根据 [1]，可以使用一种由顶点非递减度排序引导的特殊 LexBFS 图搜索来识别这些图。和通常一样，这种 LexBFS 会给出一个逆序 τ；当且仅当该图是平凡完美图时，此逆序才避开模式1和2。

命题 8. 无三角形余弦图可由一组能在线性时间内检测出的模式刻画。

证明。无三角形的 $\cap$ 余弦图可以由模式 Trian-gle 和余-Chordal [9]刻画。利用算法 2，我们可以检查余-Chordal。对于三角形，我们可以使用线性时间算法计算补图中的最大独立集（如 [20], 所述），并检查其大小是否至少为 3。□

注意，在线性时间内，可以在图的补图上执行一次 LexBFS，并生成一个避免模式 0 和 3 的排序，当且仅当该图是无三角形 $\cap$ 余弦图时。

A.5 寻找排序与验证排序

根据上述定理和讨论，我们得到以下推论。

推论 1. 对于 3 个元素上的任意禁用模式集，找到一种避开该模式集合的排序，比证明该排序避开这些模式更容易。

证明。 [9] 中的定理 5 为 22 个图类中的 20 个分别提供了在线性时间内生成避开该模式的排序的算法；此外，这些算法都基于经典的图搜索，并且都可以使用划分细化来实现 [13]，这意味着该结果同样适用于它们的补图类，因为划分细化对图及其补图同样适用，而无需构造补图。因此，对于 22 个图类中的 20 个，利用定理 2，也可以在线性时间内验证其排序。

最后，只需考虑单个模式：可比性、无三角形及其补图。对于无三角形模式及其补图，所有排序都是等价的，因此选取一种排序可以在线性时间内完成。但对于可比性及其补图，要在线性时间内构造一种避开该模式的排序则更为复杂，不过可以使用 [18]完成。□

对于包含超过 4 个顶点的模式，这样的陈述并不成立。我们回顾一下，经典的 Roy-Gallai-Hasse-Vitaver 定理指出，图 G 是

k可着色的，当且仅当它存在一种定向，使得其中不含有 (k+1) 个顶点的有向路径（即模式 P_k）。对于 $k = 4$，4 个顶点的路径模式对应于可 3 着色图这一类，而判定图是否属于该类是 NP 完全的。不过，这样的模式可以在线性时间内得到认证。

B 具有 $k \geq 4$个顶点的模式的条件下界

在本节中，我们将证明，对于包含至少四个节点的广大完全指定模式族，存在条件超线性下界。请注意，一个模式是完全指定的，当且仅当其中不存在未决定对。

我们的大多数结果都源自几种强大的归约技术。特别是，对于四个节点上的大多数完全指定模式，我们得到了条件二次下界（在对所考虑的算法类型施加额外限制时，则对所有此类模式都成立）。

在第4节中，我们证明了，在一般情况下，检测一个包含 k 个顶点的模式并不比在图中寻找一个 k-团更困难。大致而言，我们将在本节中证明：对于若干类完全指定的 k节点模式 P，从检测某个函数 f对应的 $f(k)$-团问题到该 P-Detection 问题还存在一个逆向归约。

我们在 B.1 节中正式介绍复杂性假设。然后，在 B.2 节和 B.3 节中，我们介绍两种不同类型的归约，并给出其在检测某些完全指定模式中的应用。在 B.4 节中，我们完成对具有四个节点的完全指定模式的 P-Detection 复杂性分类，其中包括 P。最后，我们将在 B.5 节中给出关于规模至少为五的模式的更多结果。

下文中，H基模式指其必选边诱导出 H的完全指定模式。下文中的精确结果和技术取决于 H。

B.1 复杂性假设

首先介绍下文所需的复杂性假设。关于这些假设的合理性及其在细粒度复杂度领域中的影响，我们参考 [5,16,26,27] 的详细讨论。

假设 1. 对于每个 $k \geq 3$，判断一个图是否包含一个大小为 k 的团（或独立集）需要 $n^{\omega(\lfloor k/3 \rfloor, \lceil k/3 \rceil, \lceil (k-1)/3 \rceil) - o(1)}$ 时间，其中 $\omega(p, q, r)$是分别具有$n^p \times n^q$ 和 $n^q \times n^r$维度的两个矩阵相乘时的指数。

假设 2。对于检测图中三角形的任何组合算法（不使用快速矩阵乘法或其他代数技术），都需要$n^{3-o(1)}$ 时间。

假设 3。判断一个 3-一致超图是否包含一个含有四个节点的超团，需要 $n^{4-o(1)}$ 时间。

在本节中，假设 1 常用于 $k = 3$ 或 $k = 4$。特别地，我们用 $\omega = \omega($1,1,1$)$ 简记方阵乘法的指数。请注意，文献中引用假设 1 时通常只针对团。这对 $k \geq 4$无关紧要，因为所有给出的下界都是超二次的，而且我们可以在 $\mathcal{O}(n^2)$ 时间内对图取补图。然而，在广泛认可的猜想 $\omega = 2$成立的前提下，这可能会成为 $k = 3$ 的问题。遵循 [5]，我们提出假设 1 同时适用于团和独立集。

我们注意到，假设 2 不如另外两个复杂性假设一般，因为它仅适用于组合算法。

B.2 从 Induced subgraph isomorphism出发的一般归约

回想一下， H-Induced-SI 问题是询问给定的宿主图 G是否包含 H 作为诱导子图。

命题 9。设 H 是一个具有 k 个顶点的图，其中 k 为某个常数。对于每个以 H 为基础的模式 P_H，都存在一个随机化的 $\mathcal{O}(n)$ 时间归约，将 H-Induced-SI 归约为 P_H-Detection。

证明。我们首先描述这个归约。设 G 是 H-Induced-SI 问题的一个实例。我们对 G 的顶点随机排列得到 τ，使用 Fisher-Yates 洗牌算法 [7,10] 可在 $\mathcal{O}(n)$ 时间内完成。然后，当且仅当 τ 包含模式 P_H 时，我们输出 G 是 H-Induced-SI 的一个是实例。

如果 τ 包含 P_H，那么任何实现该模式的有序子图的顶点集都会在 G 中诱导出 H 的一个副本。特别地， G 是 H-Induced-SI 的一个是实例（等价地说，我们的归约绝不会接受否实例）。反过来，假设 G 中存在 H 的一个诱导副本，其顶点集为 S。我们对 S 的顶点作全序排列，使得到的有序图实现 P_H。在对 V 的随机排列中， S 的顶点恰好按此顺序出现的概率为 $1/(k!)$。因此，对于 H-Induced-SI 的一个是实例，我们以至多 $1 - 1/(k!)$ 的概率拒绝。□

我们指出，对于任何一族基于 H 的模式 $\mathcal{F}_H$，命题 9 也更一般地成立。

让我们说明命题 9 在线四节点模式中的应用。共有十一种 4 顶点图，即: K_4、 C_4、菱形图、爪图、爪图、这五种图各自的补图，以及路径 P_4。对于所有此类图 H 但 P_4，目前尚无用于 H-Induced-SI 的已知线性时间算法。因此，若存在一种线性时间算法能够验证任何基于 H 的模式，这将是一项重大的算法突破。现在，在假设 1、2 或 3 中任一项成立的前提下，我们给出更精确的下界（也适用于一些规模大于四的模式）。

推论 2。假设 $k \geq 3$ 的假设 1 成立，若 P 是唯一的基于 K_k 的模式（分别地，基于 $\overline{K_k}$ 的模式），则每个 P-Detection 算法都需要$n^{\omega(\lfloor k/3 \rfloor, \lceil k/3 \rceil, \lceil (k-1)/3 \rceil) - o(1)}$ 时间。

注意，只有一个基于 K_k 的模式，并且按照顶点的任意顺序在其中进行搜索，等价于判断一个图是否包含一个由 k 个顶点组成的团。因此，对于该模式而言， $n^{\omega(\lfloor k/3 \rfloor, \lceil k/3 \rceil, \lceil (k-1)/3 \rceil) + o(1)}$ 的运行时间在条件上是紧的。对于唯一的基于 $\overline{K_k}$ 的模式，情况同样如此。

已知，对于每个包含由 k 个顶点组成的团的 H， H-Induced-SI 问题至少与 k-Clique Detection [5] 一样困难。因此，

推论 3. 假设关于 $k \geq 3$ 的假设 1 成立，如果 H 是一个包含由 k 个顶点组成的团的图，那么对于每个基于 H 的模式 P_H， P_H-Detection 至少需要 $n^{\omega(\lfloor k/3 \rfloor, \lceil k/3 \rceil, \lceil (k-1)/3 \rceil) - o(1)}$ 的时间。

以下图 H 都包含三角形：菱形图、爪图和 $K_3 + K_1$ （也称为爪图的补图）。因此：

推论 4. 令 H 为菱形图、爪图或 $K_3 + K_1$。假设对于 $k = 3$，假设 1 成立，则对于每个基于 H 的模式 P_H， P_H-Detection 至少需要 $n^{\omega - o(1)}$ 时间。

将输入图取补图所需时间为 $\mathcal{O}(n^2)$，据此， [5] 中的结果也适用于带有一个含 k 个顶点的独立集的图 H，对每个 $k \geq 4$ 均成立。对于以 H 为基础的模式，我们的推论 3 也同样成立。然而，我们不知道是否有类似结果适用于带有一个含三个顶点的独立集的图 H。接下来我们证明，若 H 是一个四顶点图，则这样的结果成立。

引理 1. 令 H 为 $K_2 + 2K_1$、反爪 paw 图或爪图。从在一个有 n 个顶点的图中检测大小为三个的独立集这一问题，可以在线性时间内归约到在有 $\mathcal{O}(n)$ 个顶点的图上求解 H-Induced-SI 问题。

证明。设 $G = (V, E)$ 是任意一个无向图。我们按如下方式将 G 转换为一个更大的图 G'：

– 情形 H 是爪图。我们添加一个全邻顶点，即该顶点与 V的每个顶点都相邻。– 情形 H 等于 $K_2 + 2K_1$。我们构造 G 的两个不相交副本 G_1、 G_2，将 v_1、 v_2 称为同一个顶点 $v \in V$的两个副本。对于每个 $v \in V$，我们添加一条边 v_1v_2。最后，对于每条边 $uv \in E$，我们添加边 u_1v_2、 v_1u_2。如果 x、 y、 z 构成 G的一个独立集，那么 x_1、 y_1、 z_1、 x_2 在 G'中诱导出 H 的一个副本。反之，假设 G' 包含 H的一个诱导副本。特别地， G' 也包含一个大小为三的独立集。不失一般性地，至少有两个顶点属于 V_1。将它们记为 x_1、 y_1。根据构造， G 的顶点 x、 y 不相邻。设 z_i 为该独立集的第三个顶点，其中 $i \in \{1, 2\}$。如果 $i = 1$，那么显然 x、 y、 z构成一个

G的独立集。现在假设 $i = 2$。那么， $z \notin \{x, y\}$ 和 z都与 G 中的 x、 y 不相邻（否则， G'中将存在 x_1z_2、 y_1z_2 这两条边中的一条）。因此，在此子情形下，我们同样有 x、 y、 z构成 G的一个独立集。

– 情形 H 是反爪 paw 图。我们首先计算 G 的模分解，这可以在线性时间内完成 [24]。根据 Gallai 定理 [12], , G 要么是不连通的，要么其补图不连通，要么其商图在模分解中是素的。假设 G 不连通。若它至少有三个连通分量，则其中包含一个由三个顶点构成的独立集。否则， G 是无爪垫图，当且仅当它的两个连通分量都是团。从现在起，我们假设 G 是连通的。假设 $\overline{G}$ 不连通。那么， G 是无爪垫图，当且仅当每个补图连通分量都是I_3。因此，我们只需分别考虑每个补图连通分量。从现在起，我们假设 G 既连通又补图连通。令 G' 为其商图。 G' 的每个顶点 x 对应于 G 的某个强模 M_x。如果这样的模 M_x 不是团，那么 G 中存在一个独立集，它由 M_x 中两个不相邻的顶点以及某个与 M_x 不相邻模中的任意顶点组成（这样的不相邻模必定存在，因为 G' 是素图）。因此，我们现在假设，对于G'的每个顶点 x， G 的模 M_x 都是团。那么， G 是无爪垫图，当且仅当 I_3 是无爪垫图。Olariu 证明了，图 $\overline{G'}$ 是无爪垫图，当且仅当它的每个连通分量都是无三角形图或完全多部图 [19]。回想一下，在我们的情形中， $\overline{G'}$ 是连通的。由于 G' 也连通，其补图$\overline{G'}$ 不是完全多部图。因此， G' 是无爪垫图，当且仅当它不包含诱导补爪图。

总的来说，在这三种情况下，所得的图 G' 包含一个 H的诱导副本，当且仅当 G 包含一个由三个顶点组成的独立集。□

推论 5. 令 H 为 $K_2 + 2K_1$、反爪 paw 图或爪图之一。假设关于 $k = 3$的假设 1成立，对于每个基于 H的模式 P_H， P_H-Detection 至少需要 $n^{\omega - o(1)}$ 的时间。

现在我们还需考虑以下具有四个节点的图 H 的基于 H的模式： C_4、 $2K_2$ 和 P_4。最近已有研究证明，假设 3成立时，即使对于具有$\mathcal{O}$（$n^{1.5}$）条边的图 [6], C_4-Induced-SI 问题也至少需要 $n^{2-o(1)}$ 的时间。因此：

推论 6. 假设 3成立，对于每个基于 C_4的模式 P，每个 P-Detection 算法至少需要 $n^{2-o(1)}$ 的时间，即使输入图具有$\mathcal{O}$（$n^{1.5}$）条边。

