arXiv:2302.05575v3 [cs.CC] 3 Oct 2023

组合数据上的组合算法：预层上的层判定

ERNST ALTHAUS

计算机科学研究所 约翰内斯·古腾堡大学 施陶丁格路 9号 55128，美因茨，德国 ernst.althaus@uni-mainz.de

BENJAMIN MERLIN BUMPUS*

计算机与信息科学及工程 佛罗里达大学 纽厄尔路432号，盖恩斯维尔，佛罗里达州 32603，美国。benjamin.merlin.bumpus@gmail.com

JAMES FAIRBANKS†

计算机与信息科学及工程，佛罗里达大学，纽厄尔路432号，盖恩斯维尔，佛罗里达州32603，美国。fairbanksj@ufl.edu

DANIEL ROSIAK

美国国家标准与技术研究院，100 BureauDr，盖瑟斯堡，MD 20899，美国，danielhrosiak@gmail.com

通信地址：计算机与信息科学及工程，佛罗里达大学，纽厄尔路432号，盖恩斯维尔，佛罗里达州32603，美国。

*DARPA ASKEM和青年教师奖项目，资助编号为HR00112220038和W911NF2110323

†DARPA ASKEM和青年教师奖项目，资助编号为HR00112220038和W911NF2110323

摘要。 算法研究者很清楚，在组合式（甚至递归式）图上进行计算时，快速动态规划算法往往是正确的选择。本文开启了将这一广为人知的直觉推广到更广泛数学结构的研究。我们通过采用范畴论视角实现这种横向普遍性，该视角使我们能够证明：(1)结构化分解（许多图分解近期提出的一种抽象推广）在数据范畴（粘合范畴）上定义了格罗滕迪克拓扑；(2)任何可表示为相对于这些拓扑的层 的计算问题，都能在线性时间内判定，适用于那些允许宽度有界的分解且分解形状的反馈顶点数有界的输入类。这立即为任意 C-集范畴中的对象带来了算法；以下仅举数例：对称图、有向图、有向多重图、超图、有向超图、数据库、单纯复形、圆形端口图以及半边图。

因此，我们开始将层论、结构图论和参数化复杂性理论中的工具结合起来；我们相信，这将为动态规划算法的一般代数理论提供一种非常富有成效的方法。最后，我们将理论结果与在*AlgebraicJulia* 生态系统中对主要算法贡献进行的具体实现相结合。

关键词： 参数化复杂性、动态规划、层论、范畴论

1. 引言

正如 Abramsky 和 Shah 所指出的， [1], “计算基础中的两条主要组织原则”是：**结构**和**能力**。前者关注组合性和语义，后者则聚焦于表达能力和计算复杂性。

迄今为止，这两个领域基本上仍然彼此分离。这部分是由于数学和语言上的差异，也有一部分是因为人们认为它们的研究重点不同。然而，我们认为，其中许多差异只是表面上的，而且“能力”领域一直都高度重视组合性。事实上，尽管直到动态规划和图分解出现之前，这一点尚未被正式表述出来，但它们清楚地证明了组合性在该领域中的重要性。因此，当前故事的主要角色是：

(SC) 图论中以**结构**组合性形式出现的组合性，即图分解和图宽度度量（通过将图分解为更小、更简单的组成部分 [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21]），

(AC) 由参数化复杂性理论中复杂的动态规划例程所体现的**算法**组合性 [22, 23, 24, 25, 26],

(RC) 以及代数拓扑中体现的**表示**组合性[1]，并且几乎贯穿数学的其余领域，以层 [28, 29, 30] 的形式出现（通过这种形式，人们可以系统地

[1]我们选择“表示性”一词，以此致意 Leray 对层理论应有何种形态的 [27] 1946理解：“我们拟简要说明，我们借以研究一个空间拓扑的方法

为“空间”赋予数据，从而能够轻松追踪局部数据与全局数据之间的相互作用）。

对图分解及其相关“宽度度量”的研究一直是一个极其活跃的领域，催生了无数细微不同的分解方法 [6, 7, 4, 23, 22, 2, 31, 12, 13, 14, 15, 16, 18, 19, 20, 21, 11, 17, 32]。从计算角度来看，这些概念对于动态规划至关重要，因为它们可以被视为具有组合结构的图的数据结构 [22, 23, 25, 26]。

另一方面，层为向空间（或某种“类似空间”的对象）附加数据提供了标准的数学结构；这一结构还包含限制映射，用以编码数据之间某种局部约束或关系，并规定如何将相容的局部数据组合起来，形成整体结构。因此，层结构使我们能够推理这样的情形：我们希望追踪相容的局部数据如何拼接成整体数据赋值。到 20 世纪 50 年代，层已逐渐成为一种得到广泛应用的工具，最初在代数拓扑、多变量复分析和代数几何中发挥了决定性作用。作为处理各种局部到整体问题的框架，层的强大能力使其在众多领域和应用中发挥作用，包括传感器网络、目标追踪和动力系统（更多示例和参考资料请参见 Rosiak 的教科书 [29]）。

我们的贡献是展示如何运用层论和范畴论的工具，弥合“结构”与“能力”两个学术群体之间的鸿沟。实际上，我们的主要算法成果（定理1.1）是通过融合这三种视角得出的一个元定理。粗略地说，它指出，任何能够表述为一个层的判定问题，都可以在线性时间内求解，适用的输入类可由一些组成部分通过复合构造而成，且这些组成部分的内部复杂度有界。更具体地说，这一结果为一大类数学结构上的问题（例如H-着色及其推广）给出了线性时间（固定参数可解时间）算法；其中一些例子包括：1. 数据库，2. 简单图，3. 有向图，4. 有向多重图，5. 超图，6. 有向超图（即 Petri 网），7. 单纯复形，8. 圆形端口图 [33] 以及 9. 半边图。

为了确立（甚至仅仅表述）我们的主要算法结果，首先必须克服两个极具挑战性的难题。第一，我们需要弄清楚，数学对象（不一定是图）展现出组合结构意味着什么。第二，为了将计算问题编码为层，必须先为输入对象赋予一种拓扑 概念，而且这种拓扑还必须具有足够的表达能力，能够编码输入实例的组合结构。

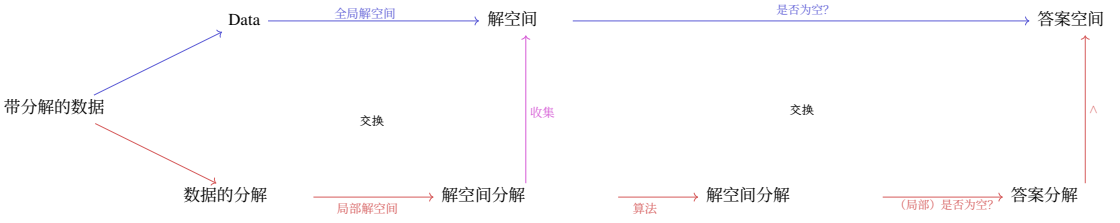
我们通过摒弃狭义的图 分解概念，转而采用更广泛且与对象无关的对应概念，即结构化分解，从而解决了第一个问题。结构化分解是近期提出的一种范畴论推广，将图分解推广到任意范畴的对象 [34]。

可以适用于研究表示的拓扑”。Gray [28] 指出，这是“‘faisceau’一词首次以近似于其当前数学含义的方式出现”。

从图论视角转向范畴论视角，也使我们能够通过将分解视为范畴中对象的覆盖 来克服第二个障碍。更确切地说，我们证明，结构化分解为数据范畴（即粘合范畴）赋予了格罗滕迪克拓扑。这是我们的主要结构性结果（推论3.8）。这使我们能够将关于层 相对于分解拓扑的概念形式化，从而表达“其组合结构与其输入的组合结构相兼容的计算问题”这一模糊概念。

1.1. 路线图

本文将逐步建立必要的范畴论表述，使下图成为精确而形式化的交换图（此图为图表 6）。



(1)

这张图中的箭头将是关联适当范畴的函子，它通过简洁地编码我们先前识别出的三种组合性，将**结构**与**能力**联系起来。底行中的范畴对应于**结构组合性**。彩色路径对应不同的算法：

1. 顶部路径（蓝色）对应暴力算法；2. 向上穿过的“中间路径”（先红色，再粉色，最后蓝色）对应组合算法，但它是较慢的算法；3. 底部路径（红色）对应快速的（固定参数可解时间）动态规划算法。

各个方块（以及整张图）的交换性体现了算法的正确性，因为这意味着该算法（就其返回的解而言）等价于暴力方法；这就是**算法组合性**。我们所有推理所依赖的关键前提，是标记为“全局解空间”的箭头表示一个层。这就是**表征组合性**，它意味着整张图是交换图。

算法组合性本身就很有趣，但只有与运行时间保证相结合时才有用。为此，我们已经可以从图表 1 中推导出关于算法子程序运行时间的精确陈述。事实上请注意，标记为“全局解空间”的箭头（分别是“局部是否为空？”）对应于计算局部解空间（分别是局部回答判定问题），以便将数据分解转化为解空间的分解（分别是布尔值）。这些箭头对应于进行线性[2] 次

[2]（关于分解中分量片段数量的线性函数）

，因此可以直觉地认为，运行时间将由那个颇具暗示性的“算法”箭头所主导。我们的大部分技术工作都将致力于证明这个函子存在（定理 4.1），并证明它可以高效计算（定理 1.1）。这个函子不仅是一个合理的选择，而且在某种意义上也是标准的选择：运用标准的范畴论工具，恰恰就会得到它[3]。所有这些论证都将以表征组合性为基础：它们要求‘全局解空间’是一个层。

鉴于上述背景，我们现在可以非正式地陈述我们的主要算法贡献（定理1.1）。该定理指出，对于任何一个以相对于数据分解所给拓扑的层形式编码的问题，都存在一种算法来解决其关联的判定问题，且所需时间的增长

1. 与分解中组成部分的数量呈线性关系，并且
2. 关于组成部分的内部复杂度有界（通常呈指数增长）。

该定理及其所实现的计算任务，正式表述如下。（第2节和第3节将逐步构建理解这一表述所需的全部基础。）

C − SHEAFDECISION
输入： 位点 (C，Dcmp) 上的层 $\mathcal{F}\colon \mathsf{C} \to$ 有限集范畴^{op}，其中 C 是一个小型粘着余完备范畴，对象 $c \in \mathsf{C}$，以及一个关于 c 的 C 值结构化分解 d。**任务：** 令 **2** 为二对象范畴 $\bot \to \top$，并令 dec: FinSet $\to$ **2** 为将集合映射到：若为空则映射到 $\bot$，否则映射到 $\top$ 的函子，计算 $\mathrm{dec}^{op}\mathcal{F}c$。

定理 1.1. 设 G 是一个有限、非自反、有向且没有反向平行边的图，并且每对顶点之间至多有一条边。设 D 是一个小型粘着余完备范畴，设 $\mathcal{F}\colon \mathsf{D}^{op} \to$有限集范畴是一个预层，且 C 是 $\{\mathsf{D}, \mathsf{D}_{mono}\}$ 之一。若 $\mathcal{F}$是位点 $(\mathsf{C}, \mathrm{Dcmp}\,|_{\mathsf{C}})$ 上的层，并且给定一个算法 $\mathcal{A}_{\mathcal{F}}$，可在时间 $\alpha(c)$ 内计算 $\mathcal{F}$ 在任意对象 c 上的值，那么存在一个算法：给定对象 $c \in \mathsf{C}$ 的任意 C 值结构化分解 $d\colon \int G \to \mathsf{C}$，以及 G 的反馈顶点集 S，该算法可在时间内计算 $\mathrm{dec}\mathcal{F}c$

$$O\Big(\max_{x \in VG} \alpha(dx) + \kappa^{|S|}\kappa^2\Big)|EG|$$

其中 $\kappa = \max_{x \in VG} |\mathcal{F}dx|$。

定理 1.1 具有三大主要优点：

1. 它使我们能够重新获得一些关于图的算法结果（例如动态规划算法、用于H-着色和用于H-反射着色的算法[4]），并且

[3]该函子源自图表余极限 $\dashv$const 伴随所给出的单子 。在这里，给定一个固定的图H，需要判断给定的图G 是否存在到H的自反同态（其中，自反同态是指一个顶点映射$f\colon VG \to VH$，使得对于G中的所有顶点对（x，y），若$f(x)\,f(y) \in EH$，则$xy \in EG$。

2. 它使我们能够将分解及其上的动态规划推广到其他类型的结构（而不仅仅是图），并且

3. 它很容易实现。

现在，我们将简要展开说明上述最后两点。请注意，由于定理 1.1 适用于任意粘合范畴，我们自然可以针对以任意一个 C-集合范畴 [35, 36]编码的众多其他结构获得算法。这是一类非常庞大的结构，以下仅是其中几个例子：1. 数据库，2. 简单图，3. 有向图，4. 有向多重图，5. 超图，6. 有向超图，7. 单纯复形，8. 圆形端口图 [33]以及 9. 半边图。

我们注意到，范畴论视角使我们几乎毫不费力地将元理论结果（具体而言，与树形分解相关的特殊情形）与一个实用且可运行的实现 [37] 在 AlgebraicJulia 生态系统中 [38]结合起来。

最后，从高层次来看，我们的方法还带来了另一项洞见（Bodlaender 和 Fomin [39] 已经指出过）：重要的并不是输入的分解的宽度；相反，解空间的分解的宽度才是算法界限的关键。

概要 我们的结果依赖于以下要素：1. 将计算问题编码为函子；2. 通过图表描述结构组合性（特别是一类适用于算法操作的特殊图表，称为 结构化分解）；3. 证明这些结构化分解能够产生格罗滕迪克拓扑；最后，4. 证明我们的主要算法结果。在第 2 节中，我们解释如何将计算问题视为函子。在第3.1 节中，我们介绍结构化分解的背景，并证明它们能够产生格罗滕迪克拓扑（推论 3.8）。我们在第 4 节中建立算法元定理（定理1.1），并在第 [37] 4.3 节中讨论这些结果在 AlgebraicJulia 中的实际实现。最后，我们在第 7 节中讨论我们的贡献以及未来工作的机会。

致谢：作者谨向 Evan Patterson 致谢，感谢他富有启发性的讨论，这些讨论影响了本文的撰写；也感谢他提供的关键见解，这些见解改进了 AlgebraicJulia [38] 中结构化分解的实现。

1.2. 符号约定

本文中，我们假设读者熟悉 Riehl 的教科书中常见的范畴论记号 [40]。我们采用层论中的标准记号；对于本文或附录 A中未明确定义的概念，请参阅 Rosiak 的教科书 [29]。最后，关于图论的背景知识，我们遵循 Diestel 的记号 [3]；而参数化复杂性理论中的标准术语和概念，则采用 Cygan 等人的教科书（例如其中）的用法 [25]。

