通过最小代表形式的素项提取对有限镶嵌进行精确算法提取

Sushish Baral[1], **Paulo Garcia**[2] **和 Warisa Sritriratanarak**[1]*

[1]朱拉隆功大学计算机工程系，曼谷，泰国 d

[2]朱拉隆功大学国际工程学院，曼谷，泰国

6773005021@student.chula.ac.th, paulo.g@chula.ac.th, warisa.s@chula.ac.th

* 通讯作者：Warisa Sritriratanarak (warisa.s@chula.ac.th)

arXiv:2603.00911v1 [cs.CV] 1 Mar 2026

摘要

在符号推理、算法合成及各类计算领域的结构优化中，识别离散网格中的重复模式仍处于初级阶段。尽管针对噪声数据的统计方法能够近似识别模式，但利用确定性提取来分析周期结构的符号分析方法仍不成熟。本文旨在弥补这一空缺，采用一种分层算法，在有限平面网格中发现精确镶嵌，并处理分层结构中可能并存多个独立模式的问题。所提出的方法采用复合体发现（双重检查和广度优先剪枝）来识别具有内部重复的矩形区域，将其归一化为最小代表形式，并通过素项提取（选择性复制和分层记忆化）处理不规则尺寸，从而实现高效计算。我们评估了从 2×2 到 32×32 各种网格尺寸下的可扩展性，结果显示，对简单重复图块进行重叠检测的处理时间不到 1 毫秒，而需要穷举搜索和系统探索的复杂模式则呈指数级增长。该算法可对精确、轴对齐的矩形镶嵌提供确定性处理，弥补了符号网格分析技术中的一个关键空缺，可应用于谜题求解推理任务以及离散符号域中精确重复结构的识别。.

1 引言

密铺[1] 作为一种建筑和艺术特色，已有数千年的历史记载[2]。对其性质的数学研究，尤其是周期铺砌[3]，始于19世纪末 Fyodorov 的研究[4],，而彭罗斯则开创性地探索了非周期铺砌[5]（最近还发现了首个非周期单块砖[6]）。尽管关于密铺的数学性质和周期瓦片生成已有大量研究[7],，但对于密铺模式的确定性精确识别（即 提取）却鲜有研究。《模式识别》领域[8]（以及近来兴起的机器学习子领域[9]）已经发展出统计提取技术[10],，主要用于从现实世界的模糊数据（例如图像[11]）中提取密铺；然而，作为一种算法技术，精确提取仍未得到充分发展。这主要有两个原因：a）对于大多数现实应用而言，数据都是模糊的，因此统计解法既必要又充分；b）该问题的计算复杂度较高，若干边界情况给算法解法带来了挑战。

本文探讨了在平面网格[12]上提取精确铺砖模式的算法解决方案及其相关数据结构。具体而言，我们研究（有限）平面网格的镶嵌[13] 识别问题：同一网格上可能存在多个彼此独立、各不相同的镶嵌，而一个镶嵌也可能由层次化的铺砖模式[14]构成。这比识别严格周期性的平面镶嵌更为一般。此问题在多种应用中都会出现，例如路由矩阵[15]（重复出现的图块代表常见路由模式，可据此进行优化）；路径规划[16]（识别出重复图块后，便可重用先前缓存的解）；二维世界中的因果推断（铺砖模式由其他结构产生，或会影响其他结构；例如 ARC-AGI[17]）；以及其他应用。

本文的工作源于作者在 ARC-AGI 挑战上的研究，其中镶嵌提取在若干谜题中起着关键作用，而机器学习迄今尚未取得令人满意的表现。具体而言，本文做出如下贡献：

• 我们提出一种用于二维结构上精确铺砌模式提取的算法。所述算法能够确定性地识别满足给定密铺的基本铺砖。

• 我们描述并评估了优化技术（发现最小代表形式并进行归一化）的性能。这些技术通过利用密铺固有的对称性来剪枝搜索空间。

• 我们提供了一个开源参考实现[1] 并评估了它在一系列代表性示例中的性能。

本文其余部分安排如下。第 2 节给出若干动机示例，说明本文所述技术的必要性，并对问题进行形式化。本文结构如下：第 3 节介绍所提出的算法及用于搜索空间剪枝的相关优化技术。第 4 节介绍用于评估本文贡献的实验设置。第 5 节概述该领域的最新研究进展，第 6节总结全文。

