arXiv:2303.05044v2 [cs.CC] 7 Jul 2023

常深度电路的值域规避：困难性与算法

Karthik Gajulapalli * Alexander Golovnev † Satyajeet Nagargoje ‡ Sidhant Saraogi §

摘要

范围规避（AVOID）是一类全搜索问题：给定一个布尔电路 C: $\{0,1\}^n \to \{0,1\}^m$, $m > n$, 任务是找到一个不在 C 值域内的 $y \in \{0,1\}^m$。对于整数 $k \geq 2$，NC^0_k-AVOID 是 AVOID 的一个特例，其中 C 的每个输出位至多依赖 k 个输入位。虽然针对 AVOID有一种非常自然的随机化算法，但该问题的确定性算法将带来许多有趣的推论。Ren、Santhanam 和 Wang（FOCS 2022）以及 Guruswami、Lyu 和 Wang（RANDOM 2022）证明，显式构造高公式复杂度函数、刚性矩阵和最优线性码，都可归约到 NC^0_4-AVOID，从而确立了 NC^0_4-AVOID 问题的条件困难性。另一方面，NC^0_2-AVOID 存在多项式时间算法，因此 NC^0_3-AVOID的复杂度仍悬而未决。

我们首次将显式构造问题归约到 NC^0_3-AVOID。具体而言，我们证明，对于$m = n + n^{2/3}$ 这一情形，若存在一个针对 NC^0_3-AVOID 的多项式时间算法（带有 NP 预言机），则可显式构造一个刚性矩阵，从而得到对数深度电路规模的超线性下界。

我们还为所有 NC^0_k-AVOID 问题提供了确定性多项式时间算法，适用于 $m \geq n^{k-1}/\log(n)$。以往的工作需要 NP 预言机，且要求更大的伸展 $m \geq n^{k-1}$。

*乔治城大学。电子邮箱：kg816@georgetown.edu。
†乔治城大学。电子邮箱：alexgolovnev@gmail.com。
‡乔治城大学。电子邮箱：satyajeetn2012@gmail.com。§乔治城大学。电子邮箱：ss4456@georgetown.edu。

1 引言

范围规避（Avoid）问题是：给定一个布尔电路 C: $\{0,1\}^n \to \{0,1\}^m$ 对于某个伸展 $m > n$，找出一个元素 $y \in \{0,1\}^m$ 不在 C的值域内。根据鸽巢原理，这样的 y 总是存在。该问题最初由 Kleinberg、Korten、Mitropolsky 和 Pa- padimitriou 提出 [KKMP21]，作为类别 丰富多项式空鸽巢原理 （Abundant Polynomial Empty Pigeonhole Principle，丰富多项式空鸽巢原理）的完全问题。简而言之，丰富多项式空鸽巢原理 包含这样的全搜索问题：其解的存在性可由联合界推出（例如 Shannon 的经典证明：大多数函数都需要规模呈指数级的电路）。

Korten [Kor21] 证明，针对 Avoid 的确定性算法将意味着显式构造出对计算复杂性领域至关重要的对象，并解决若干长期悬而未决的开放问题。这些对象包括电路复杂性很高的函数、刚性矩阵、伪随机生成器和 Ramsey 图。其核心思想是，在一个小型电路的输入空间中，有一种简洁的方式来编码所有“容易”的对象（例如低电路复杂性函数的描述），而该电路充当解码器。于是，解决 Avoid 问题便能得到一个“困难”对象（例如电路复杂性很高的函数），从而实现显式构造。事实上，前述研究 [KKMP21、Kor21] 表明，即使是一个带有NP 预言机的确定性算法，若能在 Avoid 多项式时间内解决该问题，也将带来复杂性理论上的突破。

目前已知的求解 Avoid 的唯一确定性算法是平凡的暴力算法，其运行时间为 $2^n \cdot \text{poly}(n, |C|)$。[1] 即使允许算法使用 NP 预言机，目前也没有已知的更优算法来求解 Avoid。另一方面，若同时使用 NP 预言机和随机性，则可以在多项式时间内求解Avoid：随机选取一个字符串 $y \in \{0,1\}^m$，然后用 NP 预言机直接检查 $y \in$ 是否属于 Range(C)。这表明 Avoid $\in$ FZPP^{NP}；再利用以非一致性模拟随机性的标准技巧，可得 Avoid $\in$ FP^{NP}/Poly。[2] 有趣的是，对于带有 NP 预言机的多项式时间算法类而言，无论伸展取何值， Avoid 问题的难度都相同$n+1 \le m \le \text{poly}(n)$ [KKMP21]。