2. 作为函子的计算问题

计算问题是将数据——视为解空间 ——分配给某类输入对象。我们将其视为函子$\mathcal{F}\colon \mathsf{C} \to \mathsf{Sol}$，它将某个范畴 C 中的对象映射到某个恰当选取的解空间范畴 Sol 中的对象。通常，由于解空间大得难以处理，与其计算整个解空间，不如采用更近似的表示形式来描述问题，例如判定问题、优化问题或枚举问题等。比如，可以将一个 **$\mathcal{F}$-判定问题** 表述为如下复合形式

$$\mathsf{C} \xrightarrow{\mathcal{F}} \mathsf{Sol} \xrightarrow{\mathsf{dec}} \mathbf{2}$$

其中 dec 是一个取值于 **2** （行走箭头范畴）的函子，它将解空间映射到 $\bot$ 或 $\top$，具体取决于这些解空间见证的是 $\mathcal{F}$ 的肯定实例还是否定实例。

图论学家熟悉的一些例子 包括 GAPHCOLORING$_n$、VERTEXCOVER$_k$ 和奇数圈横断集（记作 OTC$_k$），它们都可以很容易地编码为取值于有限集范畴 FinSet 的反变函子，下面将对此进行说明。

GRAPHCOLORING$_n$ 是最容易定义的问题；它就是可表函子 SimpFinGr$(-,\ K_n)$: 有限集范畴将每个图 G 映射到从 G 到 n 个顶点的非自反完全图的所有同态所组成的集合。通过令 dec: 有限集范畴 $\to \mathbf{2}$ 成为一个函子，它将任意集合映射为 $\bot$当且仅当该集合为空，即可将其转化为判定问题。

对于 VERTEXCOVER$_k$ 和 OTC$_k$，我们将改为考虑以下子范畴

$$\mathsf{SimpFinGr}_{mono} \hookrightarrow \mathsf{SimpFinGr}$$

该子范畴由有限简单图以及它们之间的单态射（子图）构成。注意，若H' 是图 G' 的子图——由满足某个我们感兴趣的子图封闭性质P 的单射 $g\colon H' \hookrightarrow G'$ 见证——且$f\colon G \hookrightarrow G'$ 是任意单态射，那么沿f 拉回g 将得到 G 的一个子图，该子图也满足性质P （因为P 是子图封闭的）。特别地，这表明对于任何此类子图封闭性质P （具体而言，包括我们举的两个例子：大小至多为k的顶点覆盖或奇圈横截集），以下映射是一个到偏序集范畴中的逆变“拉回函子”

$$F_P\colon \mathsf{SimpFinGr}_{mono} \to \mathsf{Pos}.$$

令U 为将偏序集映至其底层集合的遗忘函子，则可以将相应的判定问题（尤其包括 VERTEXCOVER$_k$ 和 OCT$_k$）表述为以下复合

$$\mathsf{SimpFinGr}_{mono}^{op} \xrightarrow{F_P} \mathsf{Pos} \xrightarrow{U} \mathsf{FinSet} \xrightarrow{\mathsf{dec}} \mathbf{2}.$$

3. 复合数据与 Grothendieck 拓扑

参数化复杂性 [25]是复杂性理论的一个二维框架，其核心观点是：分析运行时间时，不应只考虑输入的总规模，还应考虑

输入的其他参数（例如衡量组合结构 [25]的指标）。在这里，我们通过图表来表示组合结构：我们将一个对象$c \in \mathsf{C}$ 视为图表$d\colon \mathsf{J} \to \mathsf{C}$ 的余极限，并认为它是由d 分解成更小组成部分的结果。本节分为两部分。在第3.1 节中，我们将说明图表如何在粘合范畴上诱导出格罗滕迪克拓扑。然后在第3.2 节中，我们将聚焦于一类特殊的图表（适用于算法处理），称为结构化分解 [34]。粗略地说，它们由范畴中的一组对象以及一组将这些对象关联起来的跨度构成（正如图中的边将顶点关联起来一样）。

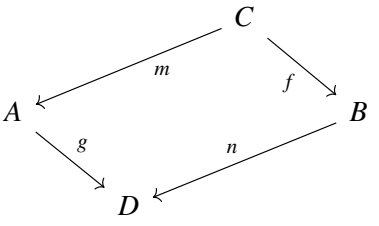
3.1. 作为格罗滕迪克拓扑的图表

本节将用到粘合范畴。我们将其视为这样一种数据范畴：“其中单态射的推出存在，且其‘行为大致与集合范畴中的情况相同’，或者等价地说，与任意拓扑斯中的情况相同。” [41] 粘合范畴涵盖了众多数学结构，从拓扑空间（实际上也包括任意拓扑斯）到熟悉的组合结构，例如：1. 数据库，2. 简单图，3. 有向图，4. 有向多重图，5. 超图，6. 有向超图（即 Petri 网），7. 单纯复形，8. 圆形端口图 [33] 以及 9. 半边图。

定义 3.1。（粘合范畴 [36]）
若范畴 C 满足以下条件，则称其为**adhesive（粘合的）**：

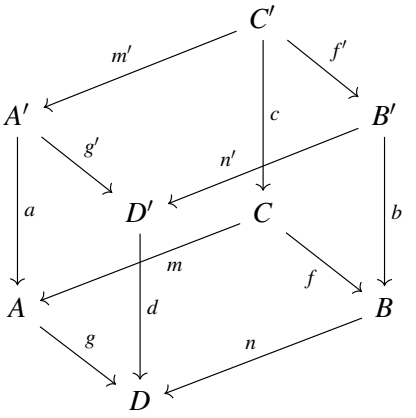
1. 范畴 C 存在沿单态射的推出；2. 范畴 C 存在拉回；3. 沿单态射的推出满足 van Kampen 条件。

反过来，形如以下形式的推出交换图



称为**van Kampen** ，如果对于下图所示的任意交换立方体，且其底面为

上方的方形，



以下结论成立：如果后侧面是拉回，那么前侧面是拉回，当且仅当顶面是推出。

我们将关注态射均为单态射的图表；我们称之为子单态图表。

定义 3.2. C **中形状为** J **的子单态图表**是一个图表$d : \mathsf{J} \to \mathsf{C}$，它保持单态射，且其定义域是一个所有态射均为单态射的有限范畴。我们称范畴 C 为**粘性余完备**，如果 (1) 它是粘合范畴，且 (2) C 中每个子单态图表都有余极限。

以下结果（[34, 引理 5。10] 此处重述以便查阅）指出，我们可以以函子性的方式，将每个粘性余完备范畴 $\mathcal{C}$ 中的对象 c 与其余极限为 c 的子单态图表集合相关联。

引理 3.3。([34])
设 C 是一个小型粘着余完备范畴。对于任意箭头$f\colon x \to y$（属于 C），以及任意图表$d_Y\colon \mathsf{J} \to \mathsf{C}$，若其余极限为 y，则可通过对f与d_Y的余极限余锥中的各态射逐点拉回，得到一个图表 $d_X\colon \mathsf{J} \to \mathsf{C}$，其余极限为 x Λ。

$$\begin{array}{ccc} X = \operatorname{colim} d_X & \xrightarrow{f} & y = \operatorname{colim} d_Y \\ \Uparrow & \lrcorner & \Uparrow \Lambda \\ d_X J & \Longrightarrow & d_Y J \end{array}$$

请注意，在引理 3.3 中，我们不能不要求图表是单态的，因为尽管粘合范畴具有所有单态跨度的推出，但它们不一定具有任意跨度的推出（参见定义 3.1）。

正如我们现在将要看到的，粘性余完备范畴具有足够的结构，使我们能够定义一种格罗滕迪克拓扑，其中覆盖由单态子范畴的余极限给出 [5]。我们将利用这一结果证明（推论3.8），结构化分解产生所需的格罗滕迪克拓扑。

定理 3.4。 若 C 是一个小型粘着余完备范畴，记 Λ_d 为任意图表d的余极限余锥，则以下映射通过拉回构成一个函子。

$$\begin{aligned}&\mathsf{subMon}\colon \mathsf{C}^{op} \to \mathbf{Set}\\ &\mathsf{subMon}\colon c \mapsto \{\Lambda_d \mid d \text{ a submonic diagram and colim} d = c\}\\ &\qquad\qquad \bigcup\{\{f\} \mid f\colon a \xrightarrow{\cong} c \text{ an iso }\}.\end{aligned}$$

此外，我们还有：

- 对于任意这样的 C，二元组 $(\mathsf{C}, \mathsf{subMon})$ 是一个位点，并且
- 记 C_{mono} 为 C 的子范畴，它与 C 具有相同的对象，但仅包含 C 的单态射，则以下二元组是一个位点

$$\left(\mathsf{C}_{mono}, \mathsf{subMon}|_{\mathsf{C}_{mono}}\right).$$

证明：
.....这一事实subMon 由拉回构成一个反变函子，可由拉回保持同构这一观察以及引理3.3得出。

现在请注意，只需证明subMon 定义了一个格罗滕迪克预拓扑（定义 A.1），因为已知每个格罗滕迪克预拓扑都会确定一个真正的格罗滕迪克拓扑 [29]。我们仅对第一种情况进行证明（第二种情况的证明类似），为此，我们直接验证公理（参见附录 A.1 A 中的定义）。首先注意，公理 **(PT1)** 根据 subMon的定义显然成立。公理 **(PT2)** 也显然成立，因为我们刚刚已在本定理的第一部分证明 subMon 是一个由拉回构成的反变“函子”。公理**(PT3)** 也可轻易证明如下：

- 由于图表的图表仍是图表，因此余极限的余极限也是余极限，所以得到的余极限余锥属于覆盖；并且
- 如果给定任意一个由同构组成的单元素覆盖 $\{f\colon b \xrightarrow{\sim} c\}$，那么有两种情况：1. 如果给定另一个这样的覆盖 $f'\colon a \xrightarrow{\sim} b$，那么复合 ff' 是到 c 中的一个同构，因此属于覆盖性；2. 如果 b 上的覆盖是子单态图表 d的余极限余锥 Λ_d，那么将余锥的分量（它们是单态射）与同构 f 复合，就确定了 c 的一个子对象图，而该图表将以 c 为其余极限，正如所需。

[5]对于关注规模问题的读者，请注意：(1) 给定范畴 C 和 D，只要 C 和 D 本身也是小的，函子范畴 $[C, D]$就是小的；(2) 由于我们关注的是定义域只有有限个对象和态射的图表，因此，以某个给定对象为余极限的图表构成的集合确实是一个集合。

□

回顾第1节，某些计算问题（例如 VERTEXCOVER$_k$ 和 OCT$_k$）是在有限简单图及其间单态射构成的范畴 SimpFinGr$_{mono}$ 上提出的。事实上请注意，如果我们试图在范畴 SimpFinGr 上提出这些概念，就会失去控制顶点覆盖（分别为奇圈横截集等）基数的能力：尽管沿满态射对顶点覆盖（分别为奇圈横截集等）作拉回后仍是顶点覆盖（分别为奇圈横截集等），但拉回得到的顶点覆盖大小可能会任意增大。因此，如果我们想将计算问题（包括上述特定问题，也包括更一般的问题）视为层，那么能够将结构化分解用作“单态射范畴”（即某个黏着范畴 C 上的 C$_{mono}$ ）上的格罗滕迪克拓扑会特别有用；这类范畴通常不具有推出。

旁注 请注意，与定理 3.4 和推论 3.8 中的思路类似的方法，可用于在“合成空间”（即编码为流形结构化分解的混合维流形）上定义层。这里的想法是，人们希望在由不同维度的流形粘合而成的拓扑空间上定义层，同时记住，所得对象应被视为“合成混合维流形”（即流形的结构化分解）。结构化分解可用作拓扑这一事实意味着，我们可以谈论能够意识到 复合拓扑空间需要被视为混合维流形的层。这超出了本文的讨论范围，但却是一个令人着迷、值得进一步研究的方向。

3.2. 结构化分解

出于算法目的，使用结构化分解而非任意图表来定义格罗滕迪克拓扑会更方便。这有两个原因：（1）结构化分解是一类特别优良的图表，当目的是通过递归算法构造余极限时，在算法层面更容易处理；（2）尽管诚然，给定图表的余极限锥后，总能将图表转化为跨度图表（正如我们在推论3.8中所论证的），但这一操作的计算开销很大（尤其是，正如我们稍后将讨论的，当处理解空间等规模大到难以应付的对象的分解时）。解空间）。

结构化分解的背景 结构化分解 [34] 是许多组合不变量的范畴论推广，例如树宽 [6, 7, 4], 分层树宽 [19, 20], 余树宽 [42], $\mathcal{H}$-树宽 [21] 以及图分解宽度 [18]，这些不变量在图论和参数化复杂性中发挥了核心作用。

直观地说，结构化分解应被视为广义图：图是一个定义在一个集合的元素上的关系，而结构化分解则是一个定义在一组对象上的广义关系（即一组跨度），这些对象属于某个范畴。例如，请参见图1，其中绘制了一个形状为K_3（三顶点完全图）的图值结构化分解。

形式上，对于任意范畴 C，我们将 C 值结构分解定义为范畴 C 中某些类型的图表，如下述定义所示。就本文而言，我们只需要

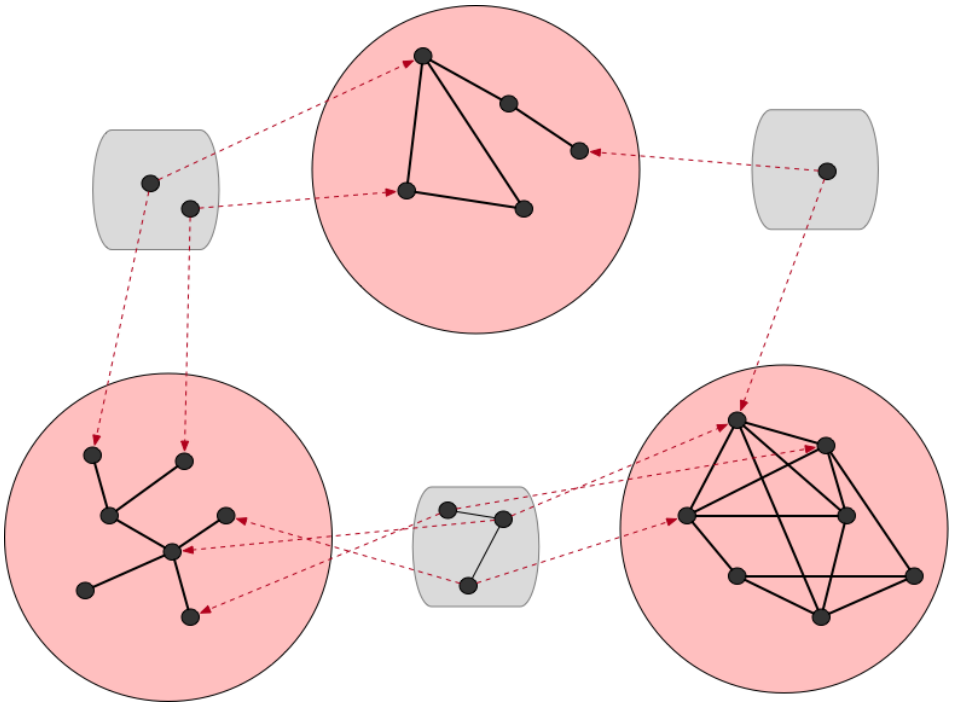


图 1. 形状为 K_3 的 $\mathrm{Gr_H}$-值结构化分解。跨度（图中为单态）以红色点线（逐分量地）绘出。袋以粉色突出显示，粘合以灰色突出显示。

由所有箭头均为单态的图表给出的结构化分解这一特殊情形；不过，为求完整，我们指出，结构化分解理论并不依赖于这一限制。因此，为了便于阅读，并且由于本文档只讨论单态结构分解，我们将省略“单态”这一修饰语，改称结构化分解（或简称“分解”）。