B.3 拓扑排序

给定一个基于 H 的模式 P_H，对于任意的 H，我们可以将每条边从较小的端点指向较大的端点。这样便得到一个定向

H，其中该模式表示一个拓扑排序。我们称之为与 P_H 相关联的 H 的定向，记为 $\overrightarrow{Or}(P_H)$。一个广为人知的结果是：有向无环图当且仅当包含一条有向哈密顿路径时，才存在唯一的拓扑排序。接下来，我们证明检测完全指定模式的复杂度与其关联定向中存在某些有向路径之间的一般关系。

命题 10。设 P 是一个大小为常数的完全指定模式。若 $\overrightarrow{Or}(P)$包含一条经过 k 个顶点的唯一有向路径，其中 k为某个常数，则存在一个 $\mathcal{O}(n^2)$时间的归约，将 k顶点图上的 Clique Detection - n归约为P(Detection)顶点有序图上的 $\mathcal{O}$-n。

特别地，若 $k \geq 4$ ，则根据假设 1，每个 P-Detection算法都需要至少 $n^{\omega(\lfloor k/3 \rfloor, \lceil k/3 \rceil, \lceil (k-1)/3 \rceil) - o(1)}$ 时间。若 $k \geq 3$ ，则根据假设 2，每个组合式 P-Detection 算法都需要至少 $\mathcal{O}(n^{3-o(1)})$ 时间。

证明。设 $G = (V, E)$ 是 k-Clique Detection的一个实例。我们将 $V(P)$ 划分为 $\{v^1, v^2, \ldots, v^k\} \cup \{u^1, u^2, \ldots, u^p\}$，使得顶点 v^1、v^2、.....、v^k 位于与 P相关联的定向中的唯一有向 k路径上。设 Q 是由 v^1、v^2、.....、v^k诱导的 P的k节点子模式。

我们首先按如下方式构造一个中间有序图 G_Q：

1. $V(G_Q) = V_1 \cup V_2 \cup \ldots \cup V_k$，使得每个 V_i 都是 V的一个不相交副本。对于每个 $v \in V$，我们用 v_i 表示它在 V_i中的副本。 2. V_1、V_2、.....、V_k 是独立集。 3. 接着，令 i、j 满足 $1 \leq i < j \leq k$。如果 v^i 和 v^j 相邻（当 $j = i+1$ 时，这总是成立），则对于每一对相邻顶点w、$x \in V$，我们添加边 $w_i x_j$。否则，对于每一对相等或不相邻的顶点 w、$x \in V$，我们添加边 $w_i x_j$。 4. 最后，我们为 G_Q 赋予一个全序 τ_Q ，使得：对于每个 $1 \leq i <j \leq k$， V_i 中的所有顶点都必须排在 V_j中的每个顶点之前。

该构造可以在 $\mathcal{O}((kn)^2)$ 时间内完成。

首先假设 x^1、x^2、.....、x^k 构成 G 的完全子图。那么，由 x_1^1、x_2^2、.....、x_k^k 在 G_Q 中诱导出的有序子图实现了模式 Q。反过来，假设存在一个实现 Q 的 G_Q 诱导有序子图。由于该子图中的每两个连续顶点都必须相邻，因此其 k 个顶点按顺序必定是某些 $x_1^1 \in V_1$、$x_2^2 \in V_2$、.....、$x_k^k \in V_k$。根据构造，x_1、x_2、.....、x_k 构成 G 的完全子图。

我们按照如下方式从 G_Q 构造 G_P：

1. 我们添加新顶点 $y^1, y^2, \ldots, y^p$。 2. 对于每个 $1 \leq i < j \leq p$， y^i 和 y^j 之间有一条边，当且仅当 u^i 和 u^j 在 P 中相邻。对于每个 $1 \leq i \leq p$ 和每个 $1 \leq j \leq k$，当且仅当 u^i 和 v^j 在 P 中相邻时，我们在 y^i 与 V_j 的每个顶点之间添加边。

3. 我们为 G_P 配备一个全序 τ_P，其在 G_Q 上的限制等于 τ_Q。对于每个 $1 \leq i < j \leq p$，$\tau_P(y^i) < \tau_P(y^j)$。对于每个 $1 \leq i \leq p$ 和每个$1 \leq j \leq k$，当且仅当 u^i 在 P 中排在 v^j 之前（之后）时，y^i 必须排在 V_j 的每个顶点之前（之后）。

回想一下，如果 G 有一个 k团，那么 G_Q 中存在一个实现 Q的有序子图。根据构造，我们可以用 y^1、y^2、.....、y^p 中的一些顶点扩展后者，从而得到 G_P中实现 P的有序子图。反过来，假设存在 G_P 中实现P的有序子图。令 z^1、z^2、.....、z^k 为该有序子图中实现 Q的 k 顶点。注意，对于每个 $1 \leq j \leq k$，必有 $\#V_j \cap \{z^1, z^2, \ldots, z^k\} \leq 1$，因为该子集中的顶点必须在 k路径上连续出现，而 V_j 是一个独立集。接着，令 a^1、a^2、.....、a^k 满足：

$$\begin{cases} a^i = v^j \text{ if } z^i \in V_j \\ a^i = u^j \text{ if } z^i = y^j \end{cases}$$

对于每个满足 $a^i = v^j$ 和 $a^{i+1} = v^{j+q}$ 的 a^i、a^{i+1}（其中某个 $q \geq 2$），我们还在 a^i和 a^{i+1}之间添加序列 v^{j+1}、v^{j+2}、.....、v^{j+q-1}。这样，我们就在 $\overrightarrow{Or}$（P）中得到了一条至少包含 k个顶点的有向路径。于是，对于每个 $1 \leq i \leq k$，$a^i = v^i$。这意味着 z^1、z^2、.....、$z^k \in V(G_Q)$。因此，G包含一个k团。□

注意，如果模式 P包含一个有 k 个节点的团，那么在 $\overrightarrow{Or}(P)$ 中存在一条包含 k 个顶点的有向路径。然而，即使不存在 k-团，也可能存在这样的有向路径。例如，如果 H 表示菱形图、爪图或 C_4，并且我们假设假设 1，那么根据命题 10，对于每个基于 H 的模式 P_H（其中顶点按某条哈密顿路径排序），P-Detection 问题需要 $n^{\omega(2,1,1)-o(1)}$ 时间。后者分别改进了推论4和6。

我们展示命题 10 在一些基于 P_4 的模式中的另一项应用。回顾一下，我们可以在线性时间内识别无 P_4 图（即余图）[4]。因此，对于基于 P_4 的模式，无法从命题 9 得出有意义的下界。本小节和 B.4 节将证明，即使对于基于 P_4 的模式，P-Detection 也不太可能存在次二次时间算法。P_4 有四种定向，每种都与一个或多个基于 P_4 的模式相关联。这四种定向列于图 8 中。本小节讨论除 $\overrightarrow{P}$(1,1,1) 之外的所有这些定向。

推论 7. 对于 $k = 4$，假设假设 1；与 $\overrightarrow{P_4}$ 相关联的唯一模式 P，P-Detection 需要 $n^{\omega(2,1,1)-o(1)}$ 时间。
证明. 该结果可由命题 10 以及 $\overrightarrow{P_4}$ 中存在唯一的有向 4-路径得出。□

推论 8. 在假设 2 下，对于每个基于 P_4 的模式 P（与 $\overrightarrow{P}$(2, 1) 或 $\overrightarrow{P}$(1, 2) 相关联），每种组合式 P-Detection 算法都至少需要 $n^{3-o(1)}$ 时间。

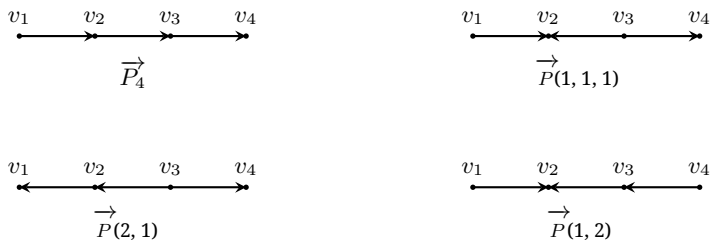


图 8： P_4 的四种定向。

证明。该结论由命题 10 以及 $\overrightarrow{P}$(2,1) 和 $\overrightarrow{P}$(1,2) 中存在唯一有向 3-路径得出。□

最后，模式 P 的补图 $\overline{P}$ 是通过将 P 的每条必选（分别为禁止）边替换为一条禁止（分别为必选）边而得到的。如果图 G 上的排序 τ 避免了 P，那么将 τ 视为 $\overline{G}$ 上的排序时，它也会避免 $\overline{P}$。这意味着存在一个平凡的 $\mathcal{O}(n^2)$ 时间归约，可将 n 个顶点的有序图上的 P-Detection 归约为 n 个顶点的有序图上的 $\overline{P}$-Detection 。特别地，命题 10 对满足以下条件的模式也成立：在 $\overrightarrow{Or}$（$\overline{P}$）中存在唯一的有向 k-路径。

B.4 四顶点模式

我们通过给出每个完全指定的四节点模式的条件下界，补全 B.2 节和 B.3 节。令 H 为任意一个四顶点图。

$H = K_4$ 或 $H = \overline{K_4}$ 的情形已由推论 2 涵盖。

H 为菱形图、爪图或 $K_3 + K_1$ 的情形已由推论 4 涵盖。H 为 K_2+$2K_1$、反爪 paw 图或爪图的情形已由推论 5 涵盖。

$H = C_4$ 的情形已由推论 6 涵盖。现在我们讨论特殊的 $2K_2$ 基模式情形。这类模式共有三种，见图 9。

命题 11。令 P 为某个 $2K_2$ 基模式。

– 若 $P = Q_3(2K_2)$，则在假设 $k = 4$，P-Detection 满足假设 1 的情况下，至少需要 $n^{\omega(2,1,1)-o(1)}$ 时间。– 否则，在假设 2 成立的情况下，每种组合式 P-Detection 算法至少需要 $n^{3-o(1)}$ 时间。

证明。若 $P = Q_3(2K_2)$，则 $\overrightarrow{Or}(\overline{P})$ 中存在唯一的有向 4-路径，结论由命题 10 得出。否则，结论分别由引理 2 和引理 3 得出。□

接下来给出的归约对于所考虑的两种模式都类似。

引理 2. 存在一个 $\mathcal{O}(n^2)$ 时间归约，将 Triangle Detection 归约到$Q_1(2K_2)$-Detection ，作用于一个有序图，该图有 $\mathcal{O}(n)$ 个顶点。

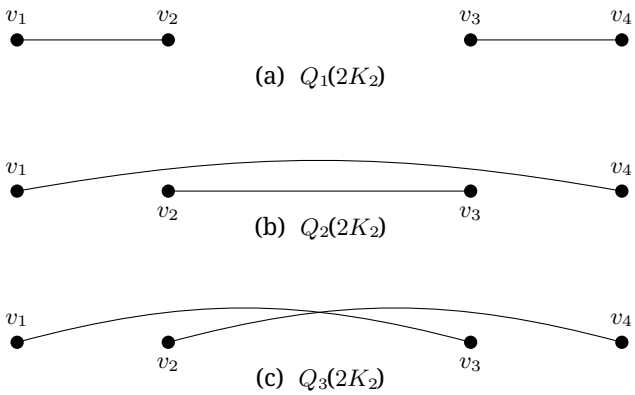


图 9：所有基于 $2K_2$ 的模式（为便于绘图，省略了禁用边）。

证明。设 $G = (V, E)$ 是 Triangle Detection 的一个实例。我们按如下方式构造一个有序图 G'：

1. $V(G') = \{u\} \cup V_1 \cup V_2 \cup V_3$，使得每个 V_i 都是 V 的一个不相交副本。对于每个 $v \in V$，我们将它在 V_i 中的副本记为 v_i。2. $N_{G'}(u) = V_1$。3. V_1 是一个团，且 V_2、V_3 是独立集。4. 对于每条边 $vw \in E$，我们添加一条边 v_2w_3。5. 对于每个顶点 $v \in V$，我们添加边 v_1v_2、v_1v_3。对于每个满足 v、w 不相邻的 $w \in V \setminus \{v\}$，我们也添加边 v_1w_2、v_1w_3。6. 最后，我们为 G' 赋予一个全序 τ'，满足：顶点 u 排在最前面，后面依次是 V_1 的所有顶点、V_2 的所有顶点，然后是 V_3 的所有顶点。

该构造可在 $\mathcal{O}(n^2)$ 时间内完成。

首先假设 G 中存在某个三角形 xyz。那么，由 u, x_1, y_2, z_3 诱导的 G' 的有序子图实现了 $Q_1(2K_2)$。反过来，假设存在一个有序子图 T'，实现了 $Q_1(2K_2)$。由于 $\{u\} \cup V_1$ 诱导出一个团，T' 中最后两个顶点（相对于排序 τ'）必定属于 $V_2 \cup V_3$。特别地，这两个顶点必定是 y_2, z_3，其中 y, z 是 G 中相邻的顶点。此外，由于 V_2 的所有顶点都排在 V_3 的所有顶点之前，T' 中最前面的两个顶点不可能也都属于 $V_2 \cup V_3$。因此，至少有一个这样的顶点必须是 x_1，其中 x 是 G 中的某个顶点。由于我们假设 T' 实现了 $Q_1(2K_2)$，顶点 x_1 不能与 y_2 或 z_3 相邻。于是，在 G 中，根据构造，顶点 x 必须同时与 y, z 相邻。因此，G 中存在一个三角形。□

引理 3. 存在一个从 Triangle Detection 到 $Q_2(2K_2)$-Detection 的 $\mathcal{O}(n^2)$ 时间归约，该归约作用于一个有 $\mathcal{O}(n)$ 个顶点的有序图。

