arXiv:2304.12642v1 [cs.DM] 25 Apr 2023

一种实用的图最大范数最优二元标记算法

Filip Malmberg[1],Alexandre X. Falcão[2]

[1] 图像分析中心，信息技术系，乌普萨拉大学
瑞典 filip.malmberg@it.uu.se[2] 计算机研究所，坎皮纳斯大学，巴西 afalcao@ic.unicamp.br

摘要。 本文探讨了一种最优二元标记方法的高效实现，该方法最初由 Malmberg 和 Ciesielski（2020）提出，用于对图顶点进行标记。该方法能够在关于图规模的二次时间内找到一个使基于 L_∞-范数的目标函数达到全局最小的标记方案。该方法支持对一类新型优化问题进行全局优化，这些问题在图像处理和计算机视觉等应用领域具有重要意义。在最初的表述中，Malmberg-Ciesielski算法的计算开销非常高，限制了其在实际应用中的实用性。本文提出了该算法的一个修改版本，通过利用原方法中的冗余来减少计算时间。尽管所提出方法的理论渐近时间复杂度相同，但我们证明了它在实践中显著更加高效。即使对于小规模问题，我们也观察到了 4 至 5 个数量级的加速比。计算时间的降低使 Malmberg-Ciesielski 方法成为许多实际应用中的可行选择。

关键词： 图标记 · 组合优化 · 字典序最大排序。

1 引言

计算机科学和模式识别中的许多问题，可以表述为寻找图的顶点标记，使该标记能够优化某个由具体应用驱动的目标函数。在近期的研究中，Malmberg 和 Ciesielski [9]提出了一种二次时间算法，为图的顶点分配二元标签，使所得标记根据一个基于最大范数（即 L_∞ 范数）的目标函数达到最优。本文探讨 Malmberg 和 Ciesielski 所提出算法的高效实现。我们给出该算法的一个版本，它在渐近时间复杂度上仍为二次，但在实践中速度快了几个数量级。

Malmberg-Ciesielski算法的一个关键部分是求解一系列布尔2可满足性 （2-SAT）问题。Malmberg 和 Ciesielski 指出，

每个这样的2-SAT问题都可以使用例如Aspvall算法 [1] 在线性时间内求解。不过，他们也指出，序列中每一对相邻的2-SAT问题都高度相似，因此孤立地求解每个2-SAT问题似乎效率低下。在此，我们展示了如何利用后续2-SAT问题之间的这种冗余，构造出效率显著更高的算法版本。

2 背景与动机

我们考虑这样一个问题：为一组由索引 1、.. . 、 n标识的变量分配二元标签（0 或 1）。一个典型问题是寻找一个二元标记ℓ： $[1, n] \rightarrow \{0, 1\}$，使其最小化如下形式的目标函数

$$E_p(\ell) := \sum_i \phi_i^p(\ell(i)) + \sum_{(i,j)\in\mathcal{N}} \phi_{ij}^p(\ell(i), \ell(j)), \tag{1}$$

其中， $\ell(i) \in \{0, 1\}$ 表示变量 i 的标签，而 $\mathcal{N}$ 是一组变量对，这些变量对被视为相邻。

函数 $\phi_i(\cdot)$ 被称为一元 项。每个一元项仅取决于单个二元变量的值，用于表示为每个特定标签分配给单个变量的偏好。

函数 $\phi_{ij}(\cdot, \cdot)$ 被称为二元 项。每个二元项同时取决于分配给两个变量的标签，因此在分配给这些变量的标签之间引入了依赖关系。通常，这种变量间的依赖关系用于表达期望的解应具有一定程度的平滑性或规则性。

正如 Kolmogorov 和 Zabih 所确立的 [7]，，在所有成对项均为子模的条件下，上述标记问题可以求得全局最优解；在此处所采用的形式中，这意味着它们必须满足不等式

$$\phi_{ij}^p(0, 0) + \phi_{ij}^p(1, 1) \leq \phi_{ij}^p(0, 1) + \phi_{ij}^p(1, 0). \tag{2}$$

如果问题包含非子模二元项，那么在一般情况下，寻找全局最优标记已知是 NP-hard 的 [7]。因此，想要解决此类优化问题的实践者必须首先验证其局部代价泛函是否满足适当的子模条件。如果不满足，他们就必须诉诸近似优化方法，而这些方法对于给定的问题实例而言未必能够产生令人满意的结果 [6]。然而，最近 Malmberg 和 Ciesielski [9] 证明，在极限情况下，随着 p 趋于无穷，子模性的要求将不复存在！为了刻画当 p 趋于无穷时使 1 最小化的标记，我们首先注意到，随着 p 趋于无穷，目标函数 E_p 自身收敛于

$$E_\infty(\ell) := \max\big\{\max_i \phi_i(\ell(i)),\ \max_{(i,j)\in\mathcal{N}} \phi_{ij}(\ell(i), \ell(j))\big\}. \tag{3}$$