定义 3.5.（单态结构化分解）
给定任意有限图 G，将其视为函子 $G\colon \mathsf{GrSch} \to \mathsf{Set}$，其中 GrSch 是由以下内容生成的双对象范畴

$$E \underset{t}{\overset{s}{\rightrightarrows}} V$$

可以构造一个范畴 $\int G$，其中每个顶点对应一个对象G，每条边对应一个对象G，并且每条边G 都通过一个跨度与其源顶点和靶顶点相连。这种构造是更一般的格罗滕迪克构造的一个实例。现在，固定一个基范畴K，我们将形状为K**的**G 值结构化分解（见图1）定义为如下形式的图表

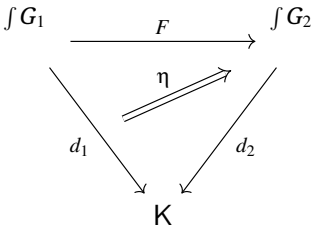
$$d\colon \int G \to \mathsf{K}$$

其态射在 K 中均为单态。给定任意顶点v （属于G），我们称 K 中的对象dv 为**d**的袋，其指标为**v**。给定$e = xy$ （属于G）中的任意边，我们称de 为以e 为指标的**粘合**，并称跨度$dx \leftarrow de \rightarrow dy$ 为以e为指标的**粘合跨度**。

定义 3.6。（结构分解的态射）
一个 K **值结构化分解从**d_1 **到** d_2 的态射是一对

$$(F, \eta) \colon \left(\int G_1 \xrightarrow{d_1} \mathsf{K}\right) \to \left(\int G_2 \xrightarrow{d_2} \mathsf{K}\right)$$

如以下图表所示，其中F 是一个函子$F \colon \int G_1 \to \int G_2$，且 η 是一个自然变换$\eta \colon d_1 \Rightarrow d_2 F$，如以下图表所示。



这种抽象的意义在于，我们现在可以用统一的方式讨论许多不同类型对象的分解。事实上，这正是本文所采用的方法：我们将计算问题的输入的组合结构与该问题的解空间所对应的组合结构联系起来。为此，我们利用了这样一个事实：结构化分解范畴的构造 [34, 命题 3.3] 具有函子性 [34, 推论 3.4]。特别地，存在一个函子

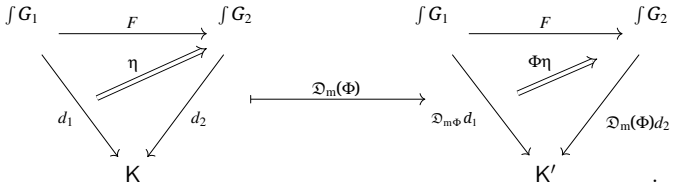
$$\mathfrak{D}_{\mathrm{m}} \colon \mathsf{Cat} \to \mathsf{Cat}$$

它将任意范畴C 映射到范畴 $\mathfrak{D}_{\mathrm{m}}$C 值的C结构化分解[6]；这在下文的定义3.7中进行了总结。

定义 3.7。（结构分解范畴）
分解范畴。固定一个范畴 K，K 值结构分解（任意形状）及其间的态射（如定义 3.5 所述）构成一个范畴 $\mathfrak{D}_{\mathrm{m}}(\mathsf{K})$，称为 K 值结构分解范畴。此外，该构造具有函子性：存在一个函子 $\mathfrak{D}_{\mathrm{m}} : \mathsf{Cat} \to \mathsf{Cat}$，它将任意范畴 K 映射到范畴 $\mathfrak{D}_{\mathrm{m}}(\mathsf{K})$，并将每个函子$\Phi : \mathsf{K} \to \mathsf{K}'$ 映射到函子 $\mathfrak{D}_{\mathrm{m}}(\Phi) : \mathfrak{D}_{\mathrm{m}}(\mathsf{K}) \to \mathfrak{D}_{\mathrm{m}}(\mathsf{K}')$，其在对象上的定义为

$$\mathfrak{D}_{\mathrm{m}}(\Phi) \colon \left(\int G \xrightarrow{d} \mathsf{K}\right) \mapsto \left(\mathfrak{D}_{\mathrm{m}}(\Phi)d \colon \int G \xrightarrow{d} \mathsf{K} \xrightarrow{\Phi} \mathsf{K}'\right)$$

并且在态射上的形式为



[6]为了与前篇 [34] 保持记号一致，并提醒读者我们在全文中只处理单态结构化分解，我们将在范畴 $\mathfrak{D}_{\mathrm{m}}$C 的记号中保留下标$m$ 。

$\mathfrak{D}_{\mathrm{m}}$ 是一个函子，这意味着每个编码某种计算问题的函子 $F\colon \mathsf{C} \to \mathsf{Sol}$ 都可以提升为一个函子

$$\mathfrak{D}_{\mathrm{m}F}\colon \mathfrak{D}_{\mathrm{m}}\,\mathsf{C} \to \mathfrak{D}_{\mathrm{m}}\,\mathsf{Sol}$$

将计算问题的输入的结构化分解（即取值于C中的分解）通过在问题的解空间（即取值于Sol中的分解）中计算F 在对象 $\mathfrak{D}_{\mathrm{m}}$C的袋和粘合上的值来映射到后者。我们将在第4 节中利用这一事实，以便轻松编码分解的袋和粘合上的局部计算，并实现我们的结果（见第4.3节）。

作为格罗滕迪克拓扑的结构化分解我们终于可以证明，结构化分解会在粘合范畴上产生格罗滕迪克拓扑。我们将以与处理任意子单态图表时大致相同的方式进行。事实上，注意到 [34, 推论 5。11] 对于任意小型粘着余完备范畴C，可以（再次通过如定理3.4中的逐点拉回）定义函子Dcmp 的一个子函子subMon （在定理3.4中定义）如下（其中我们再次将 Λ_d 记作余极限余锥中态射的集合，该余锥基于d）。

$$\begin{aligned}&\mathsf{Dcmp}\colon \mathsf{C}^{op} \to \mathbf{Set} \qquad (2)\\ &\mathsf{Dcmp}\colon c \mapsto \{\Lambda_d \mid \operatorname{colim} d = c \text{ and } d \in \mathfrak{D}_{\mathrm{m}}\,\mathsf{C}\} \bigcup \{\{f\} \mid f\colon a \xrightarrow{\cong} c \text{ an iso }\}.\end{aligned}$$

推论 3.8。 这两对 $(\mathsf{C},\mathsf{Dcmp})$和 $\left(\mathsf{C}_{mono}、\ \mathsf{Dcmp}\,|_{\mathsf{C}_{mono}}\right)$ 都是位点，其中 C是任意小的粘合余完备范畴，而Dcmp 是方程（2）中的函子。

证明:
每个取值于 C 的结构化分解都会在 C 中诱导出一个图表。反过来，由于 C 是粘合范畴，单态子范畴的余极限余锥由单态箭头构成。因此，对余极限箭头取成对拉回，即可恢复一个具有相同余极限的结构化分解（因为 C 是粘合范畴），且所有粘合都是单态的。□

一般而言，我们可以为某个特定范畴指定许多不同的格罗滕迪克拓扑，从而得到不同的位点。构造特定位点时，我们会考虑相对于它定义的层。如果能够利用位点之间的某种关系来诱导相关层之间的关系，并通过从另一个（恰当相关的）格罗滕迪克拓扑上的层提升计算，来减少我们计算某个格罗滕迪克拓扑上的层所需的工作，那就再好不过了。事实证明，范畴 C 上的格罗滕迪克拓扑构成一个偏序集（按包含关系排序），而已有结果表明，我们可以利用这一点，将一种拓扑上的层推送到其他拓扑上的层。

我们可以先如下定义任意覆盖之间的映射。

定义 3.9. 从 $\mathcal{V} = \{V_j \to V\}_{j\in J}$到 $\mathcal{U} = \{U_i \to U\}_{i\in I}$的**覆盖态射** 是一个$V \to U$箭头，并附带一对由 (σ, f) 组成的

- 一个定义在指标集上的函数 $\sigma\colon J \to I$，且

- 一组态射 $f = \{f_j \colon V_j \to U_{\sigma(j)}\}_{j \in J}$，其中

$$\begin{array}{ccc} V_j & \longrightarrow & U_{\sigma(j)} \\ \downarrow & & \downarrow \\ V & \longrightarrow & U \end{array}$$

  对所有 $j \in J$ 交换。

借助这种态射概念，我们可以定义 C 上覆盖的范畴。不过，我们主要关注覆盖态射中的一种特殊情形，即 $V = U$ 且 $V \to U$ 只是恒等映射。在这种情况下，我们称 $\mathcal{V}$ 是**U**的**加细**。

定义 3.10。设 C 是一个范畴，且 $\mathcal{U} = \{U_i \to U\}_{i \in I}$ 是一个箭头族。那么，**加细** $\mathcal{V} = \{V_j \to U\}_{j \in J}$ 是一个箭头族，使得对于每个指标 $j \in J$，都存在某个 $i \in I$，使得 $V_j \to U$ 经由 $U_i \to U$ 分解。

用筛的术语，我们可以将上述内容简化为：给定 $\mathcal{U} = \{U_i \to U\}$ 和 $\mathcal{V} = \{V_j \to U\}$，当且仅当由 $\mathcal{V}$ 生成的筛包含于由 $\mathcal{U}$ 生成的筛（的一个子筛）中时，$\mathcal{V}$ 才是 $\mathcal{U}$ 的加细。

此外，请注意，任何覆盖都是其自身的加细，而加细的加细仍是加细。因此，加细为对象 U 的覆盖赋予了一种序关系。我们可以利用这一点来定义格罗滕迪克拓扑之间的以下关系。

定义 3.11。设 C 是一个范畴，J, J' 是 C 上的两个格罗滕迪克拓扑。若 J 从属于 J'，记作 $J \leqslant J'$，则意味着 J 中的每个覆盖都有一个加细，该加细是 J' 中的覆盖。

如果 $J \leqslant J'$，且 $J' \leqslant J$，那么 J 和 J' 是**等价**的。

这实际上不仅意味着 J 中的任意覆盖都是 J' 中的覆盖——这使我们称拓扑 J' 比 J 更细——还意味着对于 J 中的任意覆盖，在 J' 的覆盖中都存在一个加细。

用筛的术语来说，$J \leqslant J'$ 意味着：对于任意对象 U 以及任意被 J 视为覆盖筛的筛 S，都存在一个筛 S'，它被 J' 视为覆盖筛，并且 S 的所有态射也都属于 S'。特别地，当且仅当两个拓扑具有相同的筛时，它们才等价。

现在我们来说明这些定义的要点。根据层的主要定义——即，函子 F</style>关于拓扑 J 是一个层，当且仅当对于属于 J 的任意筛 S，诱导映射[7]

$$F(U) \simeq \mathrm{Hom}(\mathbf{y}_U, F) \to \mathrm{Hom}(S, F)$$

是一个双射——结合前面以筛的术语表述的定义，我们得到如下结果。

[7] 这里我们用 $\mathbf{y}_U$ 表示在 U 处的 Yoneda 嵌入。

命题 3.12。 设 J,J' 是范畴 C 上的两个格罗滕迪克拓扑。若 J 从属于 J'，即 $J \leqslant J'$，那么每个满足 J' 的层条件的预层也满足 J的层条件。

换句话说：如果 J 从属于J'，那么我们知道，任何相对于J' 的层都会自动成为相对于J的层。

证明：

我们可以通过运用位点态射的概念来证明这一结果（参见 Mac Lane 和 Mo- erdijk [30]），即 $f : (C, J) \to (D, K)$ 是一种特定的“保持覆盖”的态射，其中令 $D = C$ 且 $K = J'$，并使用恒等函子 $C \to C$，将任意对象映到其自身，也将任意 J-覆盖映到其自身。与 f 作前复合会给出预层范畴之间的一个函子（方向相反）；然后我们使用推前（或直接像）函子 $f_* : Sh(C,J') \to Sh(C,J)$，它是将前复合限制到层上的函子，从而把关于 J' 的一个层推送为 J 上的层 f_*F。详情可见相关文献。□

特别地，两个等价拓扑具有相同的层。

这里最重要的结论当然是：如果我们能证明某个预层实际上是相对于拓扑 J' 的层，并且如果 J 是另一种拓扑，而我们能够证明它从属于 J'，那么我们就可以直接得到一个相对于 J 的层（事实上，也相对于任何其他从属拓扑）。此外，特别是，如果存在一个最细的覆盖，那么在该覆盖上验证层公理，就能保证它对所有覆盖都成立。

现在，让我们将这些已确立的结果与我们在定理 3.4 和推论 3.8中定义的格罗滕迪克拓扑联系起来，即 subMon 和 Dcmp，其中后者是前者的一个子函子。我们可以着重说明 subMon 与另一种值得关注的格罗滕迪克拓扑之间的关系。

命题 3.13。 设 C 是任意粘合余完备范畴。C 上由 subMon 给出的拓扑从属于子对象拓扑。

证明：

子对象拓扑仅将对象的以下内容作为覆盖：c 其并为 c 的单态射（等价类）。我们需要证明，单图拓扑中的任意覆盖都可以由子对象拓扑中的一个覆盖细化。根据推论 subMon 的定义 3.8，这一点立即可得。□

结合命题3.12，可得以下结果；在证明给定预层确实是关于分解拓扑的层时，这一结果尤其方便。

命题 3.14。 任取粘合余完备范畴 C 和预层 $\mathcal{F}$: $C^{op} \to$Set。若 $\mathcal{F}$是关于子对象拓扑的 $(C)_{mono}$ 上的层，那么它也自动是位点 $\left(C_{mono}, \text{Dcmp}|_{C_{mono}}\right)$上的层。