证明。设 $G = (V, E)$ 是 Triangle Detection 的一个实例。我们按如下方式构造有序图 G'：

1. $V(G') = V_1 \cup V_2 \cup V_3 \cup V_4$，使得每个 V_i 都是 V 的一个不相交副本。对于每个 $v \in V$，我们将其在 V_i 中的副本记为 v_i。2. V_1, V_4 是团，而 V_2, V_3 是独立集。3. 对于每个顶点 $v \in V$，我们添加边 v_1v_2, v_2v_3, v_3v_4。对于每个顶点$w \in V \setminus \{v\}$，若 v, w 相邻，则添加边 v_1w_4；若 v, w 不相邻，则添加边 v_1w_2, v_3w_4。4. 最后，设 τ 是 V 的任意全序，并令 $\tau_1, \tau_2, \tau_3, \tau_4$ 分别表示该排序在 V_1, V_2, V_3, V_4 中的副本。我们为 G' 配备全序 τ'，它就是 $\tau_1, \tau_2, \tau_3, \tau_4$ 的串接。

该构造可在 $\mathcal{O}(n^2)$ 时间内完成。

首先，假设 xyz in G 中存在某个三角形。于是，由 x_1, y_2, y_3, z_4 诱导的 G' 的有序子图实现了 $Q_2(2K_2)$。反过来，假设存在一个实现 $Q_2(2K_2)$ 的有序子图 T'。接下来分两种情况讨论。

– 情况 T' 包含 V_1 中的一个顶点。特别地，T' 的第一个顶点必定是某个顶点 $x_1 \in V_1$。由于 V_1 是一个团，且 x_1 必须与 T' 的第二个顶点不相邻，因此 T' 的另外三个顶点都必须属于 $V_2 \cup V_3 \cup V_4$。由于 V_2 是一个独立集，并且 T' 的第二个和第三个顶点之间有一条边，因此 T' 的最后三个顶点不能全都包含在 V_2 中。特别地，T' 的最后一个顶点必须包含在 $V_3 \cup V_4$ 中。由于 V_1 与 V_3 之间没有边，且 x_1 必须与 T' 的最后一个顶点相邻，因此这个最后的顶点必须是某个 $z_4 \in V_4$。由于 V_2, V_3 都是独立集，这意味着 T' 的第二个和第三个顶点分别是某个 $y_2 \in V_2$ 和 $y_3' \in V_3$。由于 y_2, y_3' 相邻，因此必有 $y = y'$。由于 x_1, z_4 相邻，因此必有 $xz \in E$。最后，由于 x_1, y_2（分别地，y_3, z_4）不相邻，因此必有 $xy \in E$（分别地，$yz \in E$）。由此，xyz 是 G 的一个三角形。

– 否则，T' 的所有四个顶点都必须包含在 $V_2 \cup V_3 \cup V_4$中。由于 V_2 的所有顶点都排在 V_3的所有顶点之前，且 V_2的全排序τ_2、τ_3相同，因此 T'的所有顶点不可能都包含在 $V_2 \cup V_3$中（否则，T' 不可能实现 Q_2（$2K_2$））。特别地，T' 的最后一个顶点必定是某个 $z_4 \in V_4$。回想一下，z_4 必须与 T'的第三个顶点不相邻，因此，T'的其余三个顶点都必须包含在 $V_2 \cup V_3$中。此外，由于 V_2 与 V_4之间没有边，T'的第一个顶点不能包含在 V_2中。因此，T'的其余三个顶点都必须包含在 V_3中。然而，T' 的第二个和第三个顶点必须相邻，这与 V_3 是一个独立集相矛盾。

因此，只可能出现第一种情况，这意味着 G 中存在某个三角形。□

最后，我们考虑基于 P_4 的模式 Q 的奇数情形。情形 $\overrightarrow{Or}(Q) = \overrightarrow{P_4}$ 已在推论 7 中讨论。情形 $\overrightarrow{Or}\ Q = \overrightarrow{P}\ 2,1$ 和 $\overrightarrow{Or}\ Q = \overrightarrow{P}\ 1,2$ 已在推论 8 中讨论。因此，剩下需要考虑的是情形 $\overrightarrow{Or}(Q) = \overrightarrow{P}$ (1, 1, 1)。后一种定向有五种不同的拓扑排序（因此，

与之对应的模式有五种）。下文中，模式节点的标签采用图 8 中给出的标记。

命题 12。设 Q 是一个基于 P_4 的模式，该模式与 $\overrightarrow{P}(1, 1, 1)$ 相关联。

1. 如果 Q 的顶点按 v_3、v_1、v_4、v_2 排序，那么在假设 1（针对 $k = 4$）成立的条件下，每个 Q-Detection 算法都需要 $n^{\omega(2,1,1)-o(1)}$ 时间。2. 否则，在假设 2 成立的条件下，每个用于 Q-Detection 的组合算法至少需要 $n^{3-o(1)}$ 时间。

证明。在下文中，设 $G = (V, E)$ 是一个任意的图。需要考虑若干种情形。每种情形取决于 Q 中四个顶点的全序 $v_{\sigma(1)}$、$v_{\sigma(2)}$、$v_{\sigma(3)}$、$v_{\sigma(4)}$。

情形 1：$\sigma = (3, 1, 4, 2)$。在 $\overrightarrow{Or}(\overline{Q})$ 中存在唯一的有向 4-路径。因此，可以应用命题 10。

情形 2：$\sigma = (3, 4, 1, 2)$。我们构造一个有序图 G'，使得：

1. $V(G') = V_1 \cup V_2 \cup V_3 \cup V_4$，其中每个 V_i 都是 V的不相交副本。对于每个顶点 $v \in V$，令 v_i 表示其在 V_i中的副本。2. 子集 V_3、V_2 是独立集，而子集 V_4、V_1 是团。3. 对于任意不同的顶点 v、$w \in V$，我们添加边 v_4w_1；如果 $vw \in E$，则添加边 v_3w_4、v_3w_2、v_1w_2。4. 最后，我们为 G' 配备一个全序 τ'，使得：V_3中的顶点最先出现，随后是 V_4中的所有顶点、V_1中的所有顶点，最后是 V_2中的所有顶点。

该构造耗时 $\mathcal{O}(n^2)$。此外，如果 xyz 是 G的一个三角形，那么由 x_3、y_4、y_1、z_2 诱导的有序子图实现了 Q。反过来，假设存在一个实现 Q的诱导有序子图 T。如果每个顶点分别属于 V_1、V_2、V_3、V_4中的不同子集，那么令 $V(T) = (x_3$、y_4、y'_1、$z_2)$。根据构造，$y = y'$ 和 xyz 是 G的一个三角形。因此，假设 T 中有两个顶点属于同一个子集，从而导致矛盾。这两个顶点不可能是 T的第二和第三个顶点，因为后两者不相邻，而只有 V_3、V_2 是独立集。因此，它们必定是 T的第一个和第二个顶点，或其第三个和第四个顶点（这两种情况不一定互斥）。于是只有两种可能：T 的前两个顶点属于 V_4，且后两个顶点中至少有一个属于 V_1；或者前两个顶点中至少有一个属于 V_4，而后两个顶点属于V_1。然而，如果第一种情况发生，那么 T 的第三个顶点（属于 V_1）在 V_4中有两个不相邻的顶点；如果第二种情况发生，那么 T 的第二个顶点（属于 V_4）在 V_1 中有两个不相邻的顶点。矛盾。

情形 3：$\sigma = (1, 3, 2, 4)$。我们构造一个有序图 G'，使得：

1. $V(G') = V_1 \cup V_2 \cup V_3 \cup V_4$，其中每个 V_i 都是 V 的一个不相交副本。对于每个顶点 $v \in V$，令 v_i 表示它在 V_i 中的副本。2. V_1、V_4 是团，而 V_2、V_3 是独立集。

3. 对于每个顶点 $v \in V$，我们添加一条边 v_2v_3。4. 接着，我们依次考虑每一对 v、$w \in V$。如果 v、w 相等或不相邻，我们就添加边 v_1w_4。如果 v、w 相邻，我们就添加边 v_1v_2、v_3v_4。5. 最后，我们为 G' 配备一个全序 τ'，使得：V_1 的顶点排在最前面，其后是 V_3 的所有顶点、V_2 的所有顶点，然后是 V_4 的所有顶点。

该构造耗时 $\mathcal{O}(n^2)$。此外，如果 xyz 是 G的一个三角形，那么由 x_1、y_3、y_2、z_4 诱导的有序子图实现了 Q。反之，假设存在一个实现 Q的诱导有序子图 T。与前一种情况一样，可以很容易地验证：如果 T 的所有顶点分别属于 V_1、V_2、V_3、V_4中的不同子集，那么 G中就存在一个三角形。因此，反设 T中有两个顶点属于同一个子集 V_i。它们不可能是第二和第三个顶点，因为后两者相邻，而只有 V_1、V_4 是团。因此，它们必须是 T的第一个和第二个顶点，或第三个和第四个顶点（这两种情况不一定互斥）。在这两种情况下，这两个连续顶点都不相邻。于是只有两种可能：T 的前两个顶点属于 V_3，且后两个顶点中至少有一个属于 V_2；或者前两个顶点中至少有一个属于 V_3，而后两个顶点属于 V_2。特别地，T的第二和第三个顶点是某个 $y \in V$的 y_3、y_2。然而，T的前两个顶点不可能都属于 V_3，因为第三个顶点必须与二者都相邻。同样地，T 的后两个顶点不可能都属于 V_2，因为第二个顶点与二者都相邻。矛盾。

情形 4：$\sigma = (1, 3, 4, 2)$。我们构造一个有序图 G'，使得：

1. $V(G') = V_1 \cup V_2 \cup V_3 \cup V_4$，其中每个 V_i 都是 V 的一个不相交副本。对于每个顶点 $v \in V$，令 v_i 表示它在 V_i 中的副本。2. 子集 V_1、V_2、V_3、V_4 都是团。3. 对于每个 $v \in V$，我们添加边 v_3v_4。4. 对于每对顶点 v、$w \in V$，若 v、w 相邻，则添加边 v_1v_2、v_3v_2；若 v、w 相等或不相邻，则添加边 v_1v_3、v_1v_4。5. 最后，我们为 G' 配备一个全序 τ'，使得：V_1 的顶点排在最前面，接着是 V_3 的所有顶点、V_4 的所有顶点，最后是 V_2 的所有顶点。

该构造需要 $\mathcal{O}(n^2)$ 时间。此外，如果 xyz 是 G的一个三角形，那么由 x_1、y_3、y_4、z_2 诱导的有序子图实现了 Q。反之，假设存在一个诱导有序子图 T，它实现了 Q。与前两种情况一样，可以很容易地验证，如果 T 的所有顶点分别属于 V_1、V_2、V_3、V_4中的不同子集，那么 G中就存在一个三角形。因此，从现在起，我们假设 T 中存在两个顶点属于同一个子集V_i。由于 V_i 是一个团，这两个顶点必须是 T的第二个和第三个顶点。于是，这两个顶点要么位于 V_3 中，要么位于 V_4中。它们不可能包含在 V_4 中，因为 T的第二个顶点和最后一个顶点相邻，而V_4、V_2之间没有边。因此，T的第二个和第三个顶点分别是某个 y_3、$z_3 \in V_3$。这意味着

T 的第一个顶点是某个 $x_1 \in V_1$。反设 T 的最后一个顶点是某个 $t_4 \in V_4$。由于 z_3、t_4 在 G'中相邻，根据构造可得 $z = t$。由于 x_1、t_4 在 G'中相邻，根据构造，x、z 在 G中要么相等，要么不相邻。但这样一来，顶点 x_1、z_3 就应当在 G'中相邻，产生矛盾。因此，T 的最后一个顶点是某个 $t_2 \in V_2$。G' 中存在边 x_1t_2、y_3t_2 意味着，t 在 G中与 x、y 都相邻。G' 中不存在边 x_1y_3 也意味着，x、y 在 G中相邻。因此，G中存在一个三角形 xyt 。

情形 5：$\sigma = (3, 1, 2, 4)$。这是前一种情形的镜像模式。 □

B.5 大小大于四的模式

最后，我们介绍另一种通用的模式间归约类型。模式 P 的前缀（分别为后缀）是由其 i 个首顶点（分别为其 i 个末顶点）诱导的子模式，其中 $i \leq |V(P)|$。

命题 13. 设某个模式 P 是某个大小为常数 $k+1$ 的较大模式 Q 的前缀（分别为后缀）。那么，Q-Detection 至少与 P-Detection 一样难（在随机化 $\mathcal{O}(n)$ 时间归约下）。

证明。由对称性，只需考虑 P 是 Q 的前缀的情形。此外，只需考虑特殊情形 $|V(Q)| = |V(P)|+1$。设 G 是任意一个有序图。有序图 G' 由 G 构造而成：添加一个新顶点 x，使其在排序中位于最后；并且对于 G 的每个顶点 v，我们以概率 $1/2$ 独立地添加一条边 vx 。

首先，假设存在一个有序子图 H in G，它实现了 P。令 H' 由 $V(H) \cup \{x\}$ 诱导得到。为了使 H' 实现 Q，只需使顶点x 与 H中的某些顶点满足一些邻接关系或非邻接关系。根据构造，满足所有这些关系的概率至少为 $1/2^k$。反之，假设存在一个有序子图 H' in G'，它实现了Q。由于顶点 x 只能在 H' 中作为其最后一个顶点出现，因此存在 H'的某个有序子图 H，其中所有顶点均在 $V(G)$ 中，该子图必须实现 P。□

对于每个 $k \geq 5$，上述结果结合我们针对大小为四的完全指定模式所作的所有归约（见 B.4 节），意味着对于一大类大小为 k 的完全指定模式，都存在条件超二次下界（如果我们将范围限制为组合算法，则对所有这些模式均成立）。对于满足命题 9 或命题 10 之一可直接应用的每个模式，还可以得到更强的下界。