2 背景

2.1 动机示例

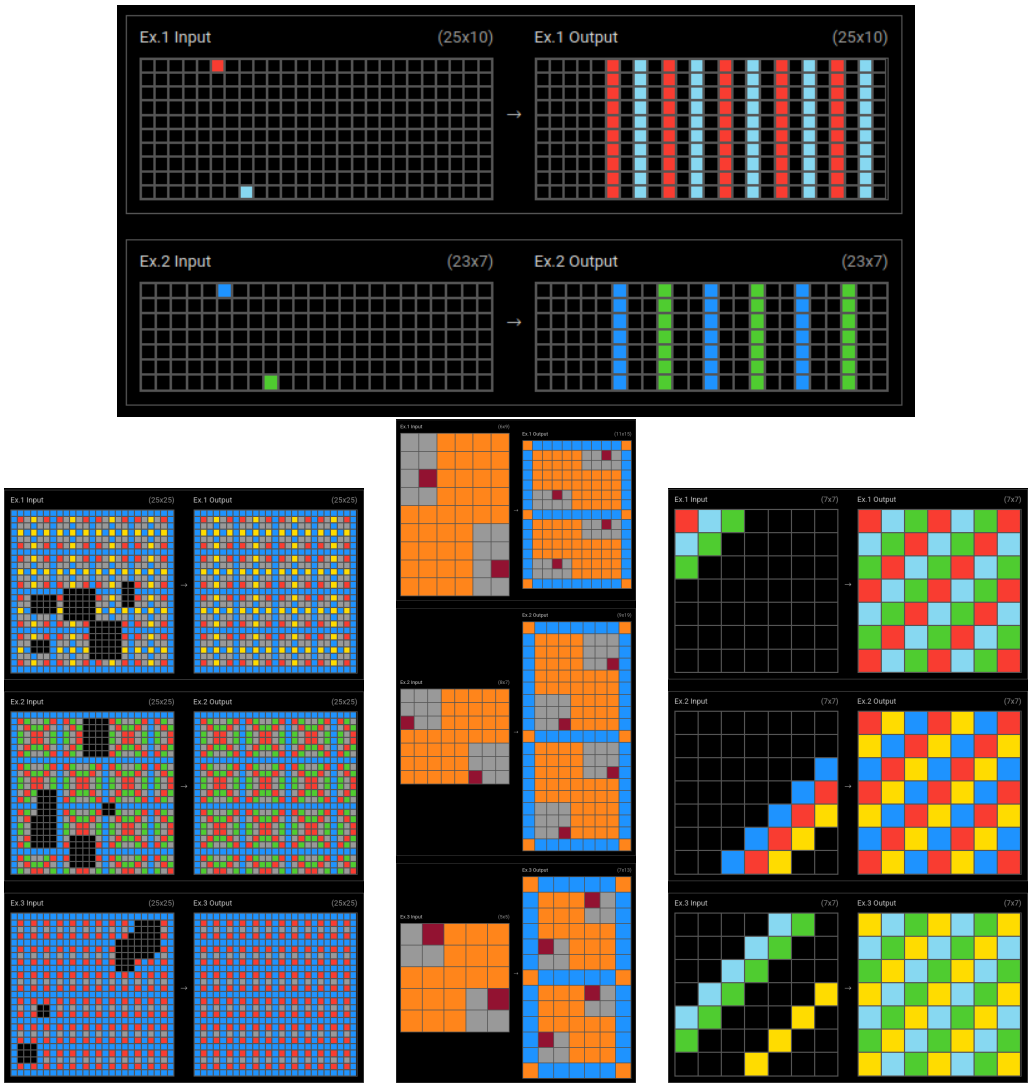


图 1。 ARC任务；转载自 https://arcprize.org/play?task=0a938d79

[1] https://github.com/paulo-chula/Exact-Tesselation-Extraction

ARC-AGI 挑战[17] 由一系列测试组成，其中训练测试公开可用，而评估测试则不公开。在每项测试中，一组示例输入-输出对（通常为 3 至 5 组）展示了一个特定过程：目标是推断该过程，并为给定输入生成输出。在训练测试和评估测试中，存在若干不同的过程。实证评估表明，人类能够推断出所有过程[18],，但迄今为止，机器学习解法尚未展现出很强的能力[19]（这主要是因为该挑战缺乏大量训练数据），尽管测试时训练最近已显示出希望[20]。

程序合成，即根据示例对推断出能够从相应输入生成输出的程序，是一种很有前景的解决方案[21], 尤其是在应用于能够高效表达领域语义的领域特定语言（DSL[22]）中[23]。请看图1顶部所示的公开训练数据集中的示例测试。其解决方案可以用自然语言描述如下：

- “将点延伸成线，延伸方向垂直于它们所邻接的‘墙’。”
- “朝网格空的一半方向，以交替方式复制这些线，使线与线之间的距离保持恒定。”

在领域特定语言或通用语言中实现这样的解决方案也很简单；但是，尽管人类能够轻松做到，首先通过计算推断出这样的解决方案并不容易。一种方法是分析输入-输出对并找出常见模式。在本例中，解决方案必须正确识别出：(1) 所有输出网格都呈现出镶嵌；(2) 镶嵌占据的面积受到限制，而且这些限制可以从输入网格中推断出来；(3) 存在一个子计算（将点延伸成线），能够根据输入网格生成适用于此类输出镶嵌的有效图块。

令我们惊讶的是，竟然不存在能够精确识别镶嵌及其对应牌块的算法，这促成了本文的研究。

2.2 预备知识

有限平面网格 **G** 定义为一个$n \times m$矩阵，其中$n, m \in \mathbb{N}$；每个元素$g_{i,j} \in \mathbb{N}, \forall i \in \{1:n\}, \forall j \in \{1:m\}$中的值，都可以看作与用于直观表示网格的颜色集合构成同构。由于具体的同构关系与算法层面无关，因此在本文余下部分中，我们将单元格中的值记为自然数，并将网格直观地表示为矩阵，而不显式或一贯地将数字映射为颜色，这不会损失通用性。我们采用标准的矩阵记法，使元素$g_{1,1}$ 位于网格左上角，元素$g_{n,m}$ 位于网格右下角。

定义 2.1。在 G 中，矩形边界框（记作 R）框定 G 的一个矩形子区域。矩形边界框是一个元组 $R = (x, y, l, h)$，其中 $x \in [1:n], y \in [1:m]$是框左上角的坐标，而 $l \in [1:n], h \in [1,m], (l > 1) \cup (h > 1)$分别是其长度和高度；$((x + l) \leq (n - 1)) \cap ((y + h) \leq (m - 1))$；若 $(x < i < (x + l - 1)) \cap (y < j < (y + h - 1))$，则认为元素 $g_{i,j}$ 位于该边界框内。

引理2.2。 如果**G** 的尺寸为 $n \times m$，则**G**中可能存在 $nm((n+1)(m+1)-4)$ 个矩形包围盒。

证明。由于矩形由其在 x、y轴上的投影定义，且长度为 l 的线段在平行轴上有 $l(l+1)$ 2 个整数子投影，因此投影的组合数为 $n(n+1) \times m(m+1) = n(n+1)m(m+1)$ 4。这包括所有尺寸为 1×1的 $n \times m$ 个矩形（正方形），必须从总数中减去：因此，$n(n+1)m(m+1)$ 4 $\neg n \times m = nm((n+1)(m+1)-4)$ 4。 □

定义2.3。矩形边界框 $R_u = (x_u, y_u, l, h)$ 是 G 中的矩形瓦片，当且仅当 $\exists R_v = (x_v, y_v, l, h) \rightarrow ((x_v = x_u + l, y_v = y_u) \cup (x_v = x_u - l, y_v = y_u) \cup (x_v = x_u, y_v = y_u + h) \cup (x_v = x_u, y_v = y_u - h)) \cap (\forall (a \in [0,l], b \in [0,h]), g_{x_u+a, yu+b} = g_{x_v+a, yv+b})$。

推论2.4。 如有需要，可以简单地扩展矩形瓦片的定义，使其包含旋转（针对正方形瓦片）和对称性。

也就是说，我们将图块定义为**G** 中的矩形边界框：水平平移l 或垂直平移h后，它们与新位置原有的值完全匹配。

定义2.5。 一个矩形瓦片的密铺 T t 是 **G** 的一个连续子集，可以从一个图块 t 开始，通过添加 t 的连通平移绘制而成，其中所有水平平移的距离均为 l，所有垂直平移的距离均为 h。此外，密铺也可以是图块的一部分（图 2）。

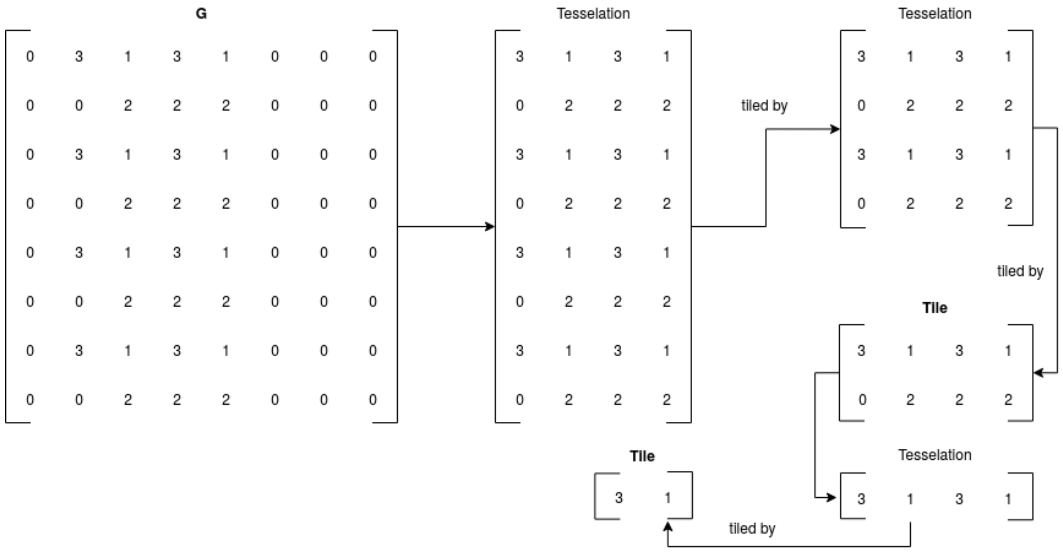


图 2. 示例网格及已识别的镶嵌，并标示与相应牌块的连接。

请注意，在数学中，通常将（周期性）图块定义为几何形状（通常以单一颜色的形式直观呈现），这些形状可以排列成密铺平面的形式。相比之下，我们关注的是作为矩形边界框的图块（它们表示几何形状的空间周期，因此必然会出现在任何足够大的镶嵌中），因为这足以确定所有可能镶嵌中的一个大子集，并能最小化所有可能图块的搜索空间。