4. 判定层

从算法的角度来看，定理3.4 和推论3.8 的核心在于讨论能够通过组合算法解决的计算问题。为此，

假设给定一个层$\mathcal{F}\colon \mathsf{C}^{op} \to$ 有限集范畴，它相对于位点(C,Dcmp)定义。我们将这个层视为一个计算问题：它规定了哪些解空间与哪些输入相关联。我们已经在第1节见过这种构造的一个具体例子，即着色函子

$$\mathsf{SimpFinGr}(-, K_n)\colon \mathsf{SimpFinGr}^{op} \to \mathsf{FinSet}$$

它将每个图映射到该图的所有n着色所构成的集合。

本文关注判定问题，具体来说是层判定问题（定义见第1节），为便于查阅，下面重述该问题。

C – SHEAFDECISION
输入： 位点 (C,Dcmp) 上的一个层 $\mathcal{F}\colon \mathsf{C} \to$ 有限集范畴^{op}，其中 C 是一个小型粘着余完备范畴，一个对象 $c \in \mathsf{C}$，以及一个取值于$\mathcal{C}$的、针对c的结构化分解 d。
任务： 令**2** 为二对象范畴 $\bot \to \top$，并令 dec: 有限集范畴 $\to \mathbf{2}$ 为如下函子：将集合映射到 $\bot$（若集合为空），否则映射到 $\top$；计算 $\mathrm{dec}^{op}\mathcal{F}c$。

本节将证明，若以给定结构化分解的宽度和分解形状的反馈顶点数[8] 作为双重参数，则层判定问题属于固定参数可处理类。请注意，当分解呈树形时，我们便得到基于树宽抽象类比概念的参数化（注意，这一概念可在任意粘合范畴中实例化，而不仅限于图所在的范畴）。

为此，请注意，若 C 有余极限，那么由于该位点上的层会将分解的余极限映射为集合的极限 [29], ，以下图表总是交换的（参考 Bumpus、Kocsis 和 Master [34] ）。

$$\begin{array}{ccccc} \mathsf{C} & \xrightarrow{\quad\mathcal{F}\quad} & \mathsf{FinSet}^{op} & \xrightarrow{\ \mathrm{dec}^{op}\ } & \mathbf{2}^{op} \\ {\scriptstyle\mathrm{colim}}\uparrow & \textit{comm.} & \uparrow{\scriptstyle\mathrm{colim}} & & \\ \mathfrak{D}_{\mathrm{m}}\,\mathsf{C} & \xrightarrow{\ \mathfrak{D}_{\mathrm{m}}\mathcal{F}\ } & \mathfrak{D}_{\mathrm{m}}\,\mathsf{FinSet}^{op} & & \end{array} \tag{3}$$

为明确起见，我们来详细说明这意味着什么。蓝色路径表示先将分解的组成部分粘合在一起（即忽略组合结构），再在整个输出上求解问题。另一方面，红色路径表示一种组合式解法：先在分解的组成部分上计算$\mathcal{F}$，然后将这些解合并[9]，从而找到整个分解上的一个解。因此，由于此图表对任意层$\mathcal{F}$都交换，我们刚刚观察到，SHEAFDECISION 存在一种组合算法。

遗憾的是，我们刚才描述的方法仍然非常低效，因为对于任意输入 c，无论我们在图表中选择哪条路径，最终总要计算 $\mathcal{F}(c)$ 的全部内容，而它通常都非常庞大（例如，我们上面描述的着色层的像的基数

[8]一个**反馈顶点集**在图G中是顶点子集$S \subseteq V(G)$的G，从G中移除该子集后可得到一个无环图。图**的反馈顶点数**是图G中反馈顶点集的最小大小，即G。[9]请注意，余极限函子colim： $\mathfrak{D}_{\mathrm{m}}$有限集范畴$^{op} \to$ 有限集范畴^{op} ——在图表3 中以红色显示——是一个集合的极限；我们请读者在全文中始终牢记这一点。

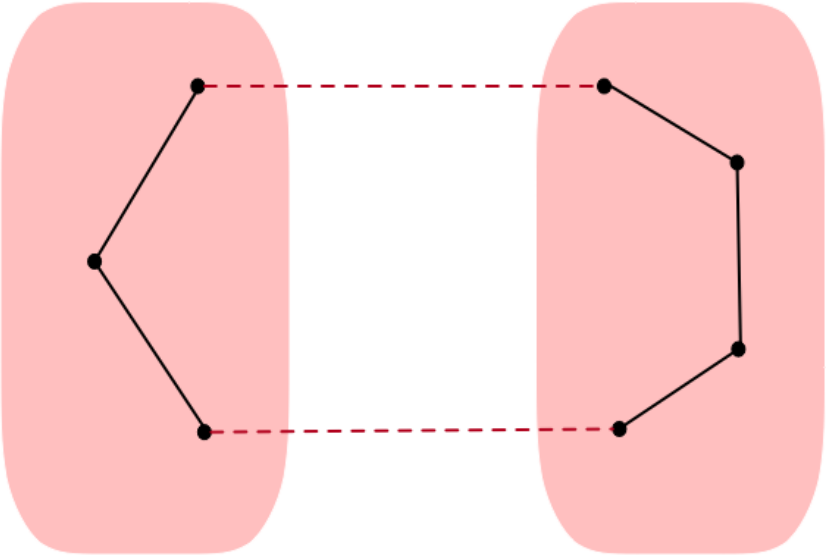


图2。一个结构化分解 $P_3 \leftarrow K_2 \rightarrow P_4$，将一个5-环分解为两条路径，分别有2条边和3条边

会随着输入图的规模呈指数增长）。人们可能希望通过将 dec 提升为一个从 FinSet^{op} 值结构化分解到 $\mathbf{2}^{op}$ 值结构化分解的函子来克服这一困难，如下图表所示。

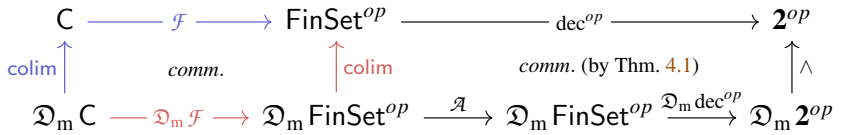
$$\begin{array}{ccccc} \mathsf{C} & \xrightarrow{\quad\mathcal{F}\quad} & \mathsf{FinSet}^{op} & \xrightarrow{\quad \mathrm{dec}^{op} \quad} & \mathbf{2}^{op} \\ {\scriptstyle \mathrm{colim}}\uparrow & \textit{comm.} & \uparrow{\scriptstyle \mathrm{colim}} & & \uparrow{\scriptstyle \mathrm{colim}=\wedge} \\ \mathfrak{D}_{\mathrm{m}}\mathsf{C} & \xrightarrow{\quad\mathfrak{D}_{\mathrm{m}}\mathcal{F}\quad} & \mathfrak{D}_{\mathrm{m}}\mathsf{FinSet}^{op} & \xrightarrow{\quad \mathfrak{D}_{\mathrm{m}}\mathrm{dec}^{op} \quad} & \mathfrak{D}_{\mathrm{m}}\mathbf{2}^{op} \end{array}$$

然而，这样做也无济于事：上述图表中的右侧方块通常不交换。要了解原因，请考虑2-着色层$\mathsf{SimpFinGr}(-,\underline{K_2})$，并令$d$为结构化分解$P_3 \leftarrow K_2 \rightarrow P_4$，它将一个5-环分解为两条路径，分别有2条边和3条边（见图2）。在 $\mathfrak{D}_{\mathrm{m}}\mathrm{dec}^{op} \circ \mathfrak{D}_{\mathrm{m}}\mathcal{F}$下，$d$的像为 $\top \leftarrow \top \rightarrow \top$（因为图$P_3$，$K_2$和$P_4$都可以2-着色），而该图表在 $\mathbf{2}^{op}$ 中的余极限（即$\mathbf{2}$中的极限，也就是合取）为 $\top$。然而，5-环无法进行2-着色。

虽然这个反例可能令人气馁，但我们将看到，刚才概述的方法与正确思路非常接近。事实上，在本节余下的内容中，我们将证明（定理4.1），存在一个自函子

$$\mathcal{A}\colon \mathfrak{D}_{\mathrm{m}}\mathsf{FinSet}^{op} \to \mathfrak{D}_{\mathrm{m}}\mathsf{FinSet}^{op}$$

使下图交换。



(4)

此外，我们将证明这个函子 $\mathcal{A}$ 可以高效地计算（这就是定理1.1，我们在引言中提到过）。

当 C 不存在余极限时。 在继续定义$\mathcal{A}$的对象映射和态射映射之前（正如我们将在定理 4.1 中证明的那样，它们组合起来构成一个函子），我们先简要说明，上文讨论的全部内容甚至也适用于定义在位点 $(\mathsf{C}_{mono}, \mathrm{Dcmp}\mid_{\mathsf{C}_{mono}})$上的层。回顾第2 节和第3 节，其中有许多情形下，我们希望讨论定义在范畴 C_{mono} 上的计算问题；该范畴与 C 具有相同的对象，但只包含 C 中作为单态射的那些态射。在这种情况下，由于 C_{mono} 一般而言不存在余极限，我们显然无法推导出“组合算法之交换方块”（图表3）的交换性，因为此时不再有余极限函子（即 colim，它在图表3中以蓝色标出）。不过请注意，在两种情况下——也就是说，取任意 $\mathsf{K} \in \{\mathsf{C}, \mathsf{C}_{mono}\}$——如果$\mathcal{F}$是关于位点 $(\mathsf{K}, \mathrm{Dcmp})$ 的层，那么对于任意覆盖，我们总有

$$(\kappa \in \mathsf{K}, d \in \mathrm{Dcmp}\,\kappa)$$

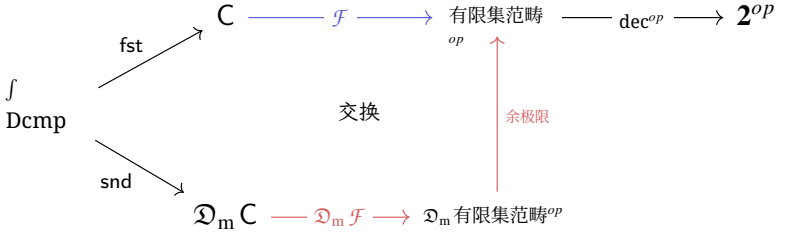
我们总是有

$$(\mathrm{dec}^{op} \circ \mathcal{F})(\kappa) = (\mathrm{dec}^{op} \circ \mathsf{const} \circ \mathfrak{D}_{\mathrm{m}}\mathcal{F})(d). \tag{5}$$

用图表来表述这一点时，我们可以使用格罗滕迪克构造[10]。为方便起见，我们明确写出范畴 $\int \mathrm{Dcmp}$ 的定义；它定义为具有

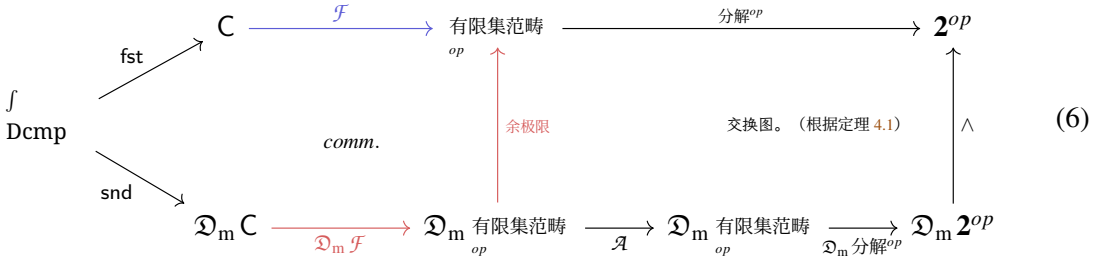
- **对象** 由以下配对给出 (c, d)，其中 $c \in \mathsf{C}$ 且 $d \in \mathrm{Dcmp}(c)$
- **态射** 从 (c, d) 到 (c', d')，由 C 中形如 $f0 : c \to c'$ 的态射给出，满足 $\mathrm{Dcmp}(f)(d') = d$。

注意，这　　这使我们能够将方程（5）改写为以下图表的交换性　　.



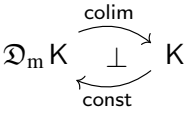
[10]~~回顾一下，格罗滕迪克构造从任意函子 $F\colon \mathrm{A} \to \mathrm{B}$ 生成一个记作 $\int F$ 的范畴~~。关于该构造及其性质的详细信息，请参阅 Riehl 的教科书 [40]

其中，第一投影 和 第二投影 是显然的投影函子，其对象映射分别为 第一投影 $:(c,d)\mapsto c$ 以及 第二投影 $:(c,d)\mapsto d$。所有这些机制的意义在于，当我们的输入范畴 C 没有余极限时，它能让我们用以下方式重述算法目标（即图表（4），我们将在定理4.1中证明其交换性）。请注意，以下图表是图表1 的形式化版本，我们曾在第 1 节用它勾勒我们的“路线图”。）



(6)

剩下要做的就是证明图表 $\mathcal{A}$ 中的函子 6。正如我们将看到的，我们将利用如下定义的单子 $\mathcal{T}$ 来定义这个函子。回想一下，对于任何具有拉回和余极限的范畴 K，都存在以下伴随 [34]（这是图表范畴中相应伴随的一个特例）



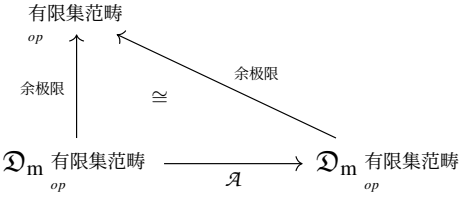
其中，函子 const 将 K 中的对象映射为平凡分解（只有一个袋），而函子 colim 将分解映射为其余极限。现在，取 K =有限集范畴^{op}，我们将用$\mathcal{T}$ 表示这个伴随定义的单子

$$\begin{aligned}\mathcal{T}&\colon \mathfrak{D}_{\mathrm{m}}\mathsf{FinSet}^{op}\to\mathfrak{D}_{\mathrm{m}}\mathsf{FinSet}^{op}\\ \mathcal{T}&\colon d\mapsto(\mathsf{const}\circ\mathsf{colim})(d)\end{aligned}\tag{7}$$

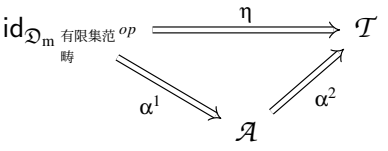
在下述定理的证明中，我们将利用这个单子，最终证明所需函子$\mathcal{A}$的存在。

定理 4.1. 存在一个函子 $\mathcal{A}\colon$ $\mathfrak{D}_{\mathrm{m}}$有限集范畴$^{op}\to\mathfrak{D}_{\mathrm{m}}$有限集范畴^{op} 使得：