C 参数化算法与合并宽度

在本节中，禁用边与强制边的作用完全相同，因此为简便起见，我们统称为“边”，并将模式 P 的边集记为 $E(P)$（即 $E(P) = M(P) \cup F(P)$）。在可能产生歧义的地方，我们会特别说明。

C.1 锚定模式操作

定义 4. 锚定模式是指其中一个顶点子集被标记为锚点的模式。

现在，我们将定义在模式上考虑的操作。实际上，每种操作都有两种对偶形式：一种将模式分解为更小的部分，另一种将更小的部分组合成更大的部分。我们将两者都定义出来，因为一方面分解操作很自然，另一方面我们的自底向上算法将使用组合操作。

定义 5. 我们将锚定模式上的以下操作定义为模式操作。

1.（分割/合并）给定锚定模式 $P = (V, E)$，划分操作将其拆分为两个模式 $P_1 = (V_1, E_1)$ 和 $P_2 = (V_2, E_2)$，并满足以下约束：(a) (E_1, E_2) 是 E 的一个划分（顶点集继承边的端点，即对于 $i = 1$，2, $\forall (x, y) \in E_i$, $x, y \in V_i$）。(b) $V_1 \cup V_2 = V$(c) $V_1 \cap V_2$ 中的任何顶点都是 V_1 和 V_2 中的锚点。(d) P 中原本作为锚点的任何顶点，在接收该顶点的模式中仍然是锚点。(e) V 中不能有两个连续顶点，一个仅属于 V_1，另一个仅属于 V_2。反向操作称为合并操作（只有当其反向操作是正确的划分操作时，合并才是正确的）。2.（顶点删除/创建）如果一个顶点在模式中是孤立的，顶点删除操作将其移除。操作完成后，紧邻该顶点之前和之后的顶点（如果存在）必须是锚点。反向操作称为顶点创建。3.（边删除/创建）如果模式仅由一条边构成，边删除操作将其移除（包括两个端点）。反向操作称为边创建。（这里实际上需要分两种情况：一种针对必选边，另一种针对禁用边。）

对于分割/合并操作，同时存在于两个模式中的锚点称为活动锚点，其余锚点称为被动锚点。直观来说，活动锚点是当前合并操作所用到的锚点，而被动锚点仅在流程的其他步骤中才会用到。

一个重要的观察是，当我们执行合并时，原来两个模式中的一个 ACTIVE 锚点可能不会成为合并模式中的锚定模式。例如，对于上文所述的平坦环，在合并 1-2 和 2-3 时，顶点 2 失去了其锚定状态。当一个顶点对当前合并有用、但之后不再有用时，就会发生这种情况。由于减少锚点数量对我们很有用，因此这是一个重要方面。顶点创建也同样如此，其结果中的顶点可能会失去锚定状态。

约束 1e 是该定义的关键部分，我们将在接下来的几段中对此进行论证。现在，让我们证明以下引理，其中“创建一个模式”是指创建该模式的某个锚定版本，然后移除锚点以获得一个标准模式。

引理 4. 任意模式都可以通过边创建操作、顶点创建操作和合并操作来创建。

证明。考虑一个任意的模式 P。我们将说明如何构造它。示例见图 10。请注意，此处我们不对锚点数量进行优化。我们首先分别创建所有边，并将每条边的两个端点都设为锚点。然后，对于每个这样的子模式，我们执行 $k-2$ 锚定顶点创建操作，以按正确顺序添加模式中（在 k 顶点上）的所有其他顶点。接着，我们逐步合并这些子模式，直到得到所需的模式。所有这些合并都是正确的，因为在所有合并后的中间模式中，已出现的顶点都被锚定。□

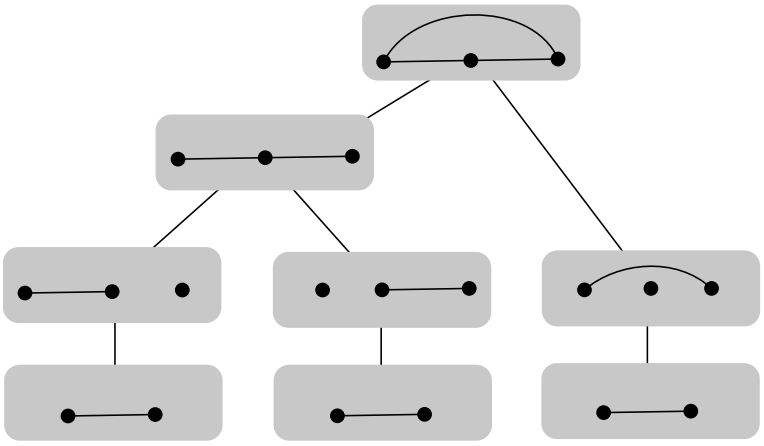


图 10：我们用引理 4 的证明中所用的技术，演示三角形模式的构造。这也展示了定义 6 中定义的合并树。此处，每一步中的所有节点都是锚点。我们从叶节点开始，创建三条边。然后添加顶点，使得这些边分别为 (1, 2)、(2, 3) 和 (1, 3)。随后我们连续执行两次合并。请注意，此处被合并的模式具有相同的顶点集，因此所有锚点都是活动锚点；这是一个特殊情况。一般而言，我们会尽量减少锚点的数量，从而也减少活动锚点的数量。

对于我们所说的分割/合并，关键性质如下。

引理 5. 设 P 是一个可划分为 P_1 和 P_2 的模式，且 G 是一个有序图。如果在 G 中存在 P_1 的一个实现和 P_2 的一个实现，并且它们

在活动锚点的位置上达成一致，那么 G 中存在 P 的一个实现。

在证明这一陈述之前，请参见图11，该图说明了如果没有定义 5 中的条款1e，这一陈述为何不成立。

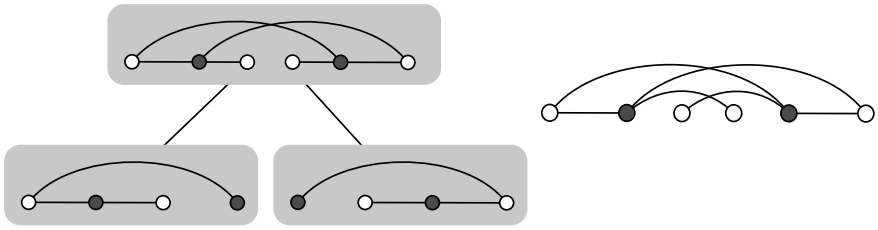


图11：说明我们为何需要定义 5 中的条件 1e。假设我们决定只保留在一次合并过程中被识别出的锚点。在右侧，我们展示了这样一种修改后的合并，其中黑色顶点是锚点。现在，考虑右侧的有序图。它确实具有这两个子模式的一个实现，并且锚点处于正确的位置；但它不具有合并模式的实现。问题在于非锚点顶点的交错。一种正确合并的替代定义，是要求它具有无歧义性：只有一个模式能够由它产生。

证明。考虑引理中的 P、P_1、P_2 和 G。考虑在活动锚点集合上一致的 P_1 和 P_2 的实现。令 H 为这两个实现的并集。根据条款1c，H 中所有不是活动锚点的顶点都彼此不同，因此 H 与 P 具有相同数量的顶点以及相同的活动锚点。在 H 中，两个活动锚点之间（以及第一个活动锚点之前、最后一个活动锚点之后）的所有顶点都只来自两个模式中的一个实现，这是因为条款1e。由于 H 和 P 的活动锚点是一致的，并且 P_1 的实现中顶点的排序（P_2 则相应地）与 P 一致，而且活动锚点之间的每个区段都只属于其中一个实现，所以 H 中的顶点以与 P 相同的顺序出现。于是边也一致，这意味着 H 是 P 的一个实现。□

C.2 合并树与合并宽度

基于上述列出的操作，我们定义了合并树这一自然概念，将图 10 所给出的直观理解形式化。

定义 6。模式 P 的合并树是一棵满足以下条件的树：

1. 每个节点都标记有一个锚定模式。

2. 根节点标记有模式 P 的锚定版本。
3. 每个节点都属于以下情况之一：
(a) 它是叶节点，并标记有一条唯一边。(b) 它只有一个子节点，并标记有一个可通过对该子节点的模式进行顶点创建而得到的模式。(c) 它有两个子节点，并标记有一个可通过合并其子节点的模式而得到的模式。

现在，我们可以定义合并树的宽度，以及参数合并宽度。

定义 7. 合并树的宽度是该合并树所有节点所标记模式中的锚点数量的最大值。模式 P 的合并宽度，是所有 P 的正确合并树的宽度中的最小值。

引入这个参数，是因为在接下来将要定义的算法中，动态规划的复杂度取决于锚点的数量。请记住，在我们的合并和顶点创建操作中（不同于引理 4 的证明），一个顶点可能会失去其锚点身份；我们正是通过这种方式来期望维持较少的锚点数量。

C.3 算法

考虑一个有序图 G、一个模式 P 及其合并树。 G的顶点按照它们在图中的顺序命名。

总体思路。该算法的总体思路如下。对于合并树中标记为带锚点的子模式 P'、且锚点为 a_1、...、 a_ℓ 的每个节点，我们希望列出 G 的所有 ℓ 元组顶点，使得在 G 中存在 P' 的一个实现，并且锚点位于这些位置。这样一来，就可以轻松回答最初的问题：根模式 P 的列表非空，当且仅当 G 中不存在 P 的出现。

我们将在树中自底向上处理节点，利用子节点的列表来创建父节点的列表。

记号。考虑树中的一个节点 u，它标记为具有 ℓ 个锚点的锚定模式 P_u。我们将把上述 ℓ 元组以一个 ℓ 维矩阵 T_u 的形式存储，其中每个维度的大小均为 n，即 T_u 。处理完节点 u 后，该矩阵必须满足以下条件：

– $T_u[v_1, ..., v_\ell] = 1$ 如果存在 G 中的一个 P_u 出现，其对应锚点位于位置 v_1、.....、 v_ℓ；– $T_u[v_1, ..., v_a] = 0$ 否则。

在下一段中，我们将描述如何根据所使用的操作类型计算这些矩阵。

边创建节点。合并树的叶节点对应于边创建。矩阵按以下方式填充:

– 如果边模式的两个端点都是锚点，那么矩阵是二维的，并且在每个满足 $i < j$ 的位置 (i, j)，以及满足 $\in E$ 的位置 (i, j) 上，矩阵的值均为 1。– 如果只有右顶点（分别为左顶点）是锚点，那么矩阵是一维的，并且当且仅当存在一条边 (x, y)（分别为一条边 (y, x)）时，$T_u[y] = 1$。

顶点创建节点。在创建顶点之前，根据定义，周围的两个顶点必须是锚点。我们操作的矩阵取决于创建后哪些顶点是锚点。首先考虑新顶点及其周围的两个顶点在创建后都是锚点的情况。令 P 为创建后的模式，并令 s 为新创建顶点的位置。

令 u 为创建后的节点，并令 w 为其子节点。矩阵 T_u 按以下方式填充：当且仅当 $T_w[v_1, ..., v_{s-1}, v_{s+1}, ..., v_\ell] = 1$ 时，$T_u[v_1, ..., v_{s-1}, v_s, v_{s+1}, ..., v_\ell] = 1$；并且 v_{s-1}、v_{s+1} 不是 G 中的连续顶点。

现在，如果 $\{s - 1, s, s + 1\}$ 中的某些顶点不是 P_u 中的锚点，我们执行类似的操作，但矩阵 T_u 不包含与这些顶点对应的字段。

合并节点示例。在说明合并两个锚定模式的一般算法之前，我们先通过图12展示一个简单示例。

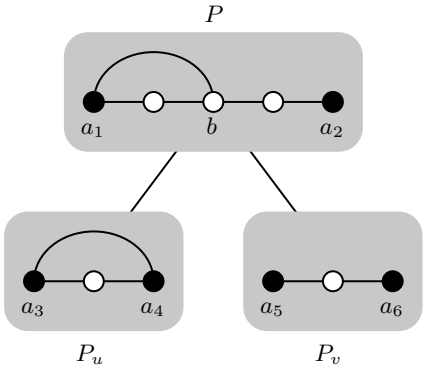


图 12：合并示例。锚点为黑色，其他顶点为白色。模式 P_u 中的锚点 a_4 与模式 P_v 中的锚点 a_5 被合并为顶点 b，从而生成模式 P。这两个锚点是活动锚点，因为合并需要它们。其他锚点是非活动锚点，意味着这次合并不需要它们，但在本图未展示的后续合并中可能会用到。

假设我们已经计算出与 P_u 对应的矩阵 T_u，以及与 P_v 对应的矩阵 T_v。这意味着我们有一个二维矩阵 T_u，其中填有 0 和 1；对于图 G 的顶点集中的每个顶点 v_1、v_2，当且仅当 P_u 能够在 G 中通过 $v_1 = a_3$ 和 $v_2 = a_4$ 实现时，矩阵中对应的值为 1。由于 P_u 锚定了 2 个顶点，因此可以使用标准的二维矩阵。类似地，我们有一个二维矩阵 T_v；对于图 G 的顶点集中的每个顶点 v_1、v_2，当且仅当 P_v 能够在 G 中通过 $v_1 = a_5$ 和 $_2 = a_6$ 实现时，矩阵中对应的值为 1。

我们希望计算矩阵 T 的内容，它对应于模式 P。要计算条目 $T[v_1, v_2]$，我们必须知道 P能否通过 $v_1 = a_1$和 $v_2 = a_2$在 G 中实现。为此，我们讨论 P 的顶点 b。它必须是 G的某个顶点 v_3。因此，$T[v_1, v_2] = \max_{v_3 \in V(G)} T_u[v_1, v_3] \cdot T_v[v_3, v_2]$。这是矩阵乘法的公式，只不过用最大值代替了求和。我们可以先进行经典的矩阵乘法，然后将非零结果替换为 1，从而解决这个问题。利用这种技术，我们可以像矩阵乘法一样快地根据 T_u 和 T_v 计算出 T。用合并的术语来说，我们可以像矩阵乘法一样快地将 P_u 和P_v 合并为 P。