也就是说，目标函数变成了包含所有一元项和成对项的向量的最大范数。刻画该解的一种更精细方式是字典序最大排序（Lex-MO）框架 [3,4,5]。Levi 和 Zorin 也研究了相同的概念，并使用了严格极小化解 [8]这一术语。在该框架中，通过将所有元素（在我们的情形中，即给定解的所有一元项和成对项的值）按非递增顺序排列，然后进行字典序比较，来比较两个解。这样可以避免 E_∞ 目标函数的潜在缺陷，即它无法区分最大误差以下具有高误差或低误差的解。Malmberg-Ciesielski 算法 [9] 能够在多项式时间内计算出一个全局最小化 E_∞的标记，即使存在非子模成对项也是如此。在满足某些条件时，该算法还保证能够生成一个在 Lex-MO 意义下最优的解。

3 预备知识

本节简要回顾 Malmberg-Ciesielski算法，并介绍一些概念，以便在第 4 节阐述我们提出的该算法的高效实现。

3.1 布尔2可满足性

我们先回顾布尔2可满足性（$2\text{-}SAT$）问题。给定一组布尔变量 $\{x_1, \ldots, x_n\}$，$x_i \in \{0, 1\}$ 以及关于这些变量对的一组逻辑约束，2-SAT问题要求确定是否可以为这些变量赋值，使所有约束都得到满足（若存在这样的赋值，也要找出该赋值）。为正式定义2-SAT问题，我们称一个文字 为布尔变量 x 或其否定$\neg x$。因此，2-SAT问题可以用一个布尔表达式来定义，该表达式是若干子句的合取，其中每个子句都是两个文字的析取。这种形式的表达式称为2-CNF公式，其中CNF代表合取范式。2-SAT问题要求确定：对于给定的2-CNF公式，是否存在一个针对其中变量的真值赋值，使整个公式为真。如果存在这样的赋值，则称该2-SAT问题可满足；否则称其不可满足。例如，以下表达式是一个涉及三个变量 x_1， x_2， x_3和两个子句的2-CNF公式：

$$(x_1 \vee x_2) \wedge (x_2 \vee \neg x_3) \tag{4}$$

例如，将三个变量的值都赋为 1（或真 ），这个公式的计算结果为真。因此，这个 2-CNF 公式所表示的 2-SAT 问题是可满足的。

对于任意 2-CNF 公式，2-SAT 问题都可以在线性时间内求解，时间复杂度相对于子句数量而言[3]，例如使用 Aspvall算法 [1]。

现在我们介绍一些与 2-SAT 问题相关、且本文论述所需的进一步概念，约定 x_i 和 $\neg x_i$ 表示文字，而 v_i 表示其真值未知的文字，$\bar{v_i}$ 表示其互补文字。

2-CNF公式中的每个子句（$v_i \vee v_j$）在逻辑上等价于从其中一个变量指向另一个变量的蕴含：

$$(v_i \vee v_j) \equiv (\bar{v_i} \Rightarrow v_j) \equiv (\bar{v_j} \Rightarrow v_i)\ . \tag{5}$$

正如Aspvall等人所确立的 [1], ，这意味着每个2-SAT问题 F都可以与一个蕴含图 $G_F = （V，E）$相关联，这是一个具有顶点 V 和边 E 的有向图，构造如下：

1. 对于每个变量 x_i，我们向 G_F 添加两个名为 x_i 和 $\neg x_i$ 的顶点。这些顶点 x_i 和 $\neg x_i$ 被称为互补。 2. 对于 F 的每个子句 $(v_i \vee v_j)$，我们向 G_F 添加边 $(\bar{v_i}, v_j)$ 和 $(\bar{v_i}, v_j)$。