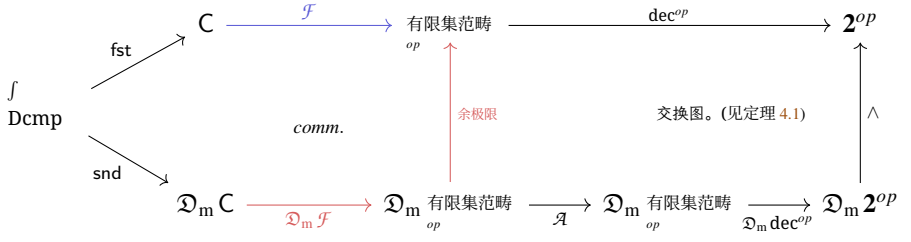
(A1) 存在一个自然同构



(A2) 存在自然变换 α^1 和 α^2，它们将 η 单子 $\mathcal{T}$ 的单位分解——参见函子（7），如下图表所示



(A3) 以及下列交换图（即图表 (6)，重述如下）



应将定理$\mathcal{A}$中的函子4.1 视为一种预处理程序，用以筛除那些无法扩展为全局解的局部解。此外，请注意，$\mathcal{A}$在某种意义上是典范的，因为它源自单子$\mathcal{T}$。事实上，正如我们将看到的，尽管证明的大部分内容都聚焦于$\mathcal{A}$的推导，但这一推导中的每一步都不是临时拼凑的：它们是由单子$\mathcal{T}$的性质完全决定的“显而易见的步骤”。我们认为，这是范畴论方法的一大优势。

4.1. 定理的证明 4.1

证明的结构如下：我们首先利用单子 $\mathcal{T}$来定义函子$\mathcal{A}$：$\mathfrak{D}_m$有限集范畴$^{op} \to \mathfrak{D}_m$有限集范畴^{op}；然后我们将指出，在 $\mathcal{A}$ 的推导过程中积累的各种事实，很容易推出定理陈述中列出的要点**(A1)**、**(A2)** 和 **(A3)** 。

4.1.1. 收集直觉

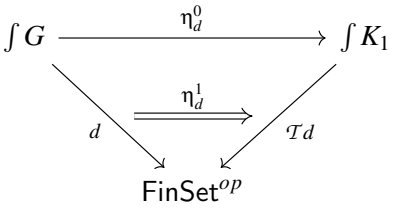
请注意，如果暂时不考虑计算效率，那么很容易为所需的函子找到一个候选者 $\mathcal{A}$。为了解释这一点，考虑将一个以有限集范畴^{op}为值的结构化分解 d 通过单子 $\mathcal{T}$时会发生什么。我们将得到 $\mathcal{T}d$，它由一个形状为 K_1（即单顶点完全图）的结构化分解组成，其袋就是我们要计算的解空间（即 d 在有限集范畴^{op}中的余极限）。因此，如果将$\mathcal{A}$替换为 $\mathcal{T}$，图表（4）显然交换。

尽管$\mathcal{T}$无法高效计算（因为正如我们前面提到的，colim$d$ 的集合可能非常大），但我们将看到，由伴随 const $\vdash$colim 给出的单子$\mathcal{T}$的单位 η，是指定所需函子$\mathcal{A}$（本节稍后将证明其可高效计算）所需的关键要素（在同构意义下）。因此，为了定义$\mathcal{A}$，我们将首先详细说明 η的定义。

注意， η 给出了一个结构化分解之间的态射

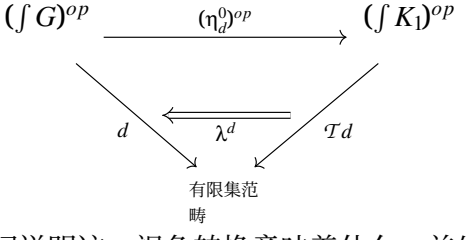
$$\eta_d : d \to \mathcal{T}d$$

按定义，它由一对 (η^0_d, η^1_d) 构成，其中 $\eta^0_d\colon \int G \to \int K_1$ 是一个函子，而 $\eta^1_d\colon d\,\mathcal{T}(d)\eta^0_d$ 是一个自然变换，如下图表所示。



(8)

现在请注意，上面的图表将d 和 $\mathcal{T}d$ 视为 有限集范畴^{op}值结构化分解。为方便阅读，我们将把图表（9）中的 op“移过去”，从而将 η_d 视为有限集范畴值结构化**共分解**（即反变结构分解）之间的一个态射；换言之，我们将把图表（9）改写如下。



(9)

为清楚起见，我们先花些时间说明这一视角转换意味着什么。首先请注意，有限集值余分解会为G 的每条边关联一个集合的余跨图，其锥腿都是满态射（因为它们在有限集范畴^{op}中是单态射）。自然变换 λ^d 不过是变换 η^1_d，只是将其各分量视为有限集范畴^{op}中的态射，而非有限集范畴中的态射（为避免记号混淆，我们选择将其简单记作 λ^d ）。 λ^d 的各分量是函数（投影），从$\mathcal{T}d$ 的唯一一个袋映射到d 的各个袋，如下所示。

$$\lambda^d_x\colon \mathcal{T}d \to dx. \tag{10}$$

特别地，这些由位于d 上方、顶点为$\mathcal{T}d$的极限锥[11] 给出。

4.1.2. 定义$\mathcal{A}$的对象映射。

有鉴于这些观察，我们现在可以定义

$$\mathcal{A}_0\colon \left(\left(\int G\right)^{op} \xrightarrow{d} \mathsf{FinSet}\right) \mapsto \left(\left(\int G\right)^{op} \xrightarrow{\mathcal{A}_0 d} \mathsf{FinSet}\right) \tag{11}$$

[11] ~~再次提醒读者，对有限集范畴值分解$d\colon \int G \to$有限集范畴^{op} 在有限集范畴^{op} 中取**余极限**，所得结果与先将d 视为有限集值**余**分解$d\colon (\int G)^{op} \to$ 有限集范畴，再在有限集范畴中计算其**极限**相同。~~

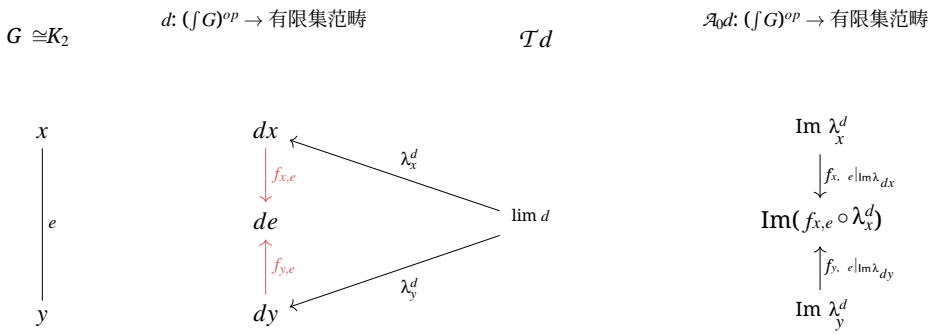


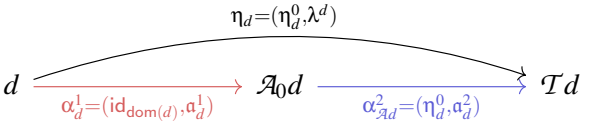
图 3. 结构化分解d 上形状为 $\vec{K}_2$ （有向边）的对象映射$\mathcal{A}$ 之定义（公式（11）和（12））的可视化。由 $\mathcal{T}$ 给出的单子的单位产生一个态射 $(\eta^0_d,\ \eta^1_d) : d \to \mathcal{T}d$。 η^1_d 的分量（即 $\eta^1_d(x)$ 和 $\eta^1_d(y)$）是有限集范畴中的态射^{op}；将它们视为有限集范畴中的态射时，记作 λ^d_x和 λ^d_y。这些态射由d上的极限锥给出（将d视为有限集范畴中的图表）。

我们力图建立的函子 $\mathcal{A}$ 的对象映射。它将 d 映射到如下分解：将 d 的所有袋和粘合限制到顶点为$\mathcal{T}d$ 的极限锥的各锥腿的像上（我们建议读者参阅图 3 以获得直观说明）。形式化地说，映射 $\mathcal{A}_0$ 将一个结构化余分解 $d\colon (\int G)^{op} \to$ 有限集范畴映射为结构化余分解$\mathcal{A}_0 d\colon (\int G)^{op} \to$ 有限集范畴；后者与 d 具有相同的形状，并定义如下：

$$\begin{aligned} &\mathcal{A}_0 d\colon \left(x \in \int G\right) \mapsto \mathsf{Im}\,\lambda^d_x \\ &\mathcal{A}_0 d\colon \left(e \xleftarrow{f_{x,e}} x \in \mathsf{Mor}\left(\int G\right)^{op}\right) \mapsto \left(\mathsf{Im}\,\lambda^d_e \xleftarrow{f_{x,e}|_{\mathsf{Im}\lambda^d_x}} \mathsf{Im}\,\lambda^d_x\right). \end{aligned} \tag{12}$$

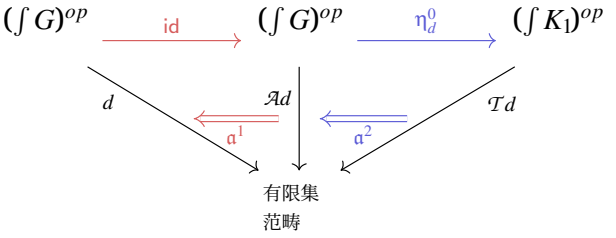
4.1.3. $\mathcal{A}$ 的态射映射定义的预备知识。

现在，为了定义态射映射$\mathcal{A}_1$, , 请注意，根据$\mathcal{A}_0$ 通过单位 η $\mathcal{T}$的定义，我们有 η 分解为



(13)

其中，对于任意 $x \in \int G$，下方所示的映射 $\mathfrak{a}^1_d$和 $\mathfrak{a}^2\mathcal{A}d$

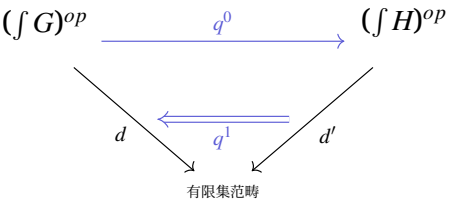


分别由限制和余限制定义[12] 的 λ^d。具体来说，这些映射被定义为有限集范畴中的以下态射

$$\begin{aligned} \mathfrak{a}^1_d(x) := \quad & \mathcal{A}dx \xhookrightarrow{\mathsf{id}_{dx}|_{\mathsf{Im}\lambda^d_x}} dx \\ \mathfrak{a}^2_{\mathcal{A}d}(x) := \quad & \mathcal{T}dK_1 \xrightarrow{\lambda^d_x|^{\mathsf{Im}\lambda^d_x}} \mathcal{A}dx \end{aligned} \tag{14}$$

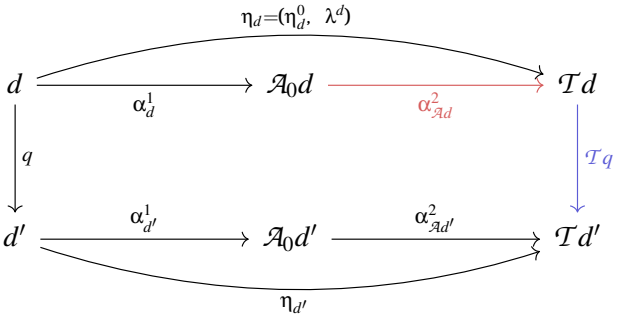
4.1.4. 定义$\mathcal{A}$的态射映射。

给定如下图表中的任意有限集范畴值结构化余分解态射$q = (q^0, q^1)$



(15)

根据图表（13）、单子 $\mathcal{T}$的函子性及其单位的自然性，我们得到以下图表交换。



(16)

请回想，上述图表中的每个态射本身都是图表范畴中的态射，该范畴由一个函子和一个“向后”指向的自然变换组成（例如图表（15）所示）。例如，请回想，态射 $\eta_d\colon\ d \to \mathcal{T}d$（绘于图表（9）中）对应于 d上的极限锥。

我们要证明存在一个态射 $\mathcal{A}_0 d \to \mathcal{A}_0 d'$，它与图表（16）交换。这将由图表（16）的交换性推出。不过，仍有必要将其展开如下。

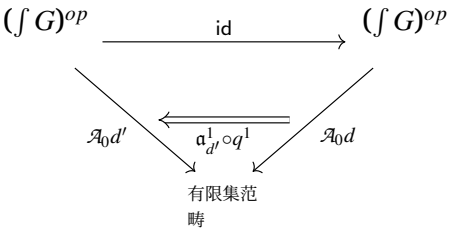
考虑图表（16）。态射 $\mathcal{T}q$ 表示一个从 d' 的极限到d 的极限的函数。态射 η_d 和 $\eta_{d'}$ 分别表示位于d 和d' 上方的极限锥。

[12]对偶于限制所使用的记号，对于任意函数 $f\colon\ A \to B$，我们将 f到其像的**余限制** 记为 $f|^{\mathsf{Im}}f$。

态射 q （在图表（15）中定义）对应于为每个对象 $x \in \mathrm{dom}d$ 指定一个态射

$$q_x^1 \colon (q^0 x \in \mathrm{dom}\, d') \to (x \in \mathrm{dom}\, d).$$

图表（16）的交换性意味着如下陈述：对于每个$x \in \mathrm{dom}d$ 以及q_x^1 的每个分量$_1$, , q_1, 的值域 q_x^1 完全包含在极限锥 λ^d 在x处的锥腿 λ_x^d 的像$\mathcal{T}d \to dx$ 中。但这意味着，将q_x^1 限制到 $\lambda^{d'}x$的像后，我们就得到了下方所示的所需态射（请回忆 $\alpha_{d'}^1 := (\mathrm{id}_{(\int G)^{op}}, \mathfrak{a}_{d'}^1)$ 。



总之， thi 该推导使我们能够定义所需的态射 $\mathcal{A}q \colon \mathcal{A}_0 d \to \mathcal{A}_0 d'$，其定义为

$$\mathcal{A}_1 \colon \big((q_0, q_1) \colon d \to d'\big) \mapsto \big((q_0, \mathfrak{a}_{d'}^1 \circ q^1)\big). \tag{17}$$

4.1.5. 完成证明

A 的函子性是显然的：它显然保持恒等态射；此外，由于 T 是一个函子，通过检查图表（16）可以看出，A 保持复合。另一方面，性质（A1）可由 A_0 的定义以及有限集范畴中的极限的性质推出（即，如果 $\mathrm{X}_1 \xleftarrow{\pi_1} \mathrm{X}_1 \times_S \mathrm{X}_2 \xrightarrow{\pi_2} \mathrm{X}_2$ 是跨度 $\mathrm{X}_1 \xrightarrow{\mathrm{f}^1} \mathrm{S} \xleftarrow{\mathrm{f}^2} \mathrm{X}_2$，的拉回，那么 $\mathrm{X}_1 \times_S \mathrm{X}_2$ 也将同构于跨度 $\mathrm{Im}\pi_1 \xrightarrow{\mathrm{f}^1|_{\mathrm{Im}\pi_1}} \mathrm{S} \xleftarrow{\mathrm{f}^2|_{\mathrm{Im}\pi_1}} \mathrm{Im}\pi_2$ 的拉回对象）。由于 α^1 和 α^2 的自然性是显然的，性质（A2）就等价于图表（13）的交换性。最后，为证明性质（A3），注意到根据对象映射 A_0 的定义，并且由于唯一存在指向空集的态射的集合就是空集，我们有 $\wedge \circ \mathfrak{D}_{\mathrm{m}}$ $\mathrm{dec}^{op} \circ \mathrm{A} = \mathrm{dec}^{op} \circ$余极限。

4.2. 计算$\mathcal{A}$的算法

回想一下，应将定理 $\mathcal{A}$ 4.1 中的函子视为一个预处理程序，它会过滤掉那些无法扩展为全局解的局部解。这一直觉引出了以下局部筛选算法：沿粘合对袋进行拉回，从而在局部过滤袋对，然后仅保留位于拉回的投影映射像中的局部解。

递归应用算法 1 使我们能够得到以下用于在树形结构化分解上进行层判定的算法。

引理 4.2. 存在一种算法（即算法 2），能够在时间 $O(\kappa^2)|EG|$ 内，正确计算任意输入的取值于有限集合范畴的结构余分解 $\mathcal{A}d \; d \colon (\int G)^{op} \to \mathrm{FinSet}$，其中 $\kappa = \max_{x \in VG} |dx|$，前提是 G 是一棵有限的、非自反的有向树。

算法 1过滤

输入： a有限集范畴- 取值于有限集合范畴的结构化余分解 $d\colon (\int G)^{op} \to \mathsf{FinSet}$，以及 G 中的一条边 $e = xy$.