3 层次复合与素元发现

3.1 定义

我们定义以下贯穿本方法所使用的关键概念：

复合体：矩阵面积内表现出内部重叠的区段，即按行/按列分割会产生相同的两半。例如， 4×6 图块 1212/3434/1212/3434 是复合体，因为按行分割会得到两个相同的 2×6 半部。

规范化复合体：复合体经迭代二分归约为最小代表形式后，即为归一化复合体。复合体 12121212（1×8）规范化为 1212（1×4），再经连续列约简后变为 1212（1×2）。

素数：素数是构建整个表面的基本图块。它是无法进一步分解的原子瓦片。素数是通过铺排重建原始复合体所用的构件。

重叠：将一个图块叠放在另一个图块上进行比较的过程，在图块按行或按列分成两半后执行。根据所处阶段，重叠检测可在有复制或无复制的情况下进行。

剪枝：从矩形瓦片的一条边（上、下、左或右）移除一行或一列，从而生成更小的子瓦片。剪枝可通过广度优先搜索，穷尽探索所有子矩形。

分支切断：当某个节点检测到重叠时，终止对 BFS 树的探索。这可以避免探索冗余子树，因为可以保证所有后代节点都包含在已发现的复合体或素数中。

复制：在拆分之前复制中间行（适用于行数）或中间列（适用于列数），以便在奇数维结构中进行重叠检测。此过程有选择地应用于检查和归一化阶段。

分层筛选：（记忆化）在素元发现阶段跳过复合体、不再对其进行进一步剪枝的过程；这些较小的复合体已在素元剪枝期间被探索和测试，该过程由按降序进行的基于面积的排序所实现。

广度优先搜索层级：在广度优先剪枝过程中探索图块的深度。第0层是初始矩阵，第 1 级包含直接剪枝得到的图块，第 2 级包含对第 1 级图块进行剪枝后得到的图块，以此类推。

累积策略：一种分解方法，逐步组合来自多个广度优先搜索层级的素数（先组合第 1 级，再组合第 1 至 2 级，然后组合第 1 至 3 级，以此类推），从而在所有粒度下找到所需图块放置数量最少的解。
逐层策略：一种分解方法，针对每个广度优先搜索层级（仅第 1 级、仅第 2 级等）分别求解铺砖问题，从而揭示解的质量如何随图块粒度变化，并支持分析图块尺寸与放置数量之间的权衡。

3.2 高层方法论

给定一个尺寸为$n \times m$的平面网格G，我们采用三阶段分层流程来识别镶嵌：**(1)复合体发现**，用于识别网格上所有存在内部重叠的区域；**(2) 归一化**，使用重叠技术将复合体简化为其最小代表形式；**(3) 素项提取**，采用两种相互独立的策略，将归一化复合体分解为更小的构件。

我们方法的核心洞见是，在分解过程中，较大的复合体往往包含较小的复合体。为避免冗余，我们采用分层筛选/记忆化：按尺寸降序处理复合体，并将剪枝较大复合体时发现的任何复合体排除在后续处理之外，从而在保持完备性的同时显著降低计算开销。

我们用于素项提取的独立策略，从两个互补的角度考察分解粒度。首先，该策略会跨剪枝层级/深度组合素数候选项，以找到瓷砖放置数量最少的整体解。其次，逐层策略会独立评估每个剪枝层级/深度，从而提供不同尺度的替代分解方案。

在处理奇数维网格时，我们采用选择性复制技术：仅在检查和归一化阶段进行复制，此时一次重叠检查必须明确识别或缩减某个模式。在这些阶段中，我们会在拆分前复制中间行（当行数为奇数时）或中间列（当列数为奇数时）。关键的是，我们不会在广度优先搜索剪枝阶段进行复制，因为对所有侧面逐步剪枝，可确保在某个节点上因奇数维度而被遮蔽的模式，会在后续具有偶数维度的节点上显现。

此外，在素项提取阶段，我们的算法会检测嵌套镶嵌。当广度优先搜索剪枝在素项提取过程中发现具有重叠部分的子节点时，我们会对重叠部分进行归一化，并将其记录为可复用模式，同时记录最大使用次数信息，以确保素数始终以规范形式存储，并正确识别嵌套结构。

3.3 算法描述

算法1 通过三个阶段发现复合体。第一阶段是**检查未裁剪网格/原始矩阵**，通过将输入分成两半并进行比较，检查整个输入是否存在行或列重叠。对于维度为奇数的输入（例如，5 行），分割前会复制中间行，以实现重叠。如果检查未通过，第二阶段**裁剪输入中的空白边框**，然后再次检查处理后的网格，同样使用复制操作。如果两次检查都未通过，算法将执行第三阶段：**对裁剪后的网格进行广度优先搜索剪枝**，系统地通过沿四个方向（上、下、左、右）移除边缘来探索所有子矩形。

图 7 展示了用于复合体发现的广度优先搜索剪枝。从根节点 Mk（`10103/02000/10100/02003`）开始，树在每一层通过剪枝（上、下、左、右）扩展。绿色节点继续剪枝，红色节点表示应用分支剪枝后发现的复合体，橙色节点是已访问集合跳过的重复项，蓝色节点表示接近最后一层剪枝的节点。底部发现了三个复合体。

算法 2 通过带复制的迭代二分，将复合体规范化为更小的形式。该算法反复尝试按行和按列拆分复合体，并检查两半是否相同。如果两半匹配，且得到的半部满足最小尺寸，即 (2x1/1x2)，则用对应原子体替换该复合体。对于奇数维度，会复制中间行或中间列，然后执行上述相同过程。复制至关重要，因为如果不进行复制，奇数维度的复合体就无法均匀拆分，因此会一直处于未规范化状态。

图 4展示了迭代归一化的过程。三个存在行重叠的复合体（Tc，粉色）经过连续的带复制二分，缩减为各自的最简形式（Tc’，紫色）。例如，`1010-/0200-/1010-/0200-` 经过一次按行二分迭代后，缩减为`1010-/0200-`。

算法 3 使用广度优先搜索剪枝从复合体中提取素数，方式与算法1类似，但有一个关键区别：当在子节点中检测到重叠时，会对重叠部分进行规范化（通过算法 2）。如果归一化后的图块能够重复铺满整个子节点，就会记录该图块及其重复次数（最大使用次数）。例如，如果子节点 1212 存在列重叠，我们会将重叠部分规范化为 12，并验证它恰好可以铺满 1212 两次。这些镶嵌信息会告知谜题求解器该图块可以重复使用多少次。随后会应用分支切断，避免继续探索其后代节点，因为它们将是已识别模式的片段。