目标是将这一思路扩展到任意合并。在我们的示例中，这样做相对自然，因为它对应于经典的矩阵乘积。处理具有更多锚点的模式时，我们需要从一个 d_1 维矩阵和一个 d_2 维矩阵计算出一个 d 维矩阵，而矩阵乘法无法直接用于此。我们通过第二个示例来看看可以如何处理，该示例见图13。在这种情况下，公式变为：

$$T[v_1, v_2] = \max_{v_3, v_4 \in V(G)} T_u[v_1, v_3, v_4] \cdot T_v[v_3, v_4, v_2].$$

这不再与标准的方阵乘法相关，但我们可以通过合并一些维度，将问题转化回这种形式。设 M_u 为一个 n 行 n^2 列的矩阵。行由 G 的顶点索引，列由 G 的顶点对索引。对于每个顶点 v_1、v_2、$v_3 \in V$（G），我们令 $M_u[v_1,$（v_2，v_3）] =$T_u[v_1, v_2, v_3]$。类似地，我们定义 M_v 为一个 n^2 行 n 列的矩阵，使得 M_v[（v_1，v_2）v_3] =$T_v[v_1, v_2, v_3]$。现在，公式变得类似于矩阵乘法：

$$T[v_1, v_2] = \max_{(v_3, v_4) \in V(G) \times V(G)} M_u[v_1, (v_3, v_4)] \cdot M_v[(v_3, v_4), v_2].$$

这意味着我们可以通过将 M_u 乘以 M_v来计算 T。

合并节点与矩阵乘法。现在，我们将正式说明如何处理合并操作。最后，我们给出该计算的复杂度，其参数为所涉及的三个模式中任一模式的锚点最大数量。

设 A_u 为 P_u的锚点集合，A_v 为 P_v的锚点集合，并考虑将 P_u 与 P_v合并得到的 P'。设 A' 为合并模式P'的锚点集合。根据定义，我们知道 $A' \subseteq A_u \cup A_v$。设 $a = |A_u \cup A_v|$。

要使用矩阵乘法，我们首先需要将每个多维矩阵转换为二维矩阵。我们将不同的锚点分成若干集合：

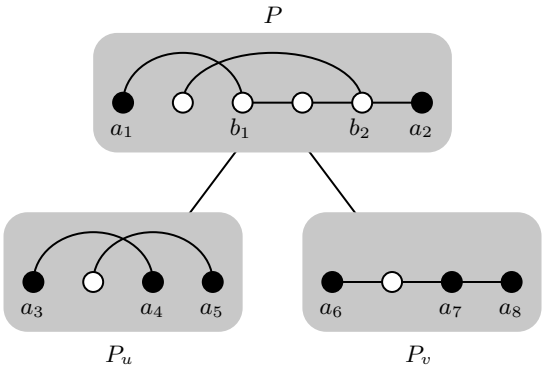


图 13：合并示例。锚点为黑色，其他顶点为白色。模式 P_v 中的 P_u 的锚点 a_4 和 a_6 被识别为模式 P 中的顶点 b_1。模式 P_v 中的 P_u 的锚点 a_5 和 a_7 被识别为模式 P 中的顶点 b_2。这四个锚点是活动锚点，因为合并需要它们。其他锚点是非活动的，这意味着本次合并不需要它们，但图中未展示的后续合并可能会需要它们。

- $U = A_u \setminus A_v$
- $V = A_v \setminus A_u$
- $X = \overline{A'} \cap A_u \cap A_v$
- $Y = A' \cap A_u \cap A_v$

这给出了 $A_u \cup A_v$ 中锚点的一个划分，因为 $A' \subseteq A_u \cup A_v$，而且 $U \cup V$ 中的每个锚点都在 A' 中（这体现了不存在删除被动锚点的操作这一事实）。我们固定 Y 中的所有锚点，并计算一个矩阵，总结对于 $U \cup V$ 中的所有锚点，是否存在一个实现 P' 且锚点位于 $Y \cup U \cup V = A'$ 中。两个二维矩阵如下：

- M_u，其中行以 U 中的元素为索引，列以 X 中的元素为索引，使得

$$M_u[(u_1, \ldots, u_\lambda), (v_1, \ldots, v_\mu)] = T_u[u_1, \ldots, u_\lambda, v_1, \ldots, v_\mu].$$

- M_v，其中行以 X 中的元素为索引，列以 V 中的元素为索引，使得

$$M_v[(v_1, \ldots, v_\mu), (w_1, \ldots, w_\rho)] = T_v[v_1, \ldots, v_\mu, w_1, \ldots, w_\rho].$$

将 M_u 与 M_v 相乘，得到矩阵 M'，其行以 U 为索引，列以 V 为索引，使得当且仅当存在一个具有锚点 $u_1, \ldots, u_\lambda, w_1, \ldots, w_\rho$ 的 P' 的实现时，$M'[(u_1, \ldots, u_\lambda), (w_1, \ldots, w_\rho)]$ 为正。对 Y 中所有锚点的所有实现进行乘法运算后，我们就能填充 T'。

其复杂度为某个 $O(n^x)$。让我们计算 x。回想一下，两个 $n \times n$ 矩阵相乘的代价为 $O(n^\omega)$，而相乘的代价

一个 $n^a \times n^b$ 矩阵与一个 $n^b \times n^c$ 矩阵相乘的代价为 $O(n^{\omega(a,b,c)})$。利用巧妙的分块矩阵乘法，我们有[14], $\omega(a,b,c) \leq \omega \times \max(a+b, b+c, a+c)/2$。在我们的情形中，对于 $|Y|$ 中锚点的每一种实现，都需要将一个 $|U| \times |X|$ 矩阵与一个 $|X| \times |V|$ 矩阵相乘，即需要计算 $O(n^{|Y|})$ 次。因此得到$x = |Y| + \omega \times \max(|U|+|X|, |X|+|V|, |U|+|V|)/2 \leq \omega \times \max(|Y|+|U|+|X|, |Y|+|X|+|V|, |Y|+|U|+|V|)/2$。令 p 为锚点的最大数量（即 $p = \max(|A_u|, |A_v|, |A'|)$），即可得到 $x \leq \omega p/2$。

因此，得到以下定理：

定理 5. 检测一个合并宽度为 p 的模式，其复杂度为 $O(n^{\omega p/2})$；其中，将两个 $n \times n$ 矩阵相乘的复杂度为 $O(n^{\omega})$。

C.4 有界合并宽度的模式

本节中，我们将证明存在一类自然的模式（我们称之为外平面模式），其合并宽度有界，这也说明了研究这一参数的意义。我们还将证明这一结果的一个推广。我们认为，还有若干其他模式类也具有有界合并宽度，但将留待后续工作探讨。

定义 8. 外平面模式是这样一种模式：将所有顶点按递增顺序放置在一条直线上，并将所有边画成位于直线上方的半圆时，各边彼此不相交。

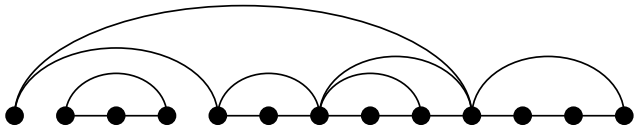


图14：一个包含14个顶点和17条边的外平面模式示例。为便于阅读，边可能不以半圆表示，而是以半椭圆甚至直线表示。

本节的第一部分（引理6和引理7）旨在证明以下定理：

定理6. 外平面模式的合并宽度至多为2。

结合定理 5，可得以下推论。

推论9. 存在一种算法，可在时间$O(n^{\omega})$内检测外平面模式，其效率与矩阵乘法算法相当。

定理6的证明由两个引理组成：引理 6 说明如何分解外平面模式，引理 7 说明如何将其转换为正确合并树。

引理 6。顶点集 p_1、...、p_k 上具有 $k > 1$ 的任意外平面模式 P 都可以通过以下方式之一进行分解:

1.（覆盖边）它有一条边（p_1, p_k），我们用 $P \setminus (p_1, p_k)$ 表示去掉这条边后的同一模式。见图 15。2.（割点）它有一个顶点 p_t，满足 $1 < t < k$，并且有一条边（p_1, p_t），使得不存在形如（p_i, p_j）且满足 $i < t < j$ 的边。我们分别用 $P[1, t]$ 和 $P[t, k]$ 表示使用顶点集（p_1, ..., p_t）和（p_t, ..., p_k）的两个模式。见图 16。3.（首顶点孤立）第一个顶点 p_1 是孤立的。我们用 $P[2, k]$ 表示模式的其余部分。见图 17。

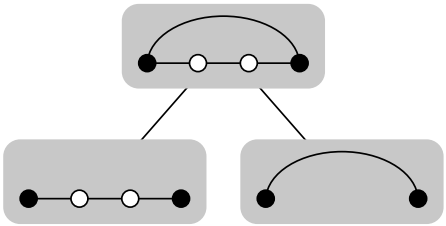


图 15：覆盖边

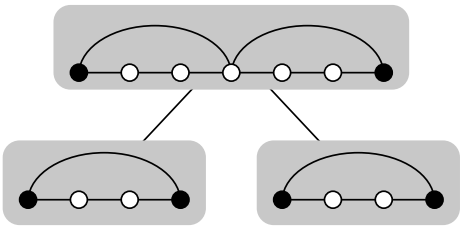


图 16：割点

证明。注意，我们声称唯一无法分解的模式是单顶点模式。若该模式不是单顶点模式，我们对 p_1 的邻域进行分类讨论。若 p_1 的度为 0，则属于首顶点孤立情形。若 p_1 和 p_k 之间有一条边，则属于覆盖边情形。否则，这意味着 p_1 与其他某些顶点 p_t 之间存在一条边，且 $t < k$。令 t 为满足条件的最大顶点。我们检查不存在形如 (p_i, p_j) 且满足 $i < t < j$ 的边。对于 $i = 1$，由于 t 是极大的，不存在形如 (p_1, p_j) 且满足 $t < j$ 的边。对于 $i > 1$，由于 P 是外平面模式，不存在与边 (p_1, p_t) 相交的边，也就是说，不存在形如 (p_i, p_j) 且满足 $1 < i < t < j$ 的边。因此，这属于割点情形。□

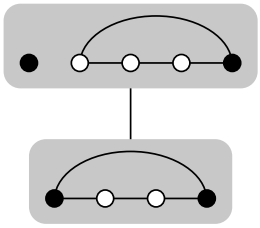


图 17：首顶点孤立

注 4. 引理 6 中的操作保持外平面性：将外平面模式分解后得到的模式仍是外平面模式。

引理 7。设 P 是一个外平面模式。存在一棵针对模式 P 的合并树，使得树中的每个节点都标记为一个具有 2 个锚点的模式，且这两个锚点分别位于该模式最左侧和最右侧的顶点上。

在图 15、16 和 17 中，这些锚点以黑色表示。

证明。我们基于引理 6 的分解，通过归纳法证明该引理。请注意，不适用该分解的情况（单顶点模式）非常适合合并树的定义，因为创建一个顶点是定义 5 中允许的操作，而且我们可以将这些顶点放在叶节点上。

设 P 是顶点集为 p_1、...、p_k、 $k > 1$ 上的一个模式。我们首先根据引理 6 中的情况分类，描述 P 的合并树，然后论证其正确性。

1.（覆盖边）存在一条边（p_1， p_k），并令 $P' = P \setminus$（p_1， p_k），即不含此边的相同模式。令 T' 为对应于 P' 的合并树，该树由归纳法构造得到。然后，将 T' 的根节点标记为 P' 的锚定版本，其中顶点 p_1 和 p_k 为锚定的。我们按如下方式构造对应于 P 的合并树 T：

– 根节点标记为 P 的锚定版本，其中顶点 p_1 和 p_k 为锚定的。– 根节点的左子树为 T'。– 根节点的右子树为单个节点，其标记为边（p_1， p_k），且两个端点均为锚定的。

2.（割点）存在一个顶点 p_t 及 $1 < t < k$，以及一条边（p_1， p_t），使得不存在形如（p_i， p_j）且满足 $i < t < j$ 的边。令 $P_1 = P[1, t]$ 和 $P_2 = P[t, k]$ 分别为使用顶点集（p_1，...，p_t）和（p_t，...，p_k）的两个模式。令 T_1 为对应于 P_1 的合并树，该树由归纳法构造得到，并令 T_2 为对应于 P_2 的合并树。我们按如下方式构造对应于 P 的合并树 T：

– 根节点标记为 P的锚定版本，其中顶点 p_1和 p_k 已锚定。– 根节点的左子树是树 T_1。请注意， T_1 的根节点标记为 P_1的锚定版本，其中锚点位于顶点 p_1 和 p_t上。

– 根节点的右子树是树 T_2。请注意， T_2 的根节点标记为 P_2的锚定版本，其中锚点位于顶点 p_t 和 p_k上。

3.（首顶点孤立）首顶点 p_1 是孤立的。令 $P' = P[2, k]$ 为模式的其余部分。令 T' 为对应于 P'、通过归纳法构造的合并树。这意味着 T' 的根节点标记为 P' 的锚定版本，其中顶点 p_2 和 p_k 均已锚定。我们按如下方式构造对应于 P 的合并树 T:

– 根节点标记为 P 的锚定版本，其中顶点 p_1 和 p_k 均已锚定。– 根节点唯一的子节点是 T'。

要完成证明，我们只需检查每个合并是否符合定义 5，即是否满足条件 (a) 至 (e)。覆盖边和孤立顶点这两种情况显而易见。我们重点讨论割点的情况。我们需要证明，顶点 p_1 和 p_k 已锚定的模式 $P = (V, E)$，可以拆分为顶点 p_1 和 p_t 已锚定的模式 $P_1 = (V_1, E_1)$，以及顶点 p_t 和 p_k 已锚定的模式 $P_2 = (V_2, E_2)$:

(a) (E_1, E_2) 是 E 的一个划分：设 (p_i, p_j) 是一条边。若 $i < j \leq t$，这条边属于 E_1。若 $t \leq i < j$，则它属于 E_2。我们知道不存在其他类型的边，因为根据割点情形的定义，不可能出现 $i < t < j$。我们还需要检查 $\forall (p_i, p_j) \in E_1, p_i, p_j \in V_1$：由于 $(p_i, p_j) \in E_1$，我们知道 $i < j \leq t$，这证明了 $V_1 = \{p_1, \ldots, p_t\}$。关于 E_2 和 V_2，情况也相同。(b) $V_1 \cup V_2 = V$，因为 $V_1 = \{p_1, \ldots, p_t\}$，$V_2 = \{p_t, \ldots, p_k\}$ 和 $V = \{p_1, \ldots, p_k\}$。(c) $V_1 \cap V_2 = \{p_t\}$，它在 P_1 和 P_2 中都是一个锚点。(d) 模式 P 的锚点是 p_1 和 p_k；它们确实是各自出现的每个模式中的锚点：p_1 是 P_1 中的一个锚点，而 p_k 是 P_2 中的一个锚点。(e) 顶点在模式 P_1 和 P_2 中的分布如下：顶点 $p_1, \ldots, p_{t-1}$ 只属于 P_1，顶点 p_t 同时属于 P_1 和 P_2，而顶点 $p_{t+1}, \ldots, p_k$ 只属于 P_2。因此，不存在这样一对连续顶点：一个只属于 P_1，另一个只属于 P_2，这要归功于位于中间的顶点 p_t。□

扩展。现在我们将证明这一结果的一个扩展。为此，我们先定义一种距离。

定义 9。模式 P 到外平面性的编辑距离，记为$dist_{out}(P)$，是从 P 中移除边，使其成为外平面模式所需的最少边数。

定理 7。任意模式 P的合并宽度均有上界 $2dist_{out}(P) + 2$。

结合定理 5，可得:

定理 8。设 P 是一个模式。存在一种算法，可在时间 $O(n^{\omega(dist_{out}(P)+1)})$ 内检测 P，即效率与两个 $n^{dist_{out}+1} \times n^{dist_{out}+1}$ 矩阵的乘法相当。

在证明定理 7 之前，我们先来看下面的一般引理，其中我们“强制”设置一个锚点。

引理 8. 设 P 是一个合并宽度为 w 的模式，且 p_i 是 P 的一个顶点。存在一棵用于模式 P 的合并树 T'，使得顶点 p_i 锚定在 T' 的根节点中，并且 T' 的宽度至多为 $w+1$。

证明。设 T 是 P 的一棵宽度为 w 的合并树。我们通过将顶点 p_i 的每个实例锚定在 T 的标签中，从 T 构造出 T'。根据定义，T' 的宽度至多为 $w+1$。接下来只需验证 T' 是正确的，也就是说，每个标记着含有两个子节点的节点的模式 $P=（V，E）$，都可以按照定义 5，通过正确合并其两个子节点 $P_1=（V_1，E_1）$和 $P_2=（V_2，E_2）$的模式得到。我们需要检查每个模式是否满足条件 (a) 至 (e)：由于 V_1、V_2、E_1 和 E_2 在 T 和 T' 中相同，条件 (a)、(b) 和 (c) 在 T' 中仍然成立。对于条件 (d)，由于我们在 T' 的每个模式中都锚定顶点 p_i，因此该条件仍然成立。最后，由于该顶点在其出现的每个模式中都已锚定，因此在合并时，它会同时锚定在 V_1 和 V_2 中，故条件 (e) 也得到满足。□

证明（（定理 7））。设 $P=(V,E)$ 是一个模式，其顶点集为 $V=p_1,\ldots,p_k$。设 $P_1=(V_1,E_1)$ 是通过从 P 中移除 $dist_{out}(P)$ 条边得到的一个外平面模式。注意，由于我们没有移除任何顶点，因此 $V_1=V$。设 $P_2=(V_2,E_2)$ 是由 P_1 中移除的边所诱导的模式，也就是说，V_2 是作为 E_2 中某条边端点的顶点集。注意，P_2 有 $dist_{out}(P)$ 条边，且至多有 $2dist_{out}(P)$ 个顶点。由于 P_1 是外平面图，根据引理 7，它有一棵宽度至多为 2 的合并树，其根节点标签锚定在 p_1 和 p_k 中。我们使用引理 8，将 V_2 中的所有顶点都添加为这棵合并树根节点的锚点。由此得到一棵宽度至多为 $2dist_{out}(P)+2$ 的合并树 T_1。令 $A=\{p_1,p_k\}\cup V_2$ 为 T_1 根节点标签的锚点集。注意，$|A|\leq 2dist_{out}(P)+2$。由于 P_2 至多有 $2dist_{out}(P)$ 个顶点，根据引理 4，使用一棵平凡树即可得到一棵宽度至多为 $2dist_{out}(P)$ 的合并树 T_2。特别地，V_2 的每个顶点都是 T_2 每个标签中的锚点。

令 T 为合并树，其根节点标记为 P，并以 A 的所有顶点为锚点，且有两个子树。左子树为树 T_1，右子树为树 T_2。T_1 的根节点（即 P_1）标记为一个以 A 的所有顶点为锚点的模式，而 T_2 的根节点（即 P_2）标记为一个以 V_2 的所有顶点为锚点的模式。接下来需要检查 T 的根节点（即以 A 为锚点的 P）与其两个子节点 P_1 和 P_2 是否构成正确合并，方法是验证定义 5 的条件 (a) 至 (e)：

(a) 根据定义，(E_1,E_2) 是 E 的一个划分。$\forall(p_i,p_j)\in E_1$，我们有 $p_i,p_j\in V_1=V$。根据 V_2 的定义，$\forall(p_i,p_j)\in E_2$，我们有 $p_i,p_j\in V_2$。(b) $V_1\cup V_2=V$，因为 $V_1=V$。(c) $V_1\cap V_2=V_2$，并且 V_2 中的每个顶点都是 P_1 和 P_2 中的锚点。(d) 模式 P 的锚点来自 A。P_1 接收来自 P 的所有顶点，其锚点集也等于 A。P_2 接收来自 P 的 V_2 中的顶点，其锚点集等于 $V_2=A\cap V_2$。

(e) 每个顶点都在 $V_1 = V$ 中，因此不存在不在 V_1 中的顶点。 □

最后，我们给出另一个推论。外平面交叉数（也称为凸交叉数、单页书交叉数或圆形交叉数）是一个常用的图参数（参见综述 [21]中的“凸交叉数”条目）。我们将其推广到模式，并证明当该参数有界时，合并宽度也有界。

定义 10。模式 P 的外平面交叉数是 P中交叉边对的数量；当且仅当 $i < i' < j < j'$ 或 $i' < i < j' < j$成立时，两条边（i, j）和（i', j'）相交。

推论 10。如果一个模式的外平面交叉数为 c，那么它的合并宽度至多为 $2c + 2$。

该命题可由以下事实推出：模式的外平面交叉数总是大于或等于将该模式变为外平面图所需的编辑距离。事实上，对于每一对交叉边，删除其中一条边。如此一来，就不再有交叉边，这意味着该模式已成为外平面图。

D 正外平面森林的线性时间检测

本节将展示一个规模任意大的自然模式族，其中任意模式都可以在线性时间内检测。这个模式族由外平面模式（即边之间没有交叉，见 C 节）、正模式（即没有禁用边）和无环模式组成。我们称这些模式为正外平面森林。

从直观上看，根据我们目前的认知，这三个约束似乎是必要的。事实上，如果没有外平面性（或近似外平面性），我们就无法判断大规模模式的性质，而我们的下界表明，稀疏图更容易处理。此外，存在禁用边时，人们往往会想对输入图取补图，而这在线性时间内是不可能的。最后，只要存在一个环，情况似乎就与三角形检测类似，此时我们需要矩阵乘法时间。

本节旨在证明以下定理。

定理 9. 正外平面森林可在 $O(n + m)$ 时间内检测。

设 $P = (V, M, F, U)$ 是一个正外平面森林。由于不存在禁用边，我们将简称“边”来指代必选边。

下面介绍一些与边的相对位置相关的术语。参见图 18。考虑一对边 (i, j)、(i', j')，其中有 $i < j$、$i' < j'$ 和 $i \le i'$。在外平面模式中，有两种配置：

– 它们要么是嵌套的：$i \le i' < j' \le j$，

– 或者并排： $i < j \leq i' < j'$。

（它们不能相交，否则将会是 $i < i' < j < j'$。）当边并排放置时，我们可以称其为左边和右边。

当边相互嵌套时（$i \leq i' < j' \leq j$），嵌套边（即（i'，j'））还可以进一步细分：

– 如果（i'，j'）直接嵌套在（i，j）内，且两者之间没有嵌套边，则称其为直接嵌套；– 否则，（i'，j'）深度嵌套在（i，j）内。

接下来，我们希望划分嵌套在某条特定边（i，j）内的所有边。为此，我们可以根据这些边在不经过（i，j）的情况下与边（i，j）的连接方式，轻松地将直接嵌套在（i，j）内的边划分为三部分：

– 通过路径与 i 相连（不使用边 (i,j)）的边，称为左嵌套边；– 通过路径与 j 相连（不使用边 (i,j)）的边，称为右嵌套边；– 以及既不与 i 相连、也不与 j 相连的边，称为中心嵌套边。

请注意，由于我们的模式是无环的，因此不可能存在一条边同时与 i 和 j 相连，否则将形成一个环。这类边称为双重嵌套边，而我们知道永远不会遇到这类边。对于深度嵌套的边，这意味着它们嵌套在另一条边 (i',j') 内，而该边已经完成分类（因为每条直接嵌套边都可以被分类为左嵌套边、右嵌套边或中心嵌套边）。我们将该边分类为与 (i',j') 相同的类别。总而言之，我们已经将嵌套在 (i,j) 内的每条边，相对于 (i,j)，分类为左嵌套边、右嵌套边或中心嵌套边。

当我们拥有一个平面森林模式 P 时，还可以引入一些特殊的边和子模式。

定义 11。设 $P = (V, M, F, U)$ 是一个正平面森林模式。

– 令 *LOME* 为最左外边，即满足 i 最小的边 $(i,j) \in M$；若存在多条这样的边，则取满足 j 最大的边。– 令 *ROME* 为最右外边，即满足 j 最大的已确定边 $(i,j) \in M$；若存在多条这样的边，则取满足 i 最小的边。

边 LOME 和最右外边具有将 P 划分为 3 个特殊子模式的特性。见图 19。

先讨论 LOME。设 LOME 是满足 $i_{\text{LOME}} < j_{\text{LOME}}$ 的边 $(i_{\text{LOME}}, j_{\text{LOME}})$。根据定义，如果 $(i,j) \in E$，那么 (i,j) 要么嵌套在 $(i_{\text{LOME}}, j_{\text{LOME}})$ 内，要么位于 $(i_{\text{LOME}}, j_{\text{LOME}})$ 的右侧。如果 (i,j) 嵌套在 $(i_{\text{LOME}}, j_{\text{LOME}})$ 内，由于 P 是森林，(i,j) 不可能是双重嵌套的。因此，(i,j) 要么是左侧嵌套、右侧嵌套，要么是中心嵌套。我们定义以下三个子模式：

– 由（$i_{\text{LOME}}, j_{\text{LOME}}$）内的左嵌套边和中心嵌套边构成的模式 P_L。

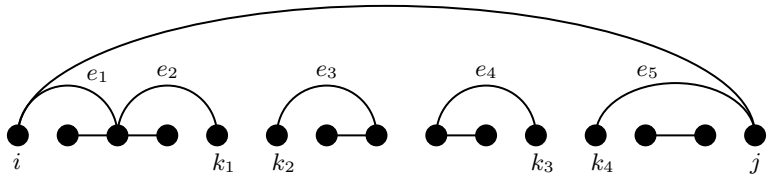


图 18：嵌套在 (i, j) 内的边的分类。除边 (i, j) 外，弧线表示的边都是直接嵌套边，直线表示的边都是深度嵌套边。直接嵌套边根据是否存在一条从它们通往 i、j 或两者都不通的路径进行分类。因此，边 e_1 和 e_2 是左嵌套的（因为避开边 (i, j) 后，它们与 i 相连，而不与 j相连），边 e_3 和 e_4 是中心嵌套的，因为它们不与 (i, j) 相连，而边 e_5 是右连接的。至于深度嵌套在 (i, j) 内的边，即直线表示的边，它们的分类与其所嵌套其中的直接嵌套边相同。因此，嵌套在 e_1 或 e_2 内的每条边，在 (i, j) 内都是左嵌套的；嵌套在 e_3 或 e_4 内的每条边，在 (i, j) 内都是中心嵌套的；嵌套在 e_5内的每条边，在 (i, j) 内都是右嵌套的。总而言之，我们可以这样概括这个例子：端点都位于顶点 i 和 k_1 之间的每条边都是左嵌套的；端点都位于顶点 k_2 和 k_3 之间的每条边都是中心嵌套的；端点都位于顶点 k_4 和 j 之间的每条边都是右嵌套的。总可以利用顶点 k_1 到 k_4 进行这样的划分。