– 计算每条边 $e = xy$ 在 G中关联的余跨图 $dx_f \xrightarrow{x} de_{x,y_f} \xleftarrow{y} dy$ 的拉回 $dx \xleftarrow{\pi_x} dx \times_{de} dy \xrightarrow{\pi_y} dy$ – 将 d_e 替换为如下分解：把 e 在 d 中关联的余跨图替换为该余跨图

$$\mathsf{Im}\,\pi_x \xrightarrow{f_x|_{\mathsf{Im}\,\pi_x}} de_{x,y} \xleftarrow{f_y|_{\mathsf{Im}\,\pi_y}} \mathsf{Im}\,\pi_y,$$

返回 d_e。

证明：
固定$e_1,\dots,e_m$为G的边任取一个枚举，我们将证明，在输入2时调用以下递归过程（算法 ）满足引理的要求$(d,(e_1,\dots,e_m))$。注意，在算法 1 中，我们总有一个单射 $\mathsf{Im}\,\pi_x \hookrightarrow dx$。根据这一事实，且由于算法 2

算法 2递归过滤

输入： 一个取值于有限集范畴^{op}的结构化分解 $d\colon (\int G)^{op} \to$ 有限集范畴，以及G的一列边 ℓ。

如果 ℓ为空**则**
  返回d
else
  – 将 ℓ拆分为首边 e 和其余部分 ℓ'– 令d_e 为算法 1 对输
  入 (d,e)的输出
– 递归调用算法 2 ，输入为 (d_e,ℓ')。

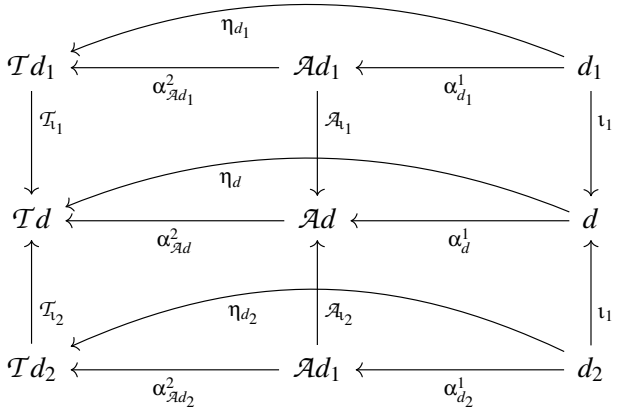
结束 if

相当于在有限集范畴中计算 $|\mathsf{EG}|$ 拉回，因此运行时间上界显而易见。现在假设输入分解是树形的（即假设 G 是一棵树）；我们将通过归纳法证明算法 1 的输出与 Ad 同构。如果 $\mathsf{G} \cong \mathsf{K}_1$ （即单顶点完全图），那么算法立即终止（因为 G 没有边）并返回 d。由于当 $\mathsf{G} \cong \mathsf{K}_1$ 时 $\mathsf{Ad} \cong \mathsf{d}$，这就确立了归纳法的基例。现在假设 $|\mathsf{EG}| > 0$，并令 $\mathsf{e} = \mathsf{x}_1\mathsf{x}_2$ 为算法 2 迭代处理的树 G 上的最后一条边。移除 e 会将 G 分成两棵子树 $\mathsf{T}_1 \hookrightarrow \mathsf{G}$ 和 $\mathsf{T}_2 \hookrightarrow \mathsf{G}$，分别包含节点 x_1 和 x_2。相应地，这两棵树诱导出 d 的两个子分解 $\iota_i\colon \mathsf{d}_i \hookrightarrow \mathsf{d}$；它们由以下复合函子给出：

$$d_1\colon \left(\int T_1\right)^{op} \hookrightarrow \left(\int G\right)^{op} \to \mathsf{FinSet} \quad \text{and} \quad d_2\colon \left(\int T_2\right)^{op} \hookrightarrow \left(\int G\right)^{op} \to \mathsf{FinSet}.$$

由归纳假设可知，对于每个 $i \in \{1,2\}$，对 1 应用算法 d_i 会得到一个对象 δ_i，它与 $\mathcal{A}d_i$ 同构：$(\int T_i)^{op} \to$ 有限集范畴。因此，根据定理 **4.1** 的要点 （A2），以下各项的跨度

结构化分解 $d_1 \overset{\iota_1}{\hookrightarrow} d \overset{\iota_2}{\hookleftarrow} d_2$ 给出如下交换图。

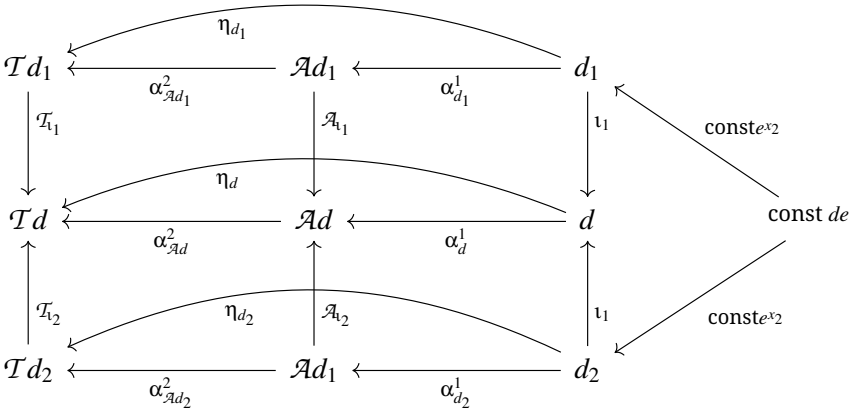


(18)

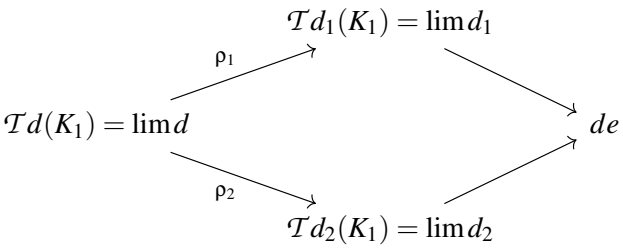
现在考虑余跨度 $dx_1 \overset{x_1}{\longrightarrow} de \overset{x_1}{\longleftarrow} dx_2$。将其通过函子 const 传递，以得到态射 ms

$$\mathsf{const}\, e_{x_1} \colon \mathsf{const}\, d' \to d_i$$

使得下图交换。

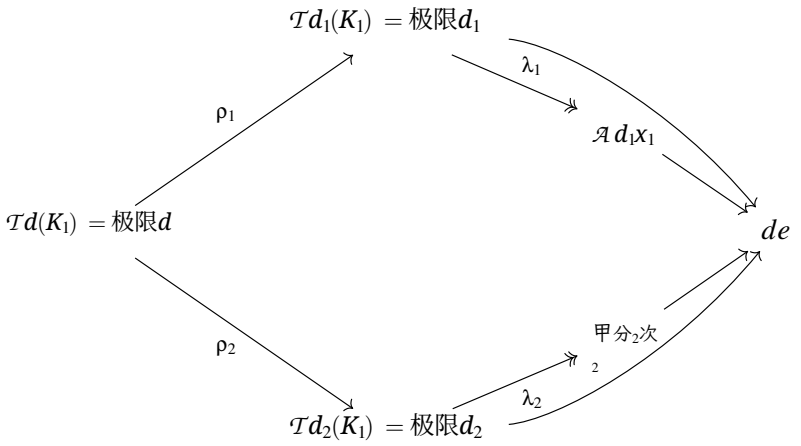


这些观察表明，以下是有限集范畴中的一个拉回图。

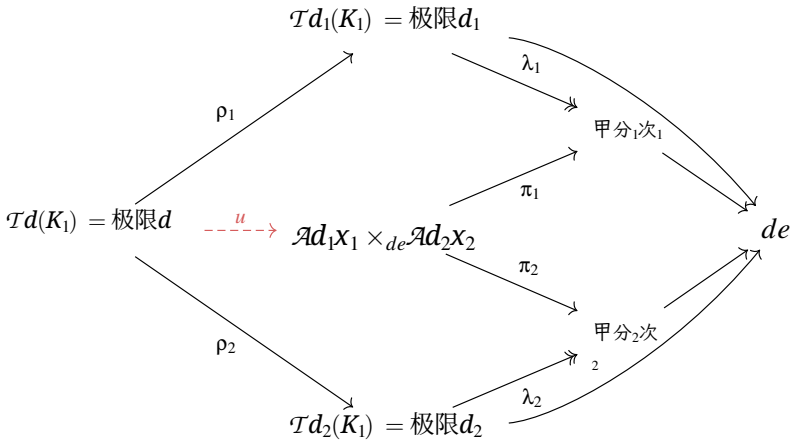


(19)

我们可以进一步将此图表分解如下（其中 λ_i 是顶点为 $\lim d_i$ 的锥腿）。



由此可见，剩下需要证明的只有下图所示的唯一拉回箭头u



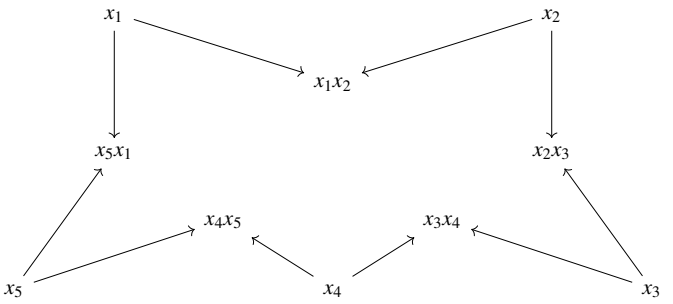
是一个满射。要理解为何这就足够了，请注意，若u 是满射，那么整个论断便成立，因为我们有

$$\lambda_i\rho_i = \pi_i u \implies \mathsf{Im}\,\lambda_i\rho_i = \mathsf{Im}\,\pi_i u = \mathsf{Im}\,\pi_i|_{\mathsf{Im}\,u} = \mathsf{Im}\,\pi_i.$$

但由此便得证，因为只要回想起u 的满射性显然成立：Ad_ix_i 的定义是Im λ_i，并且 $\lim d = \lim d_1 \times_{de} \lim d_2$（如图表19所示）。□

请注意，引理 4.2 不能直接推广到任意形状的分解。例如，

考虑以下五环图的循环分解示例，其顶点为 $\{x1$、……、$x_5\}$。



将这个分解通过二着色层 $\mathsf{SimpFinGr}(-,\ K_2)$ (即应用函子 $\mathfrak{D}_{\mathrm{m}}\,\mathsf{SimpFinGr}(-,\ K_2)$) 传递后，我们得到一个取值于 余分解 δ的结构，其值域为 有限集范畴，且其中所有袋都是二元素集合，对应于任意一条边的两种合法着色。现在注意到，由于奇环不可二着色，$\mathcal{A}\delta$ 的至少一个袋必为空。与此相反，不难验证，算法 1 在 δ 上的输出与其自身同构。 δ

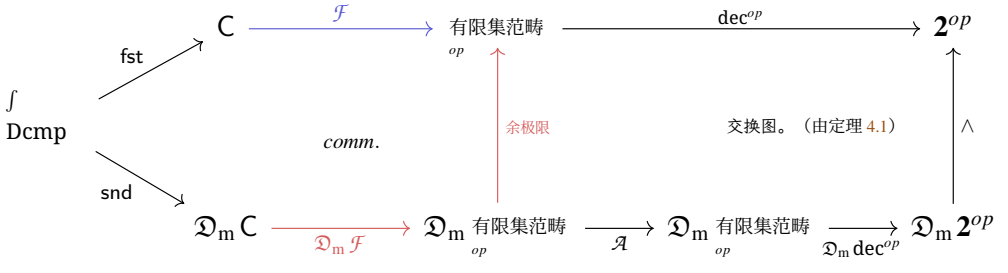
尽管这些观察似乎表明，我们无法在非树形分解上取得算法结果，但如果我们愿意接受较慢的运行时间（这些时间仍为固定参数可解时间，不过采用的是双参数化，而非引理 4.2中的单参数化），那么我们也能高效地求解其他形状分解上的层判定问题。这就是定理 1.1，我们终于可以证明它了。为清楚起见，我们想指出，若给定的分解形状为树，则以下结果与引理4.2完全相同：树的反馈顶点数显然为零。

定理 4.3。（定理1.1重述）
设 G 是一个有限的、非自反的、不含反向平行边的有向图，且每对顶点之间至多有一条边。设 D 是一个小型粘着余完备范畴，设 $\mathcal{F}\colon \mathsf{D}^{op}\to$有限集范畴是一个预层，并令 C 为 $\{\mathsf{D},\ \mathsf{D}_{mono}\}$ 中的一个。若 $\mathcal{F}$是位点 $(\mathsf{C},\mathsf{Dcmp}\,|_{\mathsf{C}})$ 上的层，且给定一个算法 $\mathcal{A}_{\mathcal{F}}$，该算法能在时间 $\mathcal{F}$ 内计算任意对象 c 上的值，且时间为 $\alpha(c)$，那么存在一个算法：给定任意对象 $c\in\mathsf{C}$ 的 C 值结构化分解 $d\colon\ \int G\to\mathsf{C}$ 以及 G 的反馈顶点集 S，可在时间内计算 $\mathsf{dec}_{\mathcal{F}}c$

$$O(\max_{x\in VG}\alpha(dx)+\kappa^{|S|}\kappa^2)|EG|$$

其中 $\kappa=\max_{x\in VG}|\mathcal{F}dx|$。

证明：
[定理 1 的证明。1] 回想一下，根据定理**(A3)** 的要点 4.1 下图交换。



我们的证明将依赖这一事实以及以下断言。

命题 4.4。考虑 d 在 $\mathfrak{D}_{\mathrm{m}}$F 下的像，并将其视为取值于有限集合范畴的结构余分解$\mathfrak{D}_{\mathrm{m}}$F d : $(\int G)^{op} \to$有限集范畴，固定任意顶点 x $\in$V G，以及与 x 相 incident 的边的任意枚举 $\ell = (e1、\ldots, e_n)$。令 γ_s 为将算法 2 应用于输入 ($\mathfrak{D}_{\mathrm{m}}$F d ℓ) 时的输出，并令 G′ $\hookrightarrow$G 表示通过移除所有与 x 相 incident 的边而得到的 G 的子图，定义 δ_s 为分解 δ_s : $(\int G')^{op} \hookrightarrow (\int G)^{op} \xrightarrow{\gamma_s}$ 有限集范畴。