图 5 展示了从规范化复合体 Tc’ (`1010-/0200-`) 开始通过广度优先搜索进行素项提取的过程。紫色节点表示素项，橙色节点表示重复项，深红色节点表示空项（已舍弃）。检测到重叠时，会对图块进行规范化，并检查是否构成镶嵌。底部面板按发现层级（第 1–4 级）排列素项，用于双策略MDL组装。

算法1复合体发现

要求： 网格 G，尺寸为 $n \times m$

确保： 复合体集合

过程DISCOVERCOMPOSITES(G)

▷ 检查未裁剪网格（含复制）

(顶行,底行) ← SplitRows(G, 若为奇数则复制=true)

如果上部 =下部且上部并非全为空白**则**

  返回 {上部}

(左侧,右侧) ← SplitCols(G, duplicateIfOdd=true)

如果左侧 = 右侧且左侧并非全为空白**则**

  返回 {左侧}

▷ 对裁剪后的网格进行检查（含复制）

G' ←裁剪外围空白(G)

(顶部，底部) ← SplitRows(G', 若为奇数则复制=真)

如果 顶部 =底部 且顶部 并非全为空白**则**

  返回 {顶部}

(左侧，右侧) ← 分割列(G', 若为奇数则复制=真)

如果 左侧 = 右侧 且 左侧 并非全为空白 **则**

  返回 {左}

▷ 广度优先搜索剪枝（不进行复制）

$Q \leftarrow \{G'\}$，已访问 $\leftarrow \{G'\}$，$\mathscr{C} \leftarrow \emptyset$

当 $Q \neq \emptyset$ **时执行**

  节点 ← 将其从队列(Q)中取出

  $\tilde{T}$ ←对其进行剪枝(节点), $\tilde{B} \leftarrow$ 剪枝底部(节点)

  $\tilde{L} \leftarrow$ 剪枝左侧(节点), $\tilde{R} \leftarrow$ 剪枝右侧(节点)

  对于子节点 在 $\{\tilde{T}, \tilde{B}, \tilde{L}, \tilde{R}\}$ **执行**

    如果 子节点 为空项或不满足最小尺寸**则**

      继续

    如果子节点 ∈已出现 **则**

      继续

    (上，下) ← SplitRows(子节点，若为奇数则复制=false)

    如果(上，下) ≠ 为 null 且 上 = 下且上并非全部为空白 **则**

      $\mathscr{C} \leftarrow \mathscr{C} \cup \{$上$\}$

      继续

    (左侧，右侧) ← 分割列(子节点, 若为奇数则复制=false)

    如果 (左侧,右侧) ≠为空且 左侧 = 右侧 且 左侧 并非全部为空白 **则**

      $\mathscr{C} \leftarrow \mathscr{C} \cup \{$左侧$\}$

      继续

    将(Q，子节点)加入队列，已访问 ← 已访问 ∪{子节点}

返回 RemoveDuplicates($\mathscr{C}$)

算法 2 归一化

要求： 复合瓦片T_c
确保： 规范化复合体 T_c'（最简形式）

过程NORMALIZE(T_c)
    约简 $\leftarrow 0$
    重复
        已缩减 $\leftarrow$假
        (顶部，底部) $\leftarrow$ SplitRows(T_c, 若为奇数则复制=真)
        如果顶部 = 底部并且 满足最小尺寸(顶部) **那么**
            $T_c \leftarrow$顶部
            缩减 $\leftarrow$ 缩减 $+1$
            已缩减 $\leftarrow$真
            继续
        (左，右) $\leftarrow$ 分割列(T_c， 若为奇数则复制=true)
        如果左 = 右并且满足最小尺寸(左) **那么**
            $T_c \leftarrow$ 左
            约简 $\leftarrow$ 约简 $+1$
            已约简 $\leftarrow$真
    直至 ¬缩减
    返回 T_c

算法 3 通过 BFS 剪枝提取素数

要求： 规范化复合体T_c
确保： 素数集合 $\mathscr{P}$，镶嵌$tess$

过程EXTRACTPRIMES(T_c)
    $tess \leftarrow$ 查找镶嵌(T_c)
    $\mathscr{P} \leftarrow /0$
                ▷ 从算法1开始应用广度优先搜索剪枝，起始值为T_c
            ▷区别：发现重叠时，进行规范化并检查是否为镶嵌
    对广度优先搜索剪枝过程中发现的每个子节点 **执行**
        如果 子节点 未达到最小尺寸（2x1 或 1x2）**则**
            继续
        如果 子节点 存在重叠（行或列）**则**
            素数 $\leftarrow$ 规范化(重叠部分) ▷算法 2
            如果 TilesMatrix(素数, 子节点)**则**
                $tess \leftarrow tess \cup \{$(素数, 最大_使用次数)$\}$
            $\mathscr{P} \leftarrow \mathscr{P} \cup \{$素数$\}$
            应用分支切断 ▷不要继续剪枝
        else
            $\mathscr{P} \leftarrow \mathscr{P} \cup \{$子节点$\}$
            继续剪枝
    返回 $\mathscr{P}$, $tess$

算法 4 层次化素数提取与双策略最小描述长度

要求： 归一化复合体集合 $\{T_{c_1}, \ldots, T_{c_k}\}$
确保： 每个复合体的最优解 $\mathscr{S}$

过程 层次化提取($\{T_{c_1}, \ldots, T_{c_k}\}$)
  按面积降序排列复合体
  已处理 $\leftarrow \emptyset$, $\mathscr{S} \leftarrow \emptyset$
  对每个复合体 T_{c_i} 按排序顺序**执行**
    if $T_{c_i} \in$已处理**则**
      继续 ▷ 已作为素数发现
    ($\mathscr{P}_i$, $tess_i$) $\leftarrow$ 提取素数(T_{c_i}) ▷算法 3
    已处理 $\leftarrow$ 已处理 $\cup \mathscr{P}_i$
    ▷ 策略 A：累积（第 1 级，然后第 1-2 级，然后第 1-2-3 级，.....）
    逐步合并各级的素数
    解法$_{cum}$ $\leftarrow$ 针对每种组合求解谜题，并保留最佳结果
    ▷ 策略 B：逐级处理（仅第 1 级，仅第 2 级，仅第 3 级，.....）
    为每个级别独立求解谜题
    解法$_{level}$ $\leftarrow$分别为每个级别求解谜题
    $\mathscr{S} \leftarrow \mathscr{S} \cup \{$（$T_{c_i}$，解法$_{cum}$，解法$_{level}$）$\}$
  返回 $\mathscr{S}$

算法 4 通过过滤、分层提取和记忆化来协调双策略谜题求解。复合体按面积降序排列，以实现分层筛选：如果某个复合体此前已被发现是更大复合体的素数（记录在“handled”集合中），则会完全跳过。对于每个经过处理的复合体，通过算法 3 提取素数。随后采用两种策略求解铺砖谜题：**累积策略** 逐步组合第 1 级、第 1–2 级、第 1–2–3 级等中的素数，并保留瓷砖放置数量最少的解法，即全局最优解。**逐层策略**则针对每个级别独立求解，从而展现粒度权衡：第 1 级使用较少的大型定制瓷砖，而更深层级使用更多的小型标准瓷砖。两种策略都采用带提前终止和去重的回溯，以确保仅保留唯一的最优解。