– 由（$i_{\texttt{LOME}}, j_{\texttt{LOME}}$）内的右嵌套边构成的模式 P_R 。– 由未嵌套在（$i_{\texttt{LOME}}, j_{\texttt{LOME}}$）内的边构成的模式 P_O ，即位于（$i_{\texttt{LOME}}, j_{\texttt{LOME}}$）右侧的边。

请注意， P 的每条已确定边恰好属于一个子模式： P_L、 P_R 或 P_O，但 LOME 除外，它不属于任何模式。

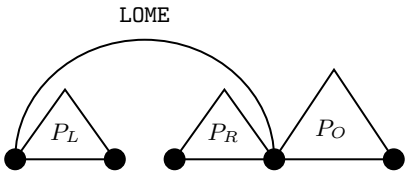


图 19：由边 LOME 诱导的子模式示意图。三角形表示一个子模式。请注意，这些模式都可能为空。此外，模式 P_L 和 P_O 可能与 LOME 不连通。这是因为 P_L 可能仅包含中心嵌套边，并且 P_O 不要求与 LOME 连通。另一方面，根据定义， P_R 始终与 LOME 连通。

对于最右外边，我们也有同样的 3 种子模式：

– 由 $(i_{\texttt{ROME}}, j_{\texttt{ROME}})$ 内的左嵌套边和中心嵌套边构成的模式 P_L 。– 由 $(i_{\texttt{ROME}}, j_{\texttt{ROME}})$ 内的右嵌套边构成的模式 P_R 。– 由未嵌套在 $(i_{\texttt{ROME}}, j_{\texttt{ROME}})$ 内的边构成的模式 P_O ，即位于 $(i_{\texttt{ROME}}, j_{\texttt{ROME}})$ 左侧的边。

给定一个模式 $P = (V(P), F(P), M(P), U(P))$、一个图 $G = (V(G), E(G))$，以及一个实现了 P的子图 H（属于 G），我们可以考虑 $u = \min_\tau V(H)$ 和$v = \max_\tau V(H)$。这意味着，根据顺序 τ， H 中的每个顶点都位于区间 $[u..v]$内。一个有趣的集合是集合 S，它包含了对每个实现 P的子图H 对应的所有 $[u..v]$ 。然而，这样的集合的大小可能与 G的规模呈二次关系。如果我们想在线性时间内验证模式，就无法使用这个集合。因此，我们考虑两个数组（m_P^+ 和 m_P^-），它们足以运行我们的算法；此外还有另外两个数组（M_P^+和 M_P^-），可以在线性时间内由前两个数组推导出来：

定义 12. 令 $m_P^+(u)$ 为满足以下条件的最小 v：存在 G 的一个子图H，使其实现 P，且 $u = \min_\tau V(H)$ 和 $v = \max_\tau V(H)$。若不存在这样的 H，则令 $m_P^+(u) = +\infty$。令 $M_P^+(u) = m_P^+(> u) = \min_{u'>u} m_P^+(u')$。这意味着， $M_P^+(u)$ 是所有满足以下条件的子图 H 中最小的 v：它们是 G 的子图，能实现 P，且满足 $u < \min_\tau V(H)$ 和 $v = \max_\tau V(H)$。

类似地，令 $m_P^-(v)$ 为所有满足以下条件的子图 H 中最大的 u：它们是 G 的子图，能实现 P，且满足 $u = \min_\tau V(H)$ 和 $v = \max_\tau V(H)$。若不存在这样的 H，则令 $m_P^-(v) = -\infty$。此外， $M_P^-(v) = m_P^+(< v) = \max_{v'<v} M_P^-(v')$。

我们有以下引理：

引理 9. 设 $P = (V(P), M(P), F(P), U(P))$ 是一棵有 k 条边的正平面树， $G = (V(G), E(G))$ 是一个有 n 个顶点和 m 条边的图，并且 f 是函数 $m_P^+(\cdot)$ 或 $m_P^-(\cdot)$。那么，对于某个常数 c，我们可以在 $ck(n+m)$ 时间内计算出 f 的所有取值。

证明。我们对 k 使用归纳法来证明。基础情形是 $k = 1$，它由仅含一条边的模式构成。通过扫描 G 每个顶点的邻接表，可以在 $O(n+m)$ 时间内计算出 f 的所有取值。

现在假设对于所有有 k 条边的模式，该引理均成立，并证明它对有 $k+1$ 条边的模式 P 也成立。首先，考虑情形 $f = m_P^+$。我们考虑 LOME 分解。设 P_L、P_R 和 P_O 是相应的 3 个子模式。根据归纳法，我们可以在 $c|E(P_L)|(n+m)$ 时间内计算出 $m_{P_L}^+$ 的所有取值，在 $c|E(P_R)|(n+m)$ 时间内计算出 $m_{P_R}^-$ 的所有取值，并在时间$c|E(P_O)|(n+m)$ 内计算出 $m_{P_O}^+$ 的所有取值。如果某个子模式为空，则其对应的函数（$m_{P_L}^+$、$m_{P_R}^-$或 $m_{P_O}^+$）是恒等函数。到目前为止，我们花费的时间为 $c(|E(P_L)|+|E(P_R)|+|E(P_O)|)(n+m) = ck(n+m)$。此外，我们还可以利用 $m_{P_L}^+$ 和 $m_{P_O}^+$ 中的取值，并通过扫描这些取值，在 $2n$ 时间内计算出 $M_{P_L}^+$ 和 $M_{P_O}^+$ 的取值。

现在，对于 G 的每个顶点 u，我们计算 $f(u)$。具体做法是遍历 G 的所有边（u, v），满足 $u \leq m_{P_L}^+(u) < m_{P_R}^-(v) \leq v$，并返回 $m_{P_O}^+(v)$ 的最小值。不过，如果 P_L 未与 LOME 连通，我们就替换

$m^+_{P_L}$ 为上述公式中的 $M^+_{P_L}$；如果 P_O 未与 LOME 连通，我们就将上述最小值中的 $m^+_{P_O}$ 替换为 $M^+_{P_O}$。所有这些操作都在 $3m$ 时间内完成。

总而言之，在 $c \geq 5$ 的条件下，我们已在时间 $ck(n+m)+2n+3m \leq c(k+1)(n+m)$ 内计算出 $f(u)$ 的所有值。

对于 $f = m^-_P$，我们也进行类似的操作，只是用最右外边替代 LOME。 □

这足以证明定理 9。为检测一个正外平面森林 P，我们对每个顶点 $u \in V(G)$ 计算 $m^+_P(u)$ 的值。若每个顶点的该值均为 $+\infty$，则模式 P 未在 G 中实现。否则，P 在 G 中实现。

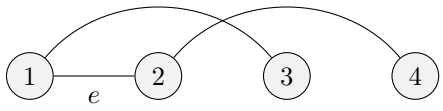
E 每个正 P_4 的线性检测

在本节中，我们证明：对于四个顶点上的正模式，若其必选边构成一条经过四个顶点的路径（P_4），则可在线性时间内检测。请注意，这对 P_4 的任何排序都成立，而不仅仅是自然排序。与 D 节中关于正外平面森林的内容相比，这一结果既更一般，因为可以存在边交叉；又更受限，因为我们只有一条由必选边组成的短路径，而不是一个森林。

定理 10. 所有四个顶点的正模式，只要其必选边构成一个 P_4，都可以在线性时间内检测。

E.1 一些记号

设 P 是四个顶点的模式之一，其必选边构成一个 P_4。模式 P 由三条边组成：中间一条，两侧各一条。例如，在下面的模式中，边 e 是 P_4 的中间边。



设 $e = (i, j)$ 是图中一条由 τ 排序的边。我们定义：

– $e^+ = (i, j')$，其中 j' 是大于 j 且使 (i, j') 成为一条边的最小顶点。– $e^- = (i, j')$，其中 j' 是小于 j 且使 (i, j') 成为一条边的最大顶点。– $^+e = (i', j)$，其中 i' 是大于 i 且使 (i', j) 成为一条边的最小顶点。– $^-e = (i', j)$，其中 i' 是小于 i 且使 (i', j) 成为一条边的最大顶点。

如果 $e = (i, j)$ 是一条边，令 $\min e$ 为 i 和 j 中最小的顶点，令 $\max e$ 为 i 和 j 中最大的顶点。当我们按名称而非端点 (i, j) 来考察一条边 e 时，这种记法很有用。

我们可以在线性时间 $O(n+m)$ 内构建一个查找表，其中包含每条边 e 对应的 e^+、e^-、${}^+e$ 和 ${}^-e$ 的值。具体做法是，在 $O(n+m)$ 内对所有邻接表进行排序，然后对它们进行扫描。

此外，我们可以在线性时间 $O(n+m)$ 内构建一个查找表，其中包含所有顶点 i 对应的 $\min N^-(i)$、$\max N^-(i)$、$\min N^+(i)$ 和 $\max N^+(i)$ 的值。

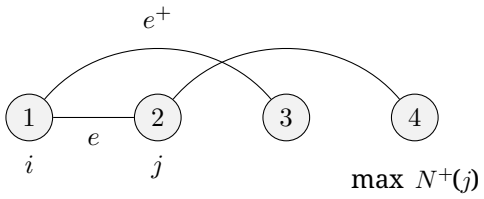
因此，对于每个顶点 i 和边 e，都可以在常数时间内找到 $N^-(i)$、$N^+(i)$、e^+、e^-、${}^+e$ 和 ${}^-e$ 的最小值和最大值。

E.2 聚焦于一个案例

我们将以一个具体示例来解释该算法，即上文（及下文）中的模式，然后说明如何调整证明。

核心思路是扫描所有边 e，并检查是否能找到一个模式 P，其中 e 是中间边。另一个关键思路是，对每条边 e 只寻找一个潜在模式，但要确保如果找不到模式，那么就不存在以 e 为中间边的模式 P。

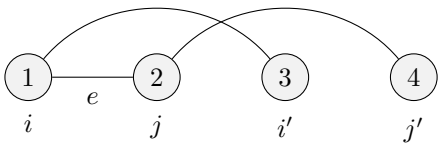
在下面的例子中，对于每条边 e，我们只尝试检测一个在以下实现中具有极大性的模式：
– 顶点 3 是 $\max e^+$– 顶点
4 是 $\max N^+(j)$。
这足以解决检测问题。



有两种情况：如果 $\max e^+ < \max N^+(j)$，那么我们就找到了一个模式P。如果 $\max e^+ \geq \max N^+(j)$，那么就不存在以 e 为中间边的模式 P。让我们证明第二种情况。

命题 14. 如果 $\max e^+ \geq \max N^+(j)$，那么就不存在以 e 为中间边的模式 P。

证明。假设存在矛盾，即存在一个模式 P，其 e 为中间边，并且 $\max e^+ \geq \max N^+(j)$。由于存在一个模式 P，其 e为中间边，因此存在两个顶点 $i' < j'$，使得 (i, i') 和 (j, j') 都是边。



由于 $j' \in N^+(j)$，我们有 $j' \leq \max\ N^+(j)$。由于 i' 是一个大于j的顶点，且 (i, i') 是一条边；又由于 $\max e^+$ 是大于j且满足 $(i, \max e^+)$ 是一条边的最小顶点，因此我们有 $\max\ e^+ \leq i'$。总之，我们有 $j' \leq \max\ N^+(j) \leq \max\ e^+ \leq i'$，这意味着 $j' \leq i'$，与 $i' < j'$ 的事实矛盾。□

因此，可以通过扫描图 G 的每条边 e，并检查条件 $\max e^+ < \max\ N^+(j)$，来检测模式 P。由于可以在常数时间内检查该条件，经过线性预处理后，就可以在线性时间内检测 P。为了推广这一思路，设 ϕ 是一个布尔函数，它以 e 为输入，并输出布尔值“$\max\ e^+ < \max\ N^+(j)$”。现在，检测算法可以写成算法 3

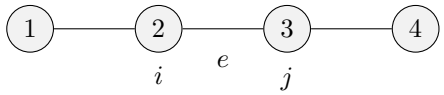
输入： G 一个以邻接表给出的图，以及 τ 一个顶点的全序输出： (G, τ) 是否包含模式 P不失一般性，假设 $\tau = 1, 2, \ldots, n$对于 $e\ in\ E$ 执行 如果 $\phi(e)$那么停止 输出“是，已找到 P”结束 结束输出“否，未找到 P”算法 3: 检测模式 P

|

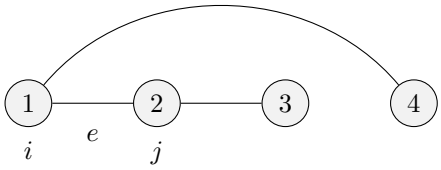
|

E.3 其他情况

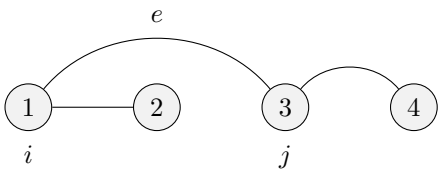
为了证明正向 P_4 的十二版本可以在线性时间内检测，只需找到一个布尔函数 ϕ，使其可以在常数时间内求值（最终可通过线性预处理实现），并给出证明：“如果 $\phi(e)$ 为假，那么不存在以 e 为中间边的模式 P。”以下列出 P_4 的八种模式，包括作为示例展示的一种，但不包括通过反转顺序 τ 得到的对称变换。每种模式都附有其函数 $\phi(e)$。这里没有给出“如果 $\phi(e)$ 为假，那么不存在以 e 为中间边的模式 P” 的证明，因为这些证明都与命题 14 非常相似。



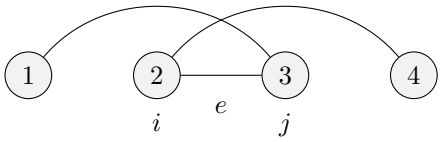
$\phi(u)$ = “$N^-(i) \neq \emptyset$ 和 $N^+(j) \neq \emptyset$”