如果 $\mathfrak{D}_{\mathrm{m}\mathcal{F}} d(x)$ 是单元素集，则 $\wedge \circ \mathrm{dec}^{op} \circ \mathcal{A} \circ \mathfrak{D}_{\mathrm{m}\mathcal{F}})(d) = (\wedge \circ \mathrm{dec}^{op} \circ \mathcal{A})(\delta_s)$。

证明:
[命题 4.4] 如果袋 $\mathfrak{D}_{\mathrm{m}\mathcal{F}}\, dx$ 恰好有一个截面，我们称之为 ξ，那么 $\mathcal{F}$ 在 d 上的每个匹配族都必须包含 ξ。注意，显然 $\lim \mathfrak{D}_{\mathrm{m}\mathcal{F}} d = \lim \gamma_s$，因为算法 1 执行的局部拉回（因此算法2 也如此）不会改变整体极限，所以我们有

$$(\wedge \circ \mathrm{dec}^{op} \circ \mathcal{A} \circ \mathfrak{D}_{\mathrm{m}\mathcal{F}})(d) = (\wedge \circ \mathrm{dec}^{op} \circ \mathcal{A})(\gamma_s).$$

但现在注意，由于 $\mathfrak{D}_{\mathrm{m}\mathcal{F}}\, dx$ 是单元素集合，任意一组截面 $(\zeta \in \delta_s(z))_{z \in VG'}$ 都必定构成 d 上的匹配族，因为根据 δ_s的构造，族中的每个此类截面 ζ 都必须与 ξ一致。但这样一来，我们便如愿证明了

$$(\wedge \circ \mathrm{dec}^{op} \circ \mathcal{A} \circ \mathfrak{D}_{\mathrm{m}\mathcal{F}})(d) = (\wedge \circ \mathrm{dec}^{op} \circ \mathcal{A})(\delta_s).$$

□

鉴于这一结果，注意，若对每个 s $\in$S（回顾一下，S 是 G 中的反馈顶点集），F ds 都是单元素集合，那么通过反复应用命题 4.4，直到输出分解成为森林，再应用引理 4.2 中的算法，我们就能在时间 $O(\max_{x \in VG} \alpha(dx) + \kappa^2)|EG|$ 内正确求解层判定问题（因为整个过程相当于调用一次算法 A$_{\mathcal{F}}$来计算$\mathfrak{D}_{\mathrm{m}}$F d，并调用 O(|EG|) 次边筛选算法——即算法 1）。

现在，从直观上看，如果存在 $s \in S$，使得 $\mathcal{F}ds$ 不止包含一个元素，那么我们只需对 $\mathcal{F}ds$中的每个截面重复上述过程一次。更正式地说，对每个$\sigma \in \prod_{s \in S} \mathcal{F}ds$，定义取值于有限集合范畴的结构余分解 ω_σ: $(\int G)^{op} \to$有限集范畴：将 $\mathfrak{D}_{\mathrm{m}\mathcal{F}} d$ 中所有形如 $\mathfrak{D}_{\mathrm{m}\mathcal{F}} ds$ 的袋替换为袋 $\omega_\sigma s := \{\sigma_s\}$（其中 σ_s 是元组 σ中指标为 s 处的截面）。那么，由于$\mathcal{F}$是层，可得

$$\mathrm{dec}^{op}\, \mathcal{F}\, c = \bigvee_{\sigma \in \prod_{s \in S} \mathcal{F}\, ds} (\wedge \circ \mathfrak{D}_{\mathrm{m}}\, \mathrm{dec}^{op} \circ \mathcal{A})(\omega_\sigma).$$

这将具有所需的运行时间，因为它对应于先计算$\mathfrak{D}_{\mathrm{m}\mathcal{F}} d$（使用算法 $\mathcal{A}_{\mathcal{F}}$，我们可以在时间 $O(\max_{x \in VG} \alpha(dx))$ 内完成），然后至多运行 $(|\prod_{s \in S} \mathcal{F}ds| \in O(k^{|S|}))$ 次算法 2（其正确性和运行时间已由命题 4.4 和引理 4.2确立）。□

读者或许会注意到，到目前为止我们一直避免提及输入分解的宽度概念，而且这些考量确实没有出现在算法的陈述中

引理4.2 或定理1.1。这是有充分理由的：应当注意，重要的并非输入分解的宽度，而是解空间分解的宽度，它决定了算法的复杂度界。在对象具有自然「大小」概念的范畴中（例如图范畴），人们或许会认为，用输入分解中最大袋的大小来表述算法的运行时间会更方便。然而，我们认为（与 Bodlaender 和 Fomin 先前的观察一致 [39]），用输入分解的宽度来表述算法运行时间会造成误导。要理解原因，可以考虑树分解上的着色算法这一非常简单但具体的例子。在这种情况下，只要满足以下两个条件，我们的分解中就完全可以包含很大的袋：1. 与这些袋相关联的局部解空间（它们是集合）很小；2. 能在有界时间内确定这些解空间。这一现象的一个简单却能说明问题的例子是：只要分解中的大袋（大小无上界）由完全图构成，我们就允许它们存在；对于这类图，除同构外只有一种正常着色，而且可以在线性时间内确定。

4.3. 实现

与传统的图分解组合定义相比， [6, 7, 4], 我们关于结构化分解的范畴论表述具有两个我们已经遇到的优势。

1. 对象无关性 结构化分解使我们能够描述任意粘合范畴中对象的分解，因此，每当遇到一种新的组合数据时，就不必针对具体情况逐一临时定义分解。

2. 函子性算法 函子性 $\mathfrak{D}_{\mathrm{m}}$ （即结构化分解范畴的函子性）使我们能够明确利用解空间及其分解。这样，我们便可以将算法结果的正确性表述为适当图表的交换性（例如图表（4））， 并且还可以据此推断运行时间瓶颈。

然而，范畴论视角还带来了另一项更为实际的益处：它使我们能够非常顺畅地从数学过渡到实现。事实上，我们在第4 节给出的理论算法结果，可以轻松地与 [37] AlgebraicJu-lia 生态系统 [38]中的相应实现配套使用。作为概念验证，我们已经实现了结构化分解，以及用于树形分解上的 SHEAFDECISION 的算法2 ；这展示了从数学到代码的无缝过渡，而一旦将范畴论作为核心理论抽象，读者便能亲身体验这一点。我们建议读者查阅相关代码仓库 [37]，以了解有关实现的更多细节。

5. A关于分解形状的一个开放问题 s

定理 1.1为以粘合范畴上的层编码的问题提供固定参数可解时间算法。实例化后，我们可以在许多具有算法研究价值的数学对象上获得算法结果，例如：

1. 数据库，2. 简单图，3. 有向图，4. 有向多重图，5. 超图，6. 有向超图，7. 单纯复形，8. 圆形端口图 [33]以及 9. 半边图。不过需要注意的是，只有满足以下条件时，这些参数化才有用：(1) 该类中并非所有对象都具有有界的结构分解宽度；(2) 该分解形状类的反馈顶点数有界。容易验证，对于上述所有例子，并非所有对象都具有有界的树形分解。然而，对于非树形的分解形状，所有对象相对于某个固定的分解形状类是否都具有有界宽度，这个问题并不显而易见。事实上，这将引出开放问题 6：在本节末尾，我们将自然地探讨我们的算法结果（涉及任意分解形状）与图论中更传统的结果相比如何；后者仅使用树形分解。特别是，下面我们将简要论证，对于图分解宽度（由 Carmesin 定义 [18]），我们的结果最终只在树宽有界的图类上给出固定参数可解算法。探究这些观察能否推广到更一般的对象类，是一个极具吸引力的新研究方向（参见开放问题 6）。

为此，首先注意，对于图 G 的任意 FinGr 值分解 $\mathrm{d}:\int H \to \mathrm{FinGr}$，容易看出，我们的形状图 H 的树宽 tw(H) 至多为其反馈顶点数。此外，如果形状图 H 的树宽至多为 t_H，且输入图 G 的 H-宽度至多为 t_G，那么我们可以轻松构造一个宽度为 $t_H \cdot t_G$ 的 G 的树分解，这意味着我们只需在树宽有界的图上进行计算。既然现在我们只处理图，采用 Carmesin 的 [18] 更具组合性的记法[13]会更方便。设 $\mathrm{T}\,T=(T,(X_t)_{t\in V}(T))$ 是 H 的树分解，$\mathrm{T}\,G=(H,(Y_t)_{t\in V(H)})$ 是 G 的 H-分解。我们声称，$\mathrm{T}\ =(T,(\cup_{t'\in X_t} Y_{t'})_{t\in V}(T)$ 是 G 的树分解。覆盖性显然成立，因为 T H 的每个袋都包含在 T 的至少一个袋中，而 T G 中覆盖了所有袋。关于相干性，论证如下。假设顶点 $v \in V(G)$ 属于两个袋 $t,t' \in V(T)$，但不属于 T 中从 t 到 t′的路径上的某个袋 t″。根据 T 中各袋的构造，存在一个顶点 $u \in Y_t$，其袋包含 v；同理，存在一个顶点 $u' \in Y_{t'}$，其袋也包含 v。由 T H 的相干性可知，T H 中存在一条连接 u 和 u′的路径 p，使得对于 $\tilde{t} \in V(p)$ 的所有 t，v 都属于袋 $Y_{\tilde{t}}$（H 中由包含 v 的袋所诱导的子图是连通的）。由于 v 不在 T 的袋 t″中，V (p) 中没有顶点属于 $X_{t''}$（T T 中），这与移除 $X_{t''}$ 会将 $u \in X_t \setminus X_{t''}$ 与 $X_{t'} \setminus X_{t''} \ni u'$分隔开相矛盾。

因此，如果类 H 中允许作为结构化分解形状的图的树宽有界，那么 H-宽度有界的图也具有有界树宽。因此，要获得新的固定参数可解算法，类 H 中图的树宽不应有界。另一方面，类 H 必须受到相当严格的限制，否则每个图的 H-宽度都会有界。例如，如果 H 包含 $n \times n$ 网格，那么类中的每个 n 个顶点的图的宽度都是 1，原因如下。考虑 $n \times n$ 网格 H，并设 $v_{i,j}$（$1 \le i,j \le n$）满足：对于$2 \le i \le n$，有 $v_{i-1,j}v_{i,j} \in E(H)$；对于 $2 \le j \le n$，有 $v_{i,j-1}v_{i,j} \in E(H)$。我们声称，$(H,(\{i,j\})_{v_{i,j}\in V(H)})$ 是任意顶点集为 $\{1,\dots,n\}$ 的图的 H-分解。覆盖性显而易见，因为对于每个 $1 \le i,j \le n$，我们都构造了一个包含 i 和 j 的袋，因此 $\{1,\dots,n\}$ 上的完全图得以覆盖。关于相干性，注意顶点 k 包含在袋 $X_{k,j}$（$1 \le j \le n$）和 $X_{i,k}$（$1 \le i \le n$）中，

[13]有关如何选择与 Carmesin 的图分解记号 [18]恰好对应的范畴论记号的图论实例化，详情请参阅 Bumpus、Kocis 和 Master [34]。

它们在网格H 中构成一个十字，因此这些袋是连通的。

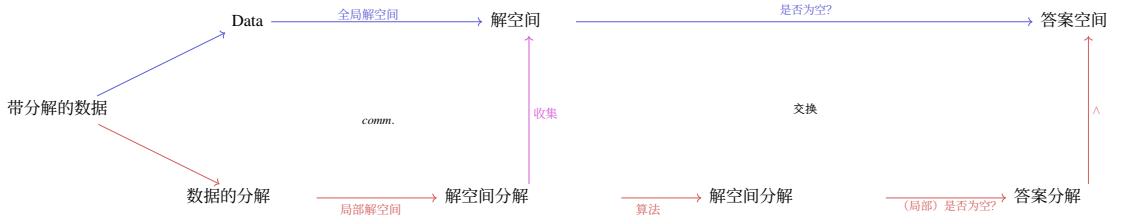
通过将与$v_{i,j}$ 收缩的顶点的袋设为袋$X_{v_{i,j}}$，并将所有被移除顶点的袋设为空，可以将这一构造推广到以$n \times n$-网格为图子式的图。因此，所有图的平面宽度至多为一。这引出了以下引人入胜的开放问题。

开放问题 6. 是否存在一个粘合范畴C，使得C中存在一个对象类χ，以及一个反馈顶点数有界的图类$\mathcal{G}$，令以下两个条件同时成立?