图 6展示了广度优先搜索提取过程中发现的素数按层级组织的情况。素数按其发现深度分组：第 1 级包含通过对复合体直接剪枝得到的 4 个图块，第 2 级包含通过对第 1 级图块进行剪枝得到的 7 个图块，第 3 级包含 1 个图块，第 4 级则包含通过更深层探索得到的 7 个图块。这种组织方式对于双策略MDL组装至关重要：累积策略逐步合并各层（1、1-2、1-2-3、1-2-3-4），而逐层策略则独立评估每一层。

4 实验结果

4.1 测试用例

我们在三个测试用例上评估算法，这些用例旨在展现不同的镶嵌属性和复杂度级别。(a)**空白带** (6×6)：
$\begin{bmatrix} - & - & - & - & - & - \\ - & 2 & 2 & 2 & 2 & - \\ - & 1 & 1 & 1 & 1 & - \\ - & 2 & 2 & 2 & 2 & - \\ - & - & - & - & - & - \end{bmatrix}$，一种周围带有空白边框的重复 2×2 图案，用于测试裁剪检查。(b) **混合噪声** (6×8)：
$\begin{bmatrix} - & - & - & 5 & 5 & 5 & - & - \\ - & 1 & 2 & - & - & - & - & - \\ - & 3 & - & 4 & - & 4 & - & - \\ - & 1 & 2 & - & - & - & - & - \\ - & 3 & - & 4 & - & - & - & - \\ - & - & - & 5 & 5 & 5 & - & - \end{bmatrix}$，包含不规则图案和散布噪声，用于测试广度优先搜索剪枝的鲁棒性。(c) **简单模式** (4×4)：
$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 2 \\ 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 2 \end{bmatrix}$，一种完美对称的棋盘格，用于测试

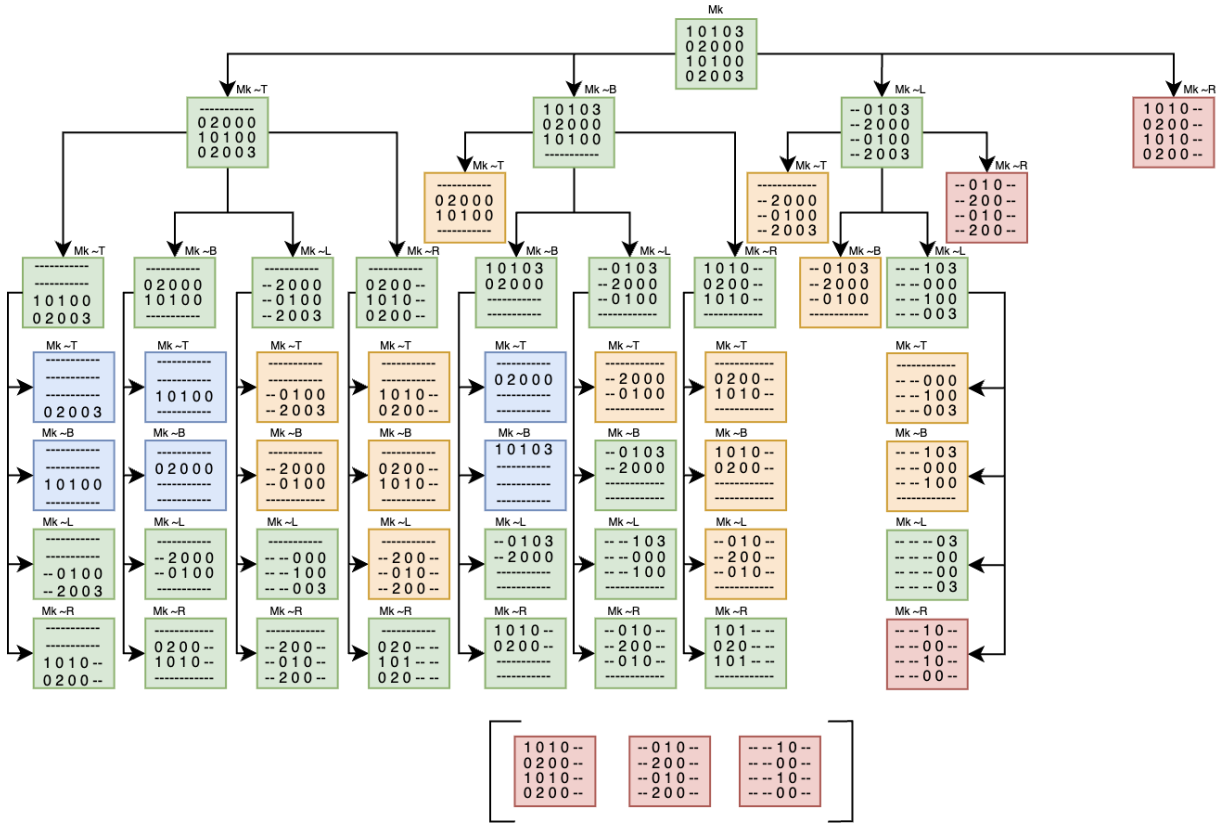


图3. 用于发现复合体的广度优先搜索剪枝树，按颜色显示节点状态：绿色（继续）、粉红色/红色（找到镶嵌）、橙色（重复项）、浅蓝色（近似最小值）、深红色（空项）、紫色（素数）。底部发现了三个复合体。