因此，蕴含图中的每个顶点都可以唯一地对应于一个文字，而每条边都对应于从一个文字到另一个文字的蕴含。因此，我们有时会用相应的文字 v_i 来指代蕴含图中的顶点。对于给定的真值赋值，如果蕴含图中的某个顶点对应的文字在该赋值下的取值为真，我们就说该顶点与该赋值相符。蕴含图 G_F 是斜对称的，也就是说，若 (v_i, v_j) 是 G_F 中的一条边，则 $(\bar{v_i}, \bar{v_j})$ 也是 G_F 中的一条边。由此可见，对于 G_F 中的每条路径 $\pi = (v_1, v_2, ..., v_k)$，路径 $\bar{\pi} = (\bar{v_k}, \bar{v}_{k-1}, ..., \bar{v}_1)$ 也属于 G_F。

在证明所提算法正确性的过程中，我们将依赖以下由 Aspvall 等人 [1] 得出的性质：

性质 1 给定的真值赋值满足公式 F，当且仅当在 G_F 中不存在这样的顶点：其对应文字与该赋值一致，且具有一条出边指向与该赋值不一致的顶点。

3.2 Malmberg-Ciesielski算法

关于 Malmberg-Ciesieleski 算法的完整描述，请参阅原始出版物（[9], 算法 1）。这里我们重点介绍该算法的一个关键方面，即求解一系列 2-SAT问题。在此步骤中，我们将需要标记的变量确定为 2-SAT问题中涉及的布尔变量。布尔变量的真值赋值 T 自然地转换为标记 ℓ。对于算法的这一步骤，我们得到一个子句的有序序列 $\mathcal{C}$，其顺序由公式 3 中一元项和二元项导出的优先级决定。简而言之，算法的运行方式如下：

[3] 这与一般的布尔可满足性问题形成对比，在一般的布尔可满足性问题中，子句可以包含多于两个文字。即使是3SAT问题（其中每个子句最多可包含三个文字）也是NP-hard问题。

– 将 F 初始化为空的2-CNF公式，不包含任何子句。– 按顺序处理 $\mathcal{C}$中的每个子句 c：

- 如果 $F \wedge c$ 可满足，则设置 $F \leftarrow F \wedge c$。

在上述算法的所有步骤中，公式 F 始终保持可满足。在算法终止时，公式 F 定义了唯一的真值赋值T，因此也定义了一个标记 ℓ。对于由 Malmberg and Ciesieleski 定义的子句特定序列 $\mathcal{C}$，所得标记保证能够全局最小化公式 3 中的目标函数。

在每次迭代中，我们需要确定 $F \wedge c$ 是否可满足，即求解与公式 $F \wedge c$相关的 2-SAT问题。Malmberg and Ciesieleski 建议为此使用 Aspvall算法，其渐近时间复杂度为 $\mathcal{O}(|F|) \leq \mathcal{O}(|\mathcal{C}|)$。令 $N = n + |\mathcal{N}|$ 表示公式 3 中一元项和二元项的总数。根据其设计，序列 $\mathcal{C}$中的子句数为 $\mathcal{O}(N)$，因此，使用 Aspvall算法实现的 Malmberg-Ciesieleski 算法的渐近时间复杂度为 $\mathcal{O}(N^2)$。

4 所提算法

如上一节所述，Malmberg-Ciesieleski 算法会迭代地构建一个公式 F，并确保该公式在算法的每一步都保持可满足。我们提高计算效率的方法是在算法的每一步维护一个满足当前公式F的真值赋值。当尝试判断序列 C 中的下一个子句 c 能否添加到 F 中而不使公式变为不可满足时，我们将说明如何利用这个已有的真值赋值来减少计算时间。我们将 2-SAT 问题的布尔变量的真值赋值 T 表示为函数 $T\colon\ [1,n] \to \{0,1\}$，使得 $T(i)$是赋给变量 x_i的值。显然，如果 T 满足 c，那么它也满足 $F \wedge c$，因此我们关注T 不满足下一个子句 c的情形。

我们将考虑带假设 [2], 的 2-SAT 求解；也就是说，给定一个可满足公式，我们要判断：如果假定变量子集取给定值，同一公式是否仍然可满足？此类假设将用蕴含图中的一组顶点表示——由于每个顶点都对应一个文字，这组顶点就对应一组都被假定为真的文字。我们假设此处使用的顶点集内部不存在冲突，即不同时包含某个顶点及其补顶点。

下面我们将介绍一种高效算法，用于在假设集 A 下求解 2-SAT问题，其中给定的真值赋值 T 满足公式（不含这些假设。为了解这种过程如何帮助我们高效实现 Malmberg-Ciesielski算法，我们注意到，根据德摩根定律，子句 ($v_i \vee v_j$) 可以重写为 $\neg(\bar{v_i} \wedge \bar{v_j})$。以这种形式，更容易看出，为了满足该子句，真值赋值 T 必须恰好满足表达式 $(v_i \wedge v_j)$、$(v_i \wedge \bar{v_j})$ 或 $(\bar{v_i} \wedge v_j)$ 中的一个。每个表达式都代表一个假设集，因此 $F \wedge (v_i \vee v_j)$ 是可满足的，当且仅当