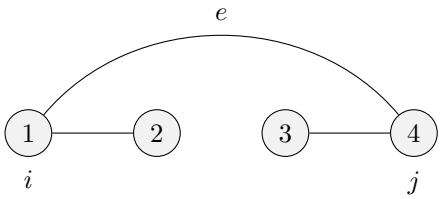
$\phi(e)$ =“最小值 $N^+(j)$ <最大值 $N^-(i)$”



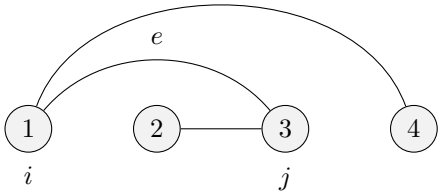
$\phi(e)$ =“最小值 $N^+(i) < j$和 $N^+(j) \neq \emptyset$”



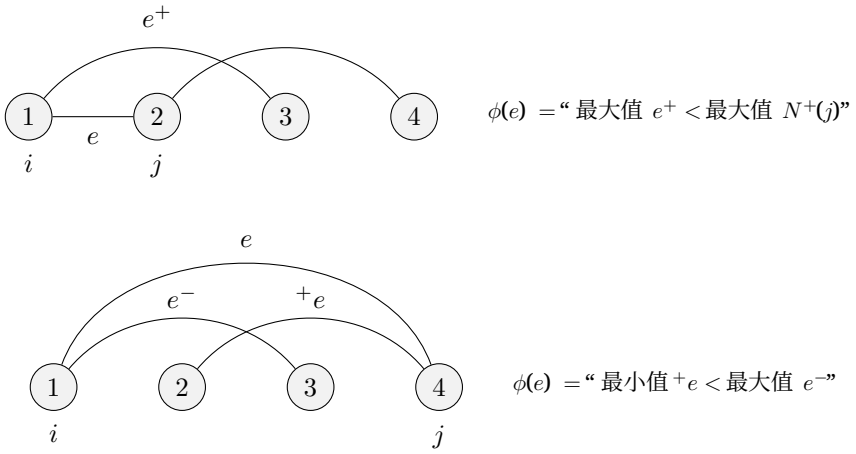
$\phi(e)$ =“最小值 $N^-(j) < i$和 $j <$最大值 $N^+(i)$”



$\phi(e)$ =“最小值 $N^+(i) <$最大值 $N^-(j)$”



$\phi(e)$ =“$i <$最大 $N^-(j)$ 和 $j <$最大 $N^+(i)$”



F 由几何产生的模式

文献中研究的大多数模式都包含三个顶点。近期研究过的一组四顶点模式是 [8],，它在研究具有底层序的某些交图时会自然出现。本节将证明其中一些模式的检测结果。

定义 13。考虑四个节点 $\{1,2,3,4\}$ 上的模式。我们对一些节点对使用以下名称：a = (1,2)、b = (2,3)、c = (3,4)、d = (1,4)（见图 20）。设 $F \subseteq \{a,b,c,d\}$，模式 P_F 是一个包含四个顶点的模式，其必选边为 (1, 3) 和 (2, 4)，禁边集为 F。类 $\mathcal{C}_F$ 是禁用 P_F 的类。

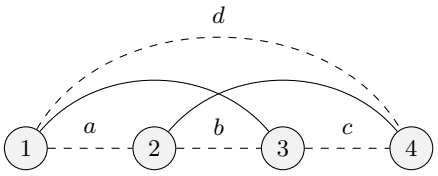


图 20： P_{abcd} 的示意图（定义 13）。

在本节中，我们将证明以下定理：

定理 11. 模式 $P_\emptyset$, P_a, P_b, P_c, P_{ab}, P_{bc} 可以在线性时间内检测。

定理 12。模式 P_{ab} 与三个顶点上的余可比模式同样难。

注意，根据命题 11，假设 $k = 4$ 满足假设 1，我们还知道，检测模式 P_{abcd} 至少需要 $n^{\omega(2,1,1)-o(1)}$ 时间，其中 $n^{\omega(2,1,1)}$ 是将一个 $n^2 \times n$ 矩阵与一个 $n \times n$ 矩阵相乘所需的时间。我们不讨论其余模式（即带有 $|F| = 3$ 的模式和带有 $d \in F$ 的模式），将其留待后续工作。

我们从模式 $P_\emptyset$ 开始。

引理 10. 模式 $P_\emptyset$ 可以在线性时间内检测出来。

证明. 要检验给定的顶点排序 τ 是否避开模式$P_\emptyset$，我们首先对每个顶点的邻接表进行排序，使某个顶点的邻居在邻接表中的顺序与在 τ 中的顺序一致。然后，每条满足$u <_\tau v$ 的边 uv 都可以视为一对括号（$_u$ 和 $)_v$。接着，只需验证 τ 是否构成一个良括号序列。众所周知，只需使用一个简单的栈扫描一次排序 τ 即可完成验证。□

现在我们来考虑模式 P_a。其主要思路是在唯一的一次扫描过程中，检查当前顶点 i 是否为模式 P_a 的第三个顶点。

命题 15. 模式 P_a 可以在线性时间内检测出来。

证明。我们证明，算法 4 可以在线性时间内检测排序是否包含模式 P_a。设在主 for 循环的第 i^{th} 步开始时，列表 ACTIVE 的值为 $ACTIVE_i$。

如果答案是 YES，这意味着对于某个 i，我们找到了一个 j，使得$NOT($ $\{x \in N^-(i)$ 且 $x <_\tau j\} \subseteq N^-(j))$，因此存在 $x <_\tau i$，使得 $xi \in E(G)$ 且 $xj \notin E(G)$。因此，$x \in ACTIVE_i$；又因为 j 也在 $ACTIVE_i$ 中，且不是列表中的第一个元素，并且至少有一个比 i 大的邻居，所以它构成了 P_a 模式。

现在来考虑 NO 情况下的主不变式。

主不变式：在 for 循环第 i^{th} 次迭代结束时，排序τ 在顶点 $\alpha <_\tau \beta <_\tau \gamma <_\tau \delta$ 上不包含任何模式 P_a，其中 $\gamma \leq_\tau i$。

我们用归纳法证明。借助 $i = 3$，算法检查顶点 1、2、3 上是否存在模式 $P_a\, x$，且 $i <_\tau x$。因此，对于 $i = 3$，不变式成立。

现在假设在 for 循环的第 i^{th}（$i \geq 3$）次迭代结束时，排序 τ 中不包含任何出现在顶点 $\alpha <_\tau \beta <_\tau i <_\tau \delta$上的模式 P_a。

如果 β 是算法计算出的最大真正活跃顶点（$\beta = j$），那么该模式已经检查过了，这就产生了矛盾。

否则 $\beta <_\tau j$。由于 $\alpha i \in E$（G），我们也知道 $\alpha j \in E$（G）。因此，我们得到一个模式 P_a，其中 $\alpha <_\tau \beta <_\tau j <_\tau \delta$ 且 $j <_\tau i$，这与归纳假设矛盾。

复杂度分析 预处理需要 O（$n+m$）。算法本身只需对 τ排序进行一次扫描。在 for 循环中，每个顶点的邻域也只会被考察一次。因此，整个算法可在 O（$n+m$）内完成。

需要注意的是，当扫描列表 ACTIVE 以查找第一个真正活跃顶点（如果存在）时，不满足邻域条件的顶点

输入： G 由其邻接表给出的图，以及 τ 顶点的一个全序 输出：(G, τ)是否包含模式 P_a邻接表假设按 τ递增顺序排列。不失一般性，假设 $\tau = 1$, 2, . . . , $nACTIVE$ 是一个初始化为 $\emptyset$的顶点列表。对于每个顶点 i，将其邻接表划分为两个列表 $N^-(i) = \{x \in N(i)$，使得$x < i\}$ ，并且 $N^+(i) = \{x \in N(i)$，使得 $i < x\}$。对于 $i = 1, 2$，执行：如果 i 有一个邻居 x 满足 $3 \leq_\tau xappend(i, ACTIVE)$。结束循环。对于 $i = 3$ to $n - 1$ ，执行：如果 $N^-(i) \neq \emptyset$ and $ACTIVE \neq \emptyset$ and $j \leftarrow max_\tau\{p \in ACTIVE$ $that$ has a $neighbour$ q $with$ $i <_\tau q\}$ $exists$ and $j \neq First(ACTIVE)$，则 // j 是在步骤 i 中最大的真正活跃顶点；如果 $NOT(\{x \in N^-(i)$ $with$ $x <_\tau j\} \subseteq N^-(j))$，则停止。输出“是，已找到 P_a ”。结束。否则，对于每个 $u \in N^-(i)$：从 $N^+(u)$中删除 i 。如果 $N^+(u) = \emptyset$ ，则从 $ACTIVE$中删除 u。结束。如果 $N^+(i) \neq \emptyset$ ，则$append(i, ACTIVE)$。结束。结束。结束。结束。输出“否，未找到 P_a ”。算法 4：检测模式 P_a

条件将在 for 循环结束时从 ACTIVE 列表中删除，因此不会再次被考虑。□

我们利用对称性处理 P_c ：若要在图 G 中检测 P_c ，我们在线性时间内将该图镜像，然后在镜像后的图中检测 P_a 。现在我们聚焦于模式 P_b。

命题 16. 可以在线性时间内检测模式 P_b 。

证明。为证明此命题，我们给出并分析一个专用检测算法 5

注意： $ACTIVE$ 不仅是一个双向链表，而且对于属于 $ACTIVE$ 的每个顶点 u ，我们还需要维护一个指针，指向其

输入： G 一个由其邻接表给出的图，以及 τ 一个顶点全序 输出：(G,τ) 是否包含模式 P_b邻接表应按 τ递增顺序排列。不失一般性，设 $\tau = 1$、2、.....、 $nACTIVE$是一个初始化为 $\emptyset$ 的顶点双向链表。对于每个顶点 i，将其邻接划分为两个列表 $N^{-}(i) = \{x \in N(i)$，使得 $x < i\}$ ，以及 $N^{+}(i) = \{x \in N(i)$，使得 $i < x\}$。对于 $i = 1$，2，执行：如果 i $admits$ a $neighbour$ x $with$ $3 \leq_\tau x$ ，则$append(i, ACTIVE)$。结束。对于 $i = 3$ to $n-1$ ，执行：对于每个 $u \in N^{-}(i)$：从 $N^{+}(u)$ 中删除 i ；如果 $N^{+}(u) = \emptyset$ ，则从 $ACTIVE$ 中删除 u；如果 $|N^{-}(i)| \geq 2$ and $ACTIVE \neq \emptyset$ ，则$j \leftarrow First(N^{-}(i))$，从 $N^{-}(i)$ 中删除 j ；如果 $ACTIVE]j, i[\neq N^{-}(i)$，则停止，输出“是，已找到 P_b ”；如果 $N^{+}(i) \neq \emptyset$ ，则$append(i, ACTIVE)$。结束。结束。输出“否，未找到 P_b ”。

算法 5：检测 P_b

放置于 $ACTIVE$ 中。因此， $ACTIVE]j, i[$ 是子列表 $ACTIVE$，由 $ACTIVE$ 中满足 $j < k < i$ 的顶点 k 构成。

算法证明

当算法输出 YES 时，我们找到了一个顶点 i ，对应于图 20 中的顶点 3：由于 $N^{-}(i) \neq \emptyset$， i 有一个邻居 $u < i$，并且由于 $ACTIVE]j, i[$ 与 $N^{-}(i)$ 不同，存在一个与 i 不相邻且满足 $v < i$ 的顶点 v。此外，由于 $v \in ACTIVE$ 与 i 不相邻， v 有一个邻居 $w >_\tau i$。因此， $\{u, v, i, w\}$ 给出了模式 P_b。

在主 for 循环的第 i^{th} 次迭代结束时，若存在一个尚未检测到的 P_b 模式，$\alpha <_\tau \beta <_\tau i <_\tau \gamma$ 满足 $\beta i \notin E(G)$，则 $j <_\tau \beta <_\tau i <_\tau \gamma$ 也同样是这样的模式。

由于在比较列表 $ACTIVE]j, i[$ 和 $N^{-}(i)$ 时，边 βi 已在 $\beta \in ACTIVE]j, i[$ 中检查过，因此产生了矛盾。

复杂度：预处理需要 $O(n+m)$。按 τ, $ACTIVE[j,i[$ 和 $N^-(i)$ 排序的两个列表在遇到第一个差异时即结束比较，其开销可计入 $N^-(i)$。因此，在 for 循环中，一个顶点的邻域只会被考虑一次。所以，整个算法可以在 $O(n+m)$ 内完成。□

引理 11。可以在 $O(n \cdot m)$ 时间内检查模式 P_{ab}。

证明。对于此模式，我们也对排序 τ进行一次唯一的扫描。将当前顶点作为图 20 中的潜在顶点 3 进行分析。我们还维护一个顶点列表 $ACTIVE$，其中的顶点在排序中当前顶点之后或与其相邻的位置有邻居。

在每一步 i，我们都需要检查：在 $ACTIVE$中，i 的邻居和非邻居是否构成完全二部图。此项检查可在线性时间内完成，因此复杂度为 $O(n \cdot m)$。□

对于模式 P_{ab}，使用以下归约无法确定是否存在一个线性时间算法。

命题 17。模式 P_{ab} 的识别难度与三个顶点上的余可比模式相同。

证明。设 G 是一个图，且 τ 是 $V(G)$ 上的一个排序。我们通过向 G 添加一个全邻顶点 u，从 G 构造一个新图G'。注意，如果 $\tau' = \tau + u$ 不包含任何 P_{ab}，那么 τ 就不包含余可比模式。□

正如 [9]，中所指出的，目前尚无已知的线性时间算法可以检测余可比模式。