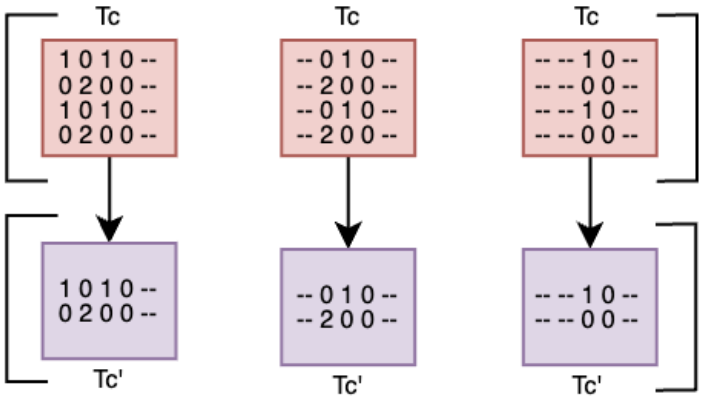


图4. 归一化过程通过迭代进行行列折半，并伴随复制，将三个复合体（Tc，粉红色）缩减为最简形式（Tc'，紫色）。

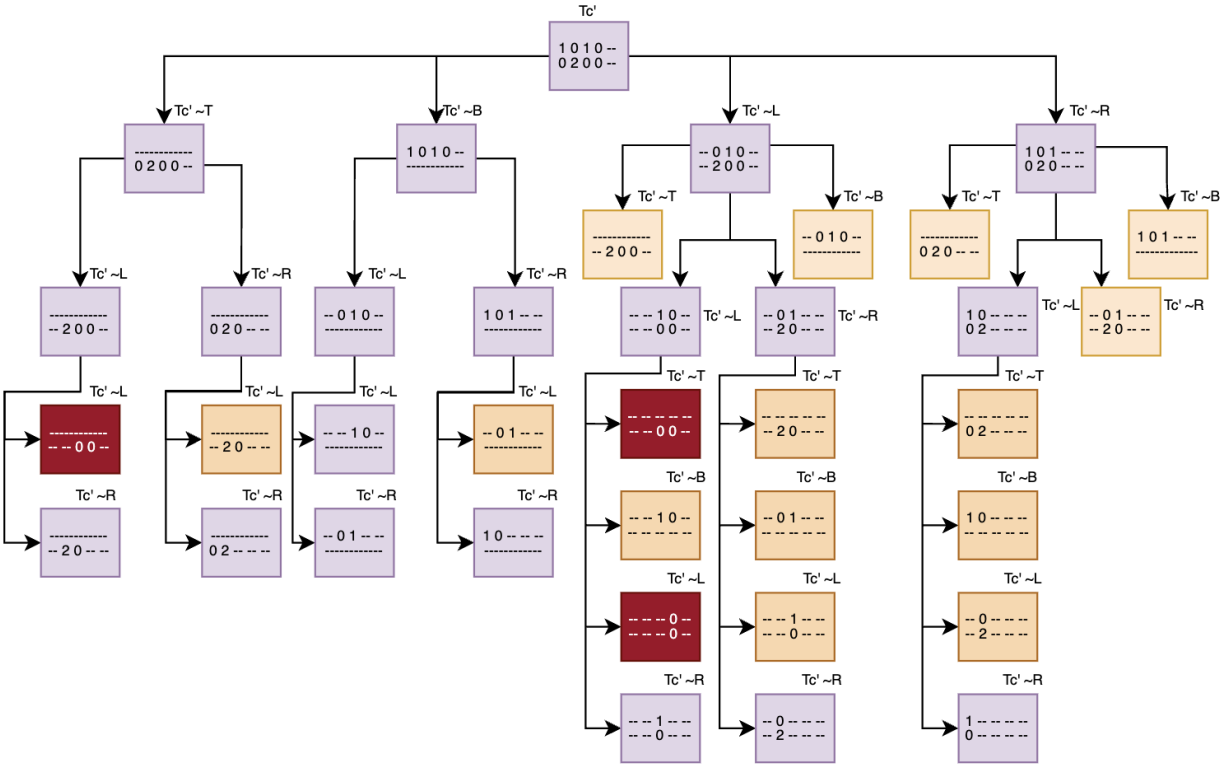


图5. 带重叠检测和归一化的素项提取BFS树。颜色表示：紫色（素数）、橙色（重复项）、深红色（空项）。底部面板显示按层级组织（第1至第4层）。

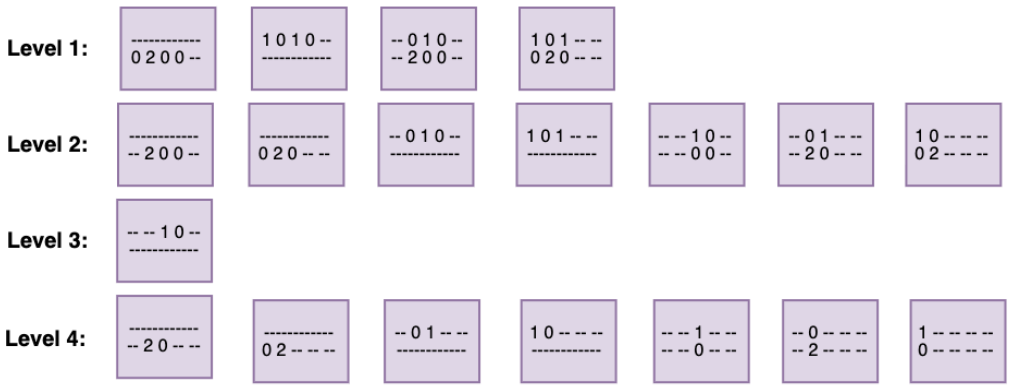


图 6. 按BFS发现层级组织的素数，用于双策略MDL组装。

正则镶嵌检测。

表1 展示了三个测试用例的时间分解。复合体发现（1.46–3.77ms）得益于通过检查或激进分支剪枝实现的提前终止。归一化（0.03–0.42ms）几乎瞬间完成。由于采用双策略谜题求解，素项提取（11.49– 167.20ms）占据了大部分运行时间。分层过滤效果显著：在混合噪声用例中，16 个复合体中有 13 个（81.25%）被跳过，从而 5.3×减少了最耗时阶段的开销。

表1。 时间分析（毫秒）

测试用例	Grid Size	复合体 已找到 / 已处理	发现	归一化	素数 + 谜题 正在求解	总计
空白带	6 × 6	1 / 1	0.17	0.04	0.16	0.64
混合噪声	6 × 8	14 / 4	3.91	0.17	2.90	8.43
简单模式	4 × 4	1 / 1	0.06	0.03	0.99	1.48

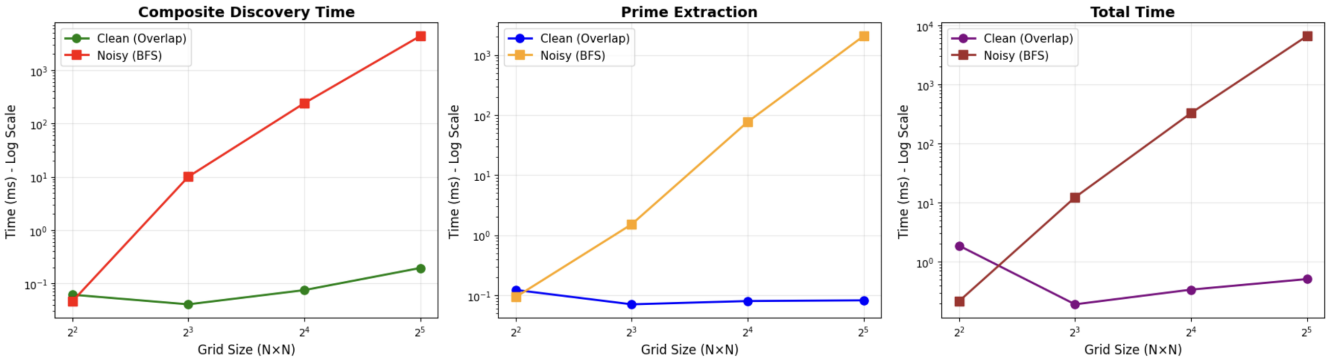


图7. 复合体发现、素项提取及总耗时的时间复杂度

4.2 分解结果

我们在三张表中展示全部三个测试用例的完整分解结果。表2 概述了统计数据，包括发现的复合体数量、需要处理的数量与通过分层筛选跳过的数量、提取的素数总数，以及找到的最优解数量。表3 详细列出每个规范化复合体，以及从中提取的所有素数，并按发现它们时所处的广度优先搜索层级进行组织。表4展示了通过累积策略和逐层策略找到的实际最优解，指出每个解中使用的具体图块以及所需的瓷砖放置数量。