算法 1： CheckSolvable(G,C,T)

输入： 表示一个 2-SAT问题的蕴含图 G 。一个子句 $c = (v_i) \vee (v_j)$。满足公式 F的真值赋值 T 由 G编码。

结果： 指示 $F \wedge c$ 是否可满足的真值。如果可满足，则 T 是一个满足 $F \wedge c$ 和 G 的真值赋值编码了 $F \wedge c$。否则， T 且 G 保持不变。

```
令 可满足 ←为假
如果 T 满足 c则
    令 可满足 ←为真
否则
    若 v_i = v_j 则
        若 SolveWithAssumptions(G,{v_i},T) 则
            设 可满足 ←为真
    else
        /* v_i ≠ v_j                                          */
        若 SolveWithAssumptions(G,{v_i, v_j},T) 则
            设 可满足 ←为真
        否则如果 SolveWithAssumptions(G,{v̄_i, v_j},T) 则
            将 satisfiable ←设为真
        否则如果 SolveWithAssumptions(G,{v_i, v̄_j},T) 则
            将 satisfiable ←设为真
如果 satisfiable则
    将边 (v̄_i, v_j) 和 (v̄_j, v_i) 添加到 G
返回可满足
```

仅当 F 在以下假设集 A中的一个下可满足： $\{v_i, v_j\}$、$\{v_i, \bar{v_j}\}$ 或 $\{\bar{v_i}, v_j\}$。我们还注意到，在 $i = j$这一特殊情况下，上述论证可以进一步简化。在此情况下，公式化简为 $F \wedge (v_i)$，这等价于在假设 $A = \{v_i\}$下求解 F。

算法 1 中列出的过程利用这一结果执行 Malmberg-Ciesieleski 算法的内循环：它确定给定子句是否可以添加到可满足公式中而不会使其变为不可满足。若可以，则更新表示该公式的蕴含图，将新子句纳入其中。算法 1 使用一个名为 $SolveWithAssumptions$ 的过程，下面将对其进行描述。

设 F 为一个公式，其对应的蕴含图为 $G_F = (V, E)$；设 T 为与 F 相关联的变量的真值赋值，并设 A 为一个假设集。我们将 $R_{A,T} \subseteq V$ 定义为这样一个顶点集：其中的顶点可从 A 中的任一顶点在 G_F 中到达，且无需经过从与 T 一致的顶点出射的边。使我们所提算法得以实现的主要理论结果总结如下：

算法2： SolveWithAssumptions(G,A,T)

输入： 表示2-SAT问题的蕴含图 G 。一组
不存在内部冲突的假设 A。满足由 T 编码的公式的真值赋值 T
满足由 G 编码的公式 F 。

结果： 一个真值，表示是否存在满足以下条件的真值赋值 T'
满足由 G 编码的公式 F，同时满足
假设 A。如果算法返回 真，那么 T 就是一个真值
赋值，满足此判据。否则， T 保持不变。

辅助结构： 一个由顶点组成的先进先出（或后进先出）队列 Q ；一个顶点集合 C。

```
设置 C ← ∅
对每个 v ∈ A 执行
    将 v 插入 Q中
    将 v 插入 C中
当 Q 不为空时执行
    从 Q中弹出一个顶点 v
    如果 v 与 T 不一致则
        对于每个具有一条出边指向 w 的顶点 w 执行
            如果 w̄ ∈ C 则
                返回false 并退出
            否则，如果 w ∉ C 则
                将 w 插入 Q中
                将 w 插入 C中
对顶点 v ∈ C 执行
    将与 v 对应的变量的 T 值设置为与 v一致。
返回真
```

定理 1. 假设 F 可满足。设 T 为满足 F 的真值赋值，并设 A 为一个假设集。那么，当且仅当子图 $R_{A,\ T}$ 不包含一对互补顶点时， F 在假设集 A 下可满足。

证明。 证明的第一部分，假设 $R_{A,\ T}$ 确实包含一对互补顶点 v_i 和 $\bar{v_i}$。那么，假设 A 直接意味着 v_i 和 $\bar{v_i}$ 同时可满足，这显然是一个矛盾，因此在假设 A 下， F 不可满足。