- 类χ具有**无界**树形结构分解宽度，并且
- 类χ具有**有界**$\mathcal{G}$形结构分解宽度。

7. 讨论

我们的主要贡献在于通过证明一个算法元定理（定理 **1.1**），连接**"结构"**与"能力"两个社群。该定理非正式地表明，将判定问题编码为层（**表征组合性**）后，可以通过动态规划（**算法组合性**）在线性时间内求解：其输入类能够进行宽度有界的结构化分解，且其分解形状的反馈顶点数有界（**结构组合性**）。因此，我们的结果通过展示如何运用范畴论与层论，将数学和理论计算机科学中发现的三种组合性融为一体，弥合了"结构"与"能力"这两个社群在数学和语言上的差异。下面的图表以高度概括的方式总结了这一点（即图表 1，其形式化版本为图表6）。



未来工作。 除开放问题6之外，进一步研究的方向还有很多。这里仅列举几个显而易见却令人兴奋的方向。首先，显然，尽管我们的元理论结果在横向一般性（就其适用的数学结构类型而言）方面达到了显著程度，但在纵向一般性（能够解决的问题范围）方面，仍不及Courcelle定理等更传统的结果 [23]。深入理解这些模型论方法与我们基于范畴和层论的方法之间的联系，是一个很有意思的后续研究方向。此外，我们提出以下两个更为具体的未来工作方向。

1. 研究其他类型的拓扑（例如由更宽松的包含关系（如图小参数）生成的拓扑）与由结构化分解所给出的拓扑之间的联系。

2. 对于其他问题——例如顶点覆盖或哈密顿路径——我们需要更多工具，因为尽管这些问题可以表示为预层，却并非层。事实上，值得注意的是，这种组合性「表面上」的缺失并非我们的方法所独有，在算法学中也同样存在：通过树分解上的动态规划求解哈密顿路径时，并不是将分解中的袋映射到局部哈密顿路径，而是映射到互不相交的路径集合（详情见 Flum 和 Grohe [24] ）。经验丰富的算法研究者会指出，通常可以遵循一种模板来选择恰当的部分解：考虑一个全局解，将其局部地限制到各个袋上，再利用这些信息确定算法组合性的障碍。从我们的角度来看，这很像是在问：「问题成为层的障碍是什么？我们又该如何系统地追踪这些信息？」幸运的是，层上同调中有许多强大的成熟工具，似乎适合完成这项任务。这是一个令人振奋的全新研究方向，我们已经在积极探索。

参考文献

[1] Abramsky S, Shah N. 关联结构与能力：计算资源的余单子语义。《逻辑与计算杂志》，2021。31(6):1390–1428。URL https://doi.org/10.1093/logcom/ exab048。[2] Diestel R.《图分解：无限图论研究》。牛津大学出版社，1990。[3] Diestel R.《图论》。施普林格，2010。ISBN:9783642142789。[4] Robertson N, Seymour PD. 图小参数 II：树宽的算法方面。《算法杂志》，1986。7(3):309–322。doi:10.1016/0196-6774(86)90023-4。[5] Robertson N, Seymour PD. 图小参数 X：树分解的障碍。《组合理论杂志》B辑，1991。52(2):153–190。doi:https://doi.org/10.1016/0095-8956(91)90061-N。[6] Bertelè U, Brioschi F. 非序列动态规划。学术出版社，1972。doi:https://doi.org/ 10.1016/s0076-5392(08)x6010-2。IS BN:9780124109827。[7] Halin R. 图的 S函数。《几何学杂志》，1976。8(1-2):171–186。doi:10.1007/BF01917434。[8] i Oum S. 有界秩宽图。博士论文，普林斯顿大学，2005。URL https: //mathsci.kaist.ac.kr/~sangil/pdf/thesis.pdf。[9] Geelen J, j Kwon O, McCarty R, Wollan P. 顶点小图的网格定理。《组合理论杂志》B辑，2020。URL https://doi.org/10.1016/j.jctb.2020.08.004。[10] Robertson N, Seymour P. 图小参数 XX：瓦格纳猜想。《组合理论杂志》B辑，2004。92(2):325 – 357。doi:https://doi.org/10.1016/j.jctb.2004.08.001。[11] Wollan P. 不包含固定浸入的图的结构。《组合理论杂志》B辑，2015。110:47–66。URL https://doi.org/10.1016/j.jctb.2014.07.003。

[12] Johnson T, Robertson N, Seymour PD, Thomas R. 有向树宽。《组合理论杂志》B辑，2001。82(1):138–154。URL https://doi.org/10.1006/jctb.2000.2031。[13] Berwanger D, Dawar A, Hunter P, Kreutzer S, Obdržálek J. 有向图的 DAG 宽度。《组合理论杂志》B辑，2012。102(4):900–923。doi:https://doi.org/10.1016/j.jctb.2012.04.004。[14] Hunter P, Kreutzer S. 有向图测度：Kelly 分解、博弈与序。《理论计算机科学》，2008。399。doi:https://doi.org/10.1016/j.tcs.2008.02.038。[15] Safari MA. D-宽度：一种更自然的有向树宽测度。载于：国际计算机科学数学基础研讨会。施普林格，2005，第 745–756 页。doi:https://doi.org/10.1007/ 11549345 64。_[16] Kreutzer S, Kwon Oj. 有界宽度有向图。载于：《有向图类》，第 405–466 页。施普林格，2018。doi:https://doi.org/10.1007/978-3-319-71840-8_9。[17] Bumpus BM, Meeks K, Pettersson W. 有向分支宽度：树宽的有向类比。arXiv 预印本 arXiv:2009.08903，2020。URL https://doi.org/10.48550/arXiv.2009.08903。[18] Carmesin J. 局部 2-分隔集。《组合理论杂志》B辑，2022。156:101–144。doi: https://doi.org/10.1016/j.jctb.2022.04.005。[19] Dujmovi´c V, Morin P, Wood DR. 小闭图类中的分层分隔集及其应用。《组合理论杂志》B辑，2017。127:111–147。URL https://doi.org/10.1016/j. jctb.2017.05.006。[20] Shahrokhi F. 平面图的新表示结果。arXiv 预印本 arXiv:1502.06175，2015。URL https://doi.org/10.48550/arXiv.1502.06175。[21] Jansen BMP, de Kroon JJH, Włodarczyk M. 以消除距离及更宽泛参数化的顶点删除。载于：第 53 届 ACM SIGACT 计算理论年会论文集，STOC 2021。美国计算机协会，纽约，美国。ISBN 9781450380539，2021，第 1757–1769 页。doi:10.1145/3406325.3451068。URL https://doi.org/10.1145/3406325.3451068。[22] Grohe M.《描述复杂性、规范化与可定义图结构理论》，剑桥大学出版社，剑桥，2017，x + 544 页。《符号逻辑通报》，2017。23(4):493–494。ISBN:1079-8986。[23] Courcelle B, Engelfriet J.《图结构与单子二阶逻辑：语言理论方法》，第 138 卷。剑桥大学出版社，2012。doi:https://doi.org/10.1017/CBO9780511977619。[24] Flum J, Grohe M. 参数化复杂性理论。2006。《理论计算机科学文本·EATCS丛书》，2006。doi:https://doi.org/10.1007/3-540-29953-X。ISBN:978-3-540-29952-3。[25] Cygan M, Fomin F V, Kowalik Ł, Lokshtanov D, Marx D, Pilipczuk M, Pilipczuk M, Saurabh S. 参数化算法。施普林格，2015。doi:https://doi.org/10.1007/978-3-319-21275-3。ISBN:978-3-319-21275-3。[26] Downey RG, Fellows MR. 参数化复杂性基础，第 4 卷。施普林格，2013。URL https://doi.org/10.1007/978-1-4471-5559-1。[27] Leray J. 表示的同调丛。巴黎科学院院报，1946。222:13661368。[28] Gray JW. 层论史片段。载于：《层的应用：层理论在逻辑、代数与分析中的应用研究研讨会论文集》，达勒姆，1977 年 7 月 9–21 日。施普林格，1966，第 1–79 页。URL https://doi.org/10.1007/BFb0061812。

[29] Rosiak D.《通过实例学习层论》。麻省理工学院出版社，2022。ISBN 9780262370424。URL 10. 7551/mitpress/12581.001.0001。[30] MacLane S，Moerdijk I.《几何与逻辑中的层：托扑斯理论初探》。施普林格科学与商业媒体，2012。URL https://doi.org/10.1007/978-1-4612-0927-0。[31] Oum Si，Seymour PD. 近似团宽度和分支宽度。《组合理论杂志》B辑，2006。96(4):514–528。doi:https://doi.org/10.1016/j.jctb.2005.10.006。[32] Bumpus BM，Meeks K. 时序图的边探索。Algorithmica，2022。第1–29页。URL https://doi.org/10.1007/s00453-022-01018-7。[33] Libkind S，Baas A，Patterson E，Fairbanks J. 动力系统的代数运算子建模：数学与计算。arXiv预印本 arXiv:2105.12282，2021。URL https://doi.org/10.48550/arXiv. 2105.12282。[34] Bumpus BM，Kocsis ZA，Master JE. 结构化分解：结构与算法复合性。arXiv预印本 arXiv:2207.06091，2022。URL https://doi.org/10.48550/arXiv.2207. 06091。[35] Patterson E，Lynch O，Fairbanks J. 面向技术计算的范畴数据结构。arXiv预印本 arXiv:2106.04703，2021。[36] Lack S，Sobocinski P. 粘合范畴。载于：Walukiewicz I（编），《软件科学与计算结构基础》。施普林格柏林海德堡出版社，柏林，海德堡。ISBN 978-3-540-24727-2，2004，第273–288页。doi:https://doi.org/10.1007/978-3-540-24727-2_20。[37] AlgebraicJulia/StructuredDecompositions.jl。访问日期：2023-02-10，URL https://github.com/ AlgebraicJulia/StructuredDecompositions.jl。[38] Patterson E，其他贡献者。AlgebraicJulia/Catlab.jl：v0.10.0。doi:10.5281/zenodo.4394460。URL https://zenodo.org/record/4394460。[39] Bodlaender HL，Fomin FV. 低成本树分解。载于：《算法理论——SWAT 2002：第八届斯堪的纳维亚算法理论研讨会，芬兰图尔库，2002年7月3日至5日，论文集 8》。施普林格，2002，第378–387页。[40] Riehl E.《语境中的范畴论》。Courier Dover出版社，2017。ISBN:048680903X。[41] 粘合范畴。https://ncatlab.org/nlab/show/adhesive+范畴。访问日期：2023-06- 02。[42] Duarte GL，de Oliveira Oliveira M，Souza US. 余退化度与余树宽：利用补图求解稠密实例。载于：Bonchi F，Puglisi SJ（编），第46届国际计算机科学数学基础研讨会，MFCS 2021，2021年8月23日至27日，爱沙尼亚塔林，LIPIcs第202卷。达格斯图尔城堡—莱布尼茨信息学中心，2021，第42:1–42:17页。URL https://doi. org/10.4230/LIPIcs.MFCS.2021.42。

A. 层论中的概念

这里汇集了本文中使用的基本层论定义。我们推荐读者阅读 Rosiak 的教科书 [29]，其中对层论的介绍适合初学者。

定义 A.1. 设 C 是一个具有拉回的范畴，并设 K 是一个函数，它为 C 中的每个对象 c指定一个由余域为 c 的态射构成的集合族，称为**覆盖集族**。如果以下三个条件对所有$c \in C$均成立，则称K 是 C 上的**格罗滕迪克预拓扑**。

(PT1) 如果$f\colon c' \to c$是一个同构，那么 $\{f\} \in K(c)$。

(PT2) 如果$S \in K(c)$，且$g : b \to c$ 是C中的一个态射，那么S 沿g 的拉回是b的一个覆盖集；也就是说 $\{g \times_c f \mid f \in S\} \in K(b)$。

(PT3) 如果 $\{f_i : c_i \to c \mid i \in I\} \in K(c)$，那么，只要给定

$$\{g_{ij} : b_{ij} \to c_i | j \in J_i\} \in K(c_i)$$

对于所有$i \in I$，我们必须有

$$\{f_i \circ g_{ij} : b_{ij} \to c_i \to c \mid i \in I, j \in J_i\} \in K(c).$$

定义 A.2. 对象$c \in \mathsf{C}$上的**筛**是由余域为c 的态射组成的族S_c，且该族在前复合下封闭。

筛本质上就是这样一种做法：我们只使用那些“饱和”的族（也就是说，它们在与C中的态射前复合后仍保持封闭）；引入筛是为了表述格罗滕迪克拓扑的定义（见 A.3），这是对 A.1的修订。

注意，（至少当 C 局部小的时候）筛可以被视为可表同态函子 $\hom(-,c) = y_c$ 的子函子（注意，C 上预层构成的范畴 $\mathbf{Set}^{\mathsf{C}^{op}}$ 有拉回，因此采用筛的处理方式后，我们可以去掉关于 C 本身的假设）。还要注意，若 $S \subseteq \hom(-,c)$ 是 c 上的一个筛，且 $f\colon b \to c$ 是指向 c 的任意态射，那么沿 f 的拉回筛

$$f^*(S) = \{g | \mathsf{cod}(g) = b, f \circ g \in S\}$$

是b上的一个筛。

通常，格罗滕迪克拓扑的定义是通过一个函数 J 给出的，它为 C 的对象指定筛，并满足三条公理（最大筛、基变换下的稳定性和传递性）。但实际上，稳定性公理——要求若 $S \in J(c)$，则对于任意箭头 $f\colon b \to c$，都有 $f^*(S) \in J(b)$——不过是要求 J 实际上是一个函子 $J : \mathsf{C}^{op} \to \mathbf{Set}$，也就是说，是预层范畴 $\mathbf{Set}^{\mathsf{C}^{op}}$ 中的一个对象。若令 $\mathsf{Sieve} : \mathsf{C}^{op} \to \mathbf{Set}$ 为这样一个函子：它将小范畴 C 的对象c 映到 c 上所有筛构成的集合，并对 C 中的任意态射 $f\colon x \to y$，使用映射

$$f^*\colon (S \in \mathsf{Sieve}(y)) \mapsto \big(\{g\colon w \to x \mid (fg : w \to x \to y) \in S\} \in \mathsf{Sieve}(x)\big),$$

那么，我们可以给出一个较为简洁的格罗滕迪克拓扑定义（将通常的稳定性公理预先纳入对J 作为特定函子的刻画中），如下所示。

定义 A.3. 设 C 是一个范畴，并考虑 $\mathsf{Sieve} : \mathsf{C}^{op} \to \mathbf{Set}$ 的子函子 J，其元素为“许可筛”。若 J 满足以下两个条件，则称其为 C 上的 **格罗滕迪克拓扑**：

(Gt1) 最大筛始终是许可的；即对于所有 $c \in \mathsf{C}$，都有 $\{f \in \mathrm{Mor}(\mathsf{C}) \mid \mathrm{cod}(f) = c\} \in J(c)$

(Gt2) 对于某个对象 $c \in \mathsf{C}$ 上的任意筛 R，如果在 c 上存在一个许可筛 $S \in J(c)$，使得对于所有 $(h : b \to c) \in S$，筛 $h^*(R)$ 在 b 上都是许可的（即 $h^*(R) \in J(b)$），那么 R 自身在 c 上也是许可的（即 $R \in J(c)$）。

对于任意 $c \in \mathsf{C}$，如果元素 $S \in J(c)$ 是 ***J*-覆盖**，或者在上下文明确 **覆盖** 的含义时简称为 **覆盖**。一个**位点**是一个二元组 (C, J)，由一个范畴 C 和 C 上的格罗滕迪克拓扑组成。

定义 A.4。设 (C, J) 是一个位点，S 是对象 $c \in \mathsf{C}$ 的一个 J-覆盖，且 $P : \mathsf{C}^{op} \to \mathbf{Set}$ 是一个预层。那么，相对于覆盖 S 的 P 的截面**匹配族**是预层之间的一个态射（自然变换）$\chi\colon\ S \Rightarrow P$。

定义 A.5. 设 $P : \mathsf{C}^{op} \to \mathbf{Set}$ 是位点 (C, J) 上的一个预层。若对于所有 $c \in \mathsf{C}$ 以及所有覆盖筛 $(\iota\colon S \Rightarrow y_c) \in J(c)$（属于 c），每个匹配族 $\chi : S \Rightarrow P$ 都有唯一延拓为态射 $\Sigma\colon y_c \Rightarrow P$，则称 P 是相对于 J 的**层**（或 J-层）。