表2. 分解统计摘要

测试用例	复合体 已找到 / 已处理 / 已跳过	素数 已找到	最优解 （累积策略）
空白带	1 / 1 / 0	1	1（1 步）
混合噪声	14 / 4 / 10	20	3（2 步）
简单模式	1 / 1 / 0	4	2（2 步）

“已发现”列显示在广度优先搜索剪枝阶段发现的复合体总数。“已处理”表示这些复合体中实际经过素项提取和谜题求解的数量。“已跳过”显示通过分层筛选排除的复合体数量，因为它们已作为更大复合体的素数被发现。例如，在混合噪声中，共发现 14 个复合体，但只有 4 个需要处理——其余 10 个（71.4%）被跳过，这表明分层筛选十分有效。

此表展示了发现的每个复合体的规范化形式，以及从中提取的完整素数集合。对于空白带，规范化复合体 1/2 (2×1) 已达到最小尺寸，因此既是复合体，也是唯一的素数。对于混合噪声，合数 #1 (2×4) 在多个广度优先搜索层级中产生了 19 个素数。复合体 #7 产生了 1 个素数。复合体 #12 和 #14 已达到最小尺寸 (2×1)，无法进一步分解，因此产生了 0 个素数。其余

表3. 复合项与提取出的素项

测试用例	复合体 ID	Size	素数图块（按广度优先搜索层级组织）
空白带	#1	2×1	素数就是复合体本身：1/2 (2×1)
混合噪声	#1	2×4	19 个素数，分布在多个层级，包括 12- (1×4)、3-4- (1×4)、12 (1×2)、1/3 (2×1) 等
	#7	3×1	1 个素数：-/Q (2×1)
	#2–#6、#8–#11、#13 #12、#14	各种 2×1	已跳过（已作为复合体 #1 的素数发现） 已处理，但未得出任何素数（最小尺寸）
简单模式	#1	2×2	**第 1 级：** 10, 02 (1×2), 1/0, 0/2 (2×1)

10 个复合体（#2–#6、#8–#11、#13）被跳过，因为它们在处理合数 #1 时已被识别为素数，这体现了层次化过滤。

表4. 累积策略与逐级策略得出的最优解

测试用例	策略	解法（需要瓷砖放置）
空白带	累积 第 1 级	解法使用复合体本身：1/2（1 次放置） 与累积相同
混合噪声	累积 第 1 级 第 2 级	3 种解法，每种都需要 2 个放置位置：(1) 12- 和 3-4-；(2) 12/3- 和 -/4-；(3) 1/3 和 2-/-4- 1 种使用大瓷砖的解法（2 次放置） 1 种使用中型拼片的解法（2 次放置）
简单模式	累积 第 1 级	2 个解，每个解均需要放置 2 次：（1）10 和 0/2；（2）1/0 和 02 与累积策略相同

此表展示了两种策略发现的最优铺砖解。累积策略通过组合多个广度优先搜索层级中的牌块来寻找解，并报告所需瓷砖放置次数最少的解。逐层策略则针对每个层级独立求解谜题。对于空白带，规范化复合体已是最小形式，可通过一次放置构成完整解。对于混合噪声，合数 #1 有 3 个最优解，每个解均需放置 2 次瓷砖；第 1 级和第 2 级各找到 1 个解。括号中的数字表示每个解所需的瓷砖放置总次数。

4.3 可扩展性分析

图 8 评估了算法在以下范围内的可扩展性：从 2×2 到 32×32 网格，使用两种测试模式：重叠模式（通过全网格重复测试检查）和剪枝模式（在噪声中嵌入小型复合体以测试广度优先搜索）。所有坐标轴均采用对数刻度。

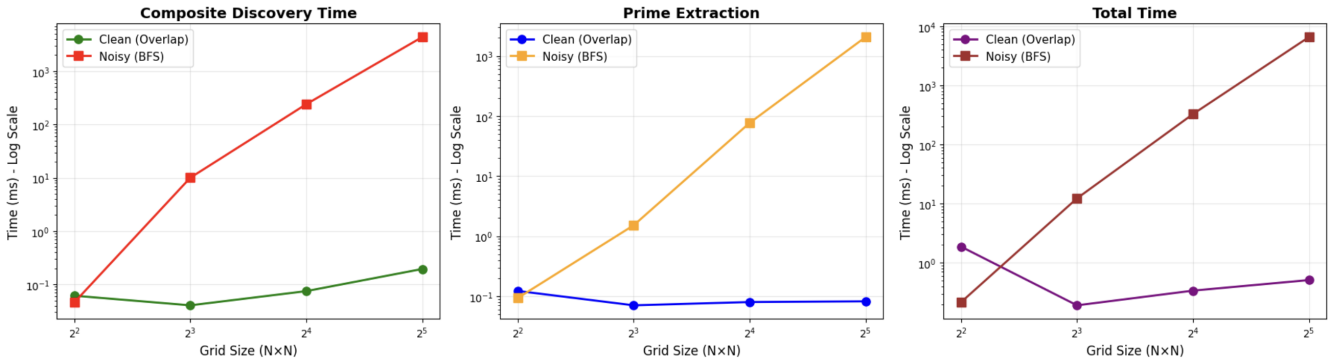


图 8. 可扩展性分析　　采用双对数刻度：重叠模式保持不变，而剪枝模式呈指数增长　　.

4.4 讨论

双策略方法提供了互补的见解。累积策略可找到全局 MDL，非常适用于压缩场景。逐层策略揭示了粒度权衡：第 1 级使用较少的定制瓷砖（组装量最少，定制

制造），而更深层级则使用更多标准单元（模块化部件，复杂装配）。分层筛选至关重要，在混合噪声下将计算量减少了 5.3×，同时保持完整性。选择性复制策略兼顾了完整性（在检查/归一化中至关重要，因此会采用）和效率（在广度优先搜索中则省略，因为穷尽式探索已足够）。嵌套镶嵌检测可确保规范表示，并识别复用机会。

局限性包括仅适用于矩形瓦片和曼哈顿几何，以及谜题求解器在最坏情况下具有指数级复杂度。未来的工作可以探索启发式剪枝、其他求解范式（整数线性规划、可满足性问题），以及用于处理真实世界噪声图像的近似镶嵌。

5 相关工作

经典与现代铺砌理论为我们的研究提供了概念背景。从准晶体中的彭罗斯铺砌，到近期的单瓦片以及对周期铺砌猜想的严格反例，这些非周期现象表明，并非所有镶嵌都存在有限的平移生成器；参见关于准晶体中彭罗斯现象的研究[5], 非周期瓦片机器[7], 近期发现的非周期单块砖[6], 以及对周期铺砌猜想的反例[3]。这些研究促使我们明确将范围限定为精确、轴对齐的矩形生成器，并阐明了为何我们的 **O**类（范围之外）包含非周期和非矩形情形。关于数学与设计中镶嵌的介绍性及应用导向论述，进一步为这一领域提供了背景[2,4]。