证明的第二部分，假设 $R_{A,\ T}$ 不包含任何一对互补顶点。于是，我们可以通过对 $R_{A,\ T}$中的每个顶点，将相应变量设置为相应真值，从给定的真值赋值 T 构造出一个定义良好的真值赋值 T' 。对于任意顶点 $v_i \notin R_{A,\ T}$，我们有 $T(i) = T'(i)$。此外，真值赋值 T'满足 A中的所有假设。

接下来，为了构造反证法，假设上文构造的真值赋值 T'不满足 F。那么，根据性质 1，至少存在一个与 T' 一致的顶点 v_i ，它有一条出边

指向一个与 T'不一致的顶点 v_j 。现在，我们考虑真值赋值 T 相对于对应于 v_i 和v_j 的变量的所有四种可能：

1. 假设 v_i 和 v_j 都与 T一致。由于 v_j 与T' 不一致，因此必有 $\bar{v_j} \in R_{A,T}$，也就是说，存在一条从 A到 $\bar{v_j}$的路径 π，它不会经过从与 T一致的顶点出发的边。根据蕴含图的斜对称性，存在一条从 $\bar{v_j}$ 到 $\bar{v_i}$的出边，因此我们可以将这条边接到路径 π 上，由此可知 $\bar{v_i}$ 也在$R_{A,T}$中，这与 v_i和 T'一致相矛盾。因此，假设 v_i 和 v_j 都与 T 一致会导致矛盾。2. 假设 v_i 与 T 一致，但 v_j 不一致。由于 v_i 有一条出边指向 v_j，这与 T 满足 F相矛盾，因此，假设 v_i 与 T 一致，但 v_j 与 T 不一致，会导致矛盾。3. 假设 v_j 与 T 一致，但 v_i 不一致。于是 v_i 和 $\bar{v_j}$ 都在 $R_{A,T}$中。从 v_i到 v_j有一条出边，且 v_i与 T不一致，因此 v_j 也在 $R_{A,T}$中，这与假设 $R_{A,T}$ 不同时包含一个顶点及其补相矛盾。因此，假设v_j 与 T 一致，但 v_i 与 T 不一致，会导致矛盾。4. 假设 v_i和 v_j都与 T不一致。由于 v_i与 T'一致，因此必有 $v_i \in R_{A,T}$，也就是说，存在一条从 A 到 v_i 的路径 π，它不会经过从与 T一致的顶点出发的边。但由于存在一条从 v_i 到 v_j 的出边，且 v_i 与 T不一致，我们可以将 π 与这条边接上，由此可知 v_j 也必定在 $R_{A,T}$中，这与 v_j 和 T'不一致相矛盾。因此，假设 v_i 和 v_j 都与 T 不一致会导致矛盾

上述四种情况涵盖了真值赋值 T 中与 v_i 和 v_j 对应的变量取值的所有可能配置，并且每种情况都会导致矛盾。由此可知，假设 T' 不满足 F 会导致矛盾，因此 T' 必须满足 F。证明完毕。□

根据上述定理，如果能找到集合 $R_{A,\ T}$，我们就可以在给定假设下求解一个 2-SAT问题。我们注意到，对于给定的假设集，利用广度优先搜索等方法，可以在 $\mathcal{O}(V+E)$ 时间内轻松找到集合 $R_{A,\ T}$。如果在广度优先搜索过程中遇到某个顶点，而其补顶点已确认属于 $R_{A,\ T}$，则可以终止搜索并返回 $false$。算法 2 给出了该方法的伪代码。若求解每个 2-SAT问题的时间上限为 $\mathcal{O}(V+E)$，那么所提出的方法与使用 Aspvall算法的方法具有相同的渐近时间复杂度。不过在实践中，我们将看到集合 $R_{A,\ T}$ 只是蕴含图中的一个很小的子集，因此与在 Malmberg-Ciesielski算法的每次迭代中都运行 Aspvall算法相比，这种方法快得多。

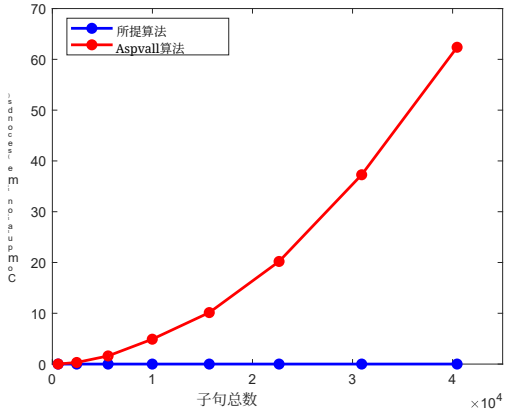

图1. 所提出的 Malmberg-Ciesielski 方法实现与使用 Aspvall算法的原始公式在计算时间方面的比较，其中横坐标为 2-SAT 序列中的子句总数。