在算法方面，关于铺砌与图像语言的判定和解析问题已有大量研究。针对特定族（例如正方形铺砌和多米诺骨牌装箱）的多项式时间结果表明，范围精细、精确的形式化具有重要价值[1]。面向由铺砌文法指定的图像的基于 SAT 的解析器与补全器，展示了一种互补的、面向二维模式规范与识别的约束求解方法[11]。在应用识别中，纺织品与瓦片图案的结构描述[10] 以及用于场景解析的分层空间铺砌模型[8] 在图像层面处理周期性规律，而分块卷积架构[9] 则将铺砌用作架构偏置，而非推理对象。我们的贡献在目标与接口上有所不同：我们针对符号化网格对齐数组，并提供确定性两层过程（先进行复合体发现，再进行素因子分解），在严格相等与极小性条件下返回显式的生成瓦片。

第二条相关工作线索来自基于网格的推理与程序归纳基准，尤其是抽象与推理语料库（ARC）。近期的技术报告与分析[17] 探讨了用于求解定义在小型网格上的离散推理任务的系统设计、评估与学习策略。我们的方法可被视为一个可解释的非参数模块，适用于可归结为在精确相等条件下识别周期性（可能分层）结构的类 ARC 任务子集。更一般地说，关于现代系统能力与适当基准测试的讨论（例如[19]）凸显了在明确定义的问题类别上具有可保证行为的透明过程的价值。

总而言之，尽管先前研究涵盖了非周期铺砌[3]、[5–7]、精确算法片段和文法／SAT 形式化[1,8,10,11]、铺砌的架构用途[9], 以及网格推理基准测试[20], ，我们的工作仍提出了一套专注且确定性的流程，用于在离散平面网格上精确识别周期性部分镶嵌，并提供了明确说明适用范围和局限性的可解性分类。

6 结论

我们提出了一种分层方法，用于发现离散网格中的精确周期结构。我们的方法能够自动识别重复模式，将其化简为最简形式，并将其分解为原子构件——整个过程采用确定性流程，且在定义明确的模式类别上具有可证明的行为。

该方法的核心贡献有三点。第一，提出了一种选择性复制策略，在检查和归一化过程中处理奇维模式，同时通过自然探索偶维后代，在穷举搜索中保持高效。第二，采用分层筛选，通过识别较小复合体已作为较大复合体的组成部分被发现，从而跳过冗余计算；在我们的实验中，这使处理开销降低了81%。第三，采用双重分解策略，既能揭示全局最优解，也能展现定制切割图块与标准模块单元之间的粒度权衡。

我们的实验评估表明，该方法能够清晰地解析从简单规则结构（通过快速检查即可解析）到复杂不规则图案（需要进行完整的广度优先搜索探索）的各类图案。已发现图块的按层级组织方式，支持对不同尺度下的分解方案进行系统分析，并对制造、组装和材料再利用具有实际意义。

我们明确说明了方法的适用范围：该方法假设精确相等、轴对齐周期性和矩形生成器。涉及近似、旋转或非周期结构的模式，按设计不在适用范围之内。这些限制确保了方法具有确定性和可证明的正确性。未来工作可以探索针对含噪模式的容差匹配、针对旋转结构的几何预对齐，以及非矩形生成器；这些都是有原则的扩展方向，且在确定性与通用性之间存在明确的权衡。

我们的贡献是为精确周期结构识别提供了一个精确、高效的基线，并明确界定了适用范围。我们希望这项工作既能成为实用的分析工具，也能为系统性地扩展至精确、轴对齐情形之外奠定基础。

参考文献

1. Aamand, A., Abrahamsen, M., Ahle, T. D. 与 Rasmussen, P. M. R. 正方形铺砌与多米诺骨牌装填的多项式时间算法。ACM算法汇刊 19, DOI: 10.1145/3597932 (2023)。2. Chang, W. 镶嵌在建筑几何设计中的应用。载于E3S会议网络，第38卷，03015（EDP Sciences，2018）。3. Greenfeld, R. 与 Tao, T. 周期铺砌猜想的一个反例。《数学年刊》200, 301–363 (2024)。4. Ždímalová, M. 镶嵌。《几何面：从Agnesi到Mirzakhani》225–234 (2020)。5. Bursill, L. 与 Ju Lin, P. 在准晶体中观测到的彭罗斯铺砌。《自然》316, 50–51 (1985)。6. Smith, D., Myers, J. S., Kaplan, C. S. 与 Goodman-Strauss, C. 一种非周期单块砖。arXiv预印本 arXiv:2303.10798 (2023)。7. Cáceres, J. 与 Márquez, A. 一种非周期铺砌机。《计算几何》23, 171–182 (2002)。8. Wang, S., Wang, Y. 与 Zhu, S.-C. 面向场景建模、解析与属性标注的分层空间铺砌学习。《IEEE模式分析与机器智能汇刊》37, 2478–2491, DOI: 10.1109/TPAMI.2015.2424880 (2015)。9. Ngiam, J. 等。平铺卷积神经网络。《神经信息处理系统进展》23 (2010)。10. Valiente, J. M., Albert, F., Carretero, C. 与 Gomis, J. M. 利用图像处理对纺织品与图块图案设计进行结构描述。载于第17届国际模式识别会议论文集，2004年，ICPR 2004，第1卷，498–503（IEEE，2004）。11. Pradella, M. 与 Reghizzi, S. C. 面向由铺砌指定图像的基于SAT的解析器与补全器。《模式识别》41, 555–566 (2008)。12. Grünbaum, B. 与 Shephard, G. C.《铺砌与图案》（Courier Dover Publications，1987）。13. Phillips, D. 镶嵌。《Wiley跨学科评论：计算统计学》6, 202–209 (2014)。14. Zhou, X. 等。分层重叠铺砌。载于第十届国际代码生成与优化研讨会论文集，207–218 (2012)。15. Hajiaghayi, M. T., Kleinberg, R. D., Räcke, H. 与 Leighton, T. 节点容量受限有向图上的无感知路由。ACM算法汇刊 3, 51–es, DOI: 10.1145/1290672.1290688 (2007)。16. Korman, A. 与 Peleg, D. 低局部密度图的动态路由方案。ACM算法汇刊 4, DOI: 10.1145/1383369.1383372 (2008)。17. Chollet, F., Knoop, M., Kamradt, G. 与 Landers, B. ARC Prize 2024：技术报告。arXiv预印本 arXiv:2412.04604 (2024)。18. LeGris, S., Vong, W. K., Lake, B. M. 与 Gureckis, T. M. H-ARC：抽象与推理语料库基准测试上人类表现的稳健估计。arXiv预印本 arXiv:2409.01374 (2024)。19. Pfister, R. 与 Jud, H. 理解人工智能并对其进行基准测试：OpenAI的o3并非通用人工智能。arXiv预印本 arXiv:2501.07458 (2025)。20. Zhu, S., Geng, S. 与 Chen, U. L. T5-ARC：ARC-AGI 挑战中传导式Transformer模型的测试时训练。OpenReview预印本 https://openreview.net/forum?id=TtGONY7UKy 。21. Ouellette, S. 面向ARC-AGI的高效神经引导程序归纳。arXiv预印本 arXiv:2411.17708 (2024)。22. Mernik, M., Heering, J. 与 Sloane, A. M. 何时以及如何开发领域专用语言。《ACM计算调查》（CSUR）37, 316–344 (2005)。23. Franzen, D., Disselhoff, J. 与 Hartmann, D. LLM架构师：求解ARC-AGI是一个视角问题。预印本 GitHub.io 。