5 评估

为了评估我们提出的 Malmberg-Ciesielski算法版本相较于使用 Aspvall算法的原始形式的性能，我们进行了一项实证研究，模拟图像处理和计算机视觉中的典型优化场景。我们对大小为 $W \times H$ 的二维图像中的像素进行二元标记。邻接关系 $\mathcal{N}$ 采用图像处理中使用的标准四连通性定义。一元项和二元项的取值均从均匀分布中随机抽取。随后，我们比较两种实现的计算时间，图像大小范围从 8×8 到 64×64。我们只测量求解这一系列 2-SAT 问题所需的时间，因为这是两种实现之间唯一不同的部分。结果见图 1。如图所示，基于 Aspvall算法的实现，其计算时间会随着问题规模增大而急剧增加。对于大小为 64×64 的图像，基于 Aspvall 算法的实现需要 62 秒，而所提出的实现只需 0.004 秒即可完成相同的计算，加速比超过四个数量级。

为进一步研究所提出的实现相对于问题规模的计算时间，我们针对大小从 128×128 到 4096×4096不等的图像进行了单独实验；对于这些图像，使用 As-pvall 算法的实现会变得慢到难以接受。结果如图 2 所示。从图中可以看出，在这一范围内，问题规模与计算时间之间的经验关系似乎更接近线性函数，而不是最坏情况渐近时间复杂度所暗示的二次关系。

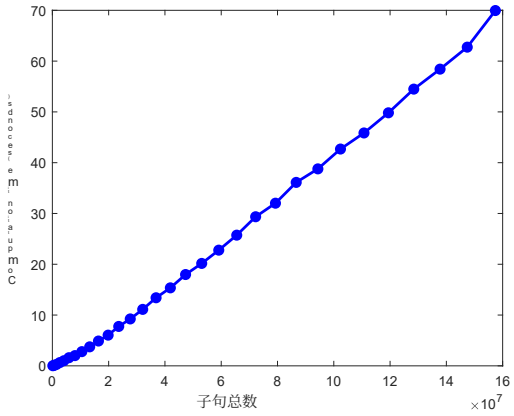

图2。 所提出的实现相对于问题规模的计算时间。

6 结论

我们提出了 Malmberg-Ciesielski 方法的一种改进型高效实现，用于图的最优二元标记。虽然我们所提出的实现与原始算法具有相同的渐近运行时间复杂度，但我们证明，在实际应用中，其速度快了数个数量级。计算时间的这一缩减使 Malmberg-Ciesielski 方法成为许多实际应用中可行的选择。

致谢 本研究获得了圣保罗研究基金会（FAPESP）提供的 SPRINT 资助（ 2019/08759- 2）以及乌普萨拉大学的支持。

参考文献

1. Aspvall, B., Plass, M.F., Tarjan, R.E.: 用于测试某些量化布尔公式真值的线性时间算法。《信息处理快报》**8**(3), 121–123 (1979) 2. Eén, N., Sörensson, N.: 一种可扩展的 SAT 求解器。载于：国际可满足性测试理论与应用会议。第 502–518 页。Springer (2003) 3. Ehrgott, M.: 字典序最大排序——一种多准则组合优化的解概念。德国国家图书馆 (1995) 4. Ehrgott, M.: 字典序最大排序解的刻画 (1999) 5. Ehrgott, M.: 多准则优化，第 491 卷。Springer Science & Business Media (2005) 6. Kolmogorov, V., Rother, C.: 使用图割最小化非次模函数——综述。《IEEE 模式分析与机器智能汇刊》**29**(7) (2007) 7. Kolmogorov, V., Zabih, R.: 哪些能量函数可以通过图割最小化？《IEEE 模式分析与机器智能汇刊》**26**(2), 147–159 (2004) 8. Levi, Z., Zorin, D.: 几何优化的严格极小化解。《ACM 图形学汇刊》**33**(6), 185 (2014) 9. Malmberg, F., Ciesielski, K.C.: 两种用于优化基于最大范数的目标函数的多项式时间图标记算法。Springer《医学图像分析与视觉化杂志》**62**(5), 737–750 (2020)