对于一类电路 $\mathcal{C}$， $\mathcal{C}$-Avoid 问题是 Avoid 的一个特例，其中每个输出都由来自 $\mathcal{C}$ 的电路计算。Ren、Santhanam 和 Wang [RSW22]，以及 Guruswami、Lyu 和 Wang [GLW22] 近期的研究证明，对于 $\mathcal{C}$-Avoid，即使针对某些简单的电路类 $\mathcal{C}$，高效算法也足以得到各种显式构造。随后，Chen、Huang、Li 和 Ren [CHLR23] 从某个 $\mathcal{C}$-Avoid 问题的高效算法出发，重新推导出了针对 ACC^0 电路的最佳已知下界。虽然这表明，为 Avoid 问题设计高效算法是解决各种显式构造问题的一条有前景的途径，但 Ilango、Li 和 Williams [ILW23] 的工作证明，在某些密码学假设下，设计多项式时间算法存在障碍。

令 NC^1 表示深度为 $O(\log(n))$ 的布尔扇入 2 电路类，NC^0_k 表示这样一类布尔函数：每个输出至多依赖 k 个输入，其中 k为常数。[RSW22] 使用 [IK00、IK02、AIK06] 的完美编码，在多项式时间内将 NC^1-Avoid 归约到 NC^0_4-Avoid 。因此，[GLW22] 将上述大多数显式构造归约.....

[1]对于 [KKMP21], 研究的一个相关问题，这个平凡算法在条件成立时是紧的：其中 C 的值域大小远小于 2^{n+1}，且 C 由一个计算从 $[N]$ 到 $[M]$ 的函数的电路给出。[KKMP21] 给出了一个确定性归约，将对 n 个变量的布尔可满足性问题归约到电路 C 的 Avoid： $[2^n] \to [2^n + 2^{o(n)}]$，且运行时间为次指数级。因此，根据指数时间假设 [IPZ98，IP99],，该问题不存在运行时间为 $2^{o(n)}$ 的确定性（及随机化）算法。[2]这里，复杂性类 FP、FE、FZPP 只是判定复杂性类 P、E、ZPP 的函数型对应类。

造归约为 [Kor21]（以及若干新构造！）到 NC^1-Avoid，进而归约到 NC^0_4-Avoid。具体而言，多项式时间确定性算法（即使带有 NP 预言机）若能解决 NC^0_4-Avoid问题，将会为复杂性理论带来突破性成果。

[GLW22] 给出了一个多项式时间算法，用于求解 NC^0_2-Avoid，适用于任意伸展 $m \geq n+1$。如上所述，NC^0_4-Avoid 可能难以高效求解。因此，NC^0_3-Avoid的复杂度问题仍未解决。

开放问题 1 ([GLW22])。 我们能否将显式构造问题归约为求解 NC^0_3-Avoid? 或者我们能否在多项式时间内求解 NC^0_3-Avoid ？

与一般的 Avoid 问题不同，对于较大的伸展值 m，NC^0_k-Avoid 可能容易得多。事实上，一方面，对于较小伸展 $m = n + o(n)$ 的 NC^0_k-Avoid 能够编码困难的显式构造问题 [RSW22, GLW22]。另一方面，对于$m = \Omega(n^k)$ 的 NC^0_k-Avoid 可以在多项式时间内轻松求解：由于至多依赖于 k 个输入（共 n 个输入）的不同函数数量为 $O(n^k)$，因此该问题的每个实例中必定有两个输出计算出相同的函数。为这两个输出赋予不同的值，即可求解 NC^0_k-Avoid。

[GLW22] 提出了一种算法，可以在多项式时间内借助 NC^0_k-输入、伸展为 $m \geq \Omega(n^{k-1})$ 的情况下求解该问题，并使用一个 NP 预言机。[3] 这一对平凡算法的改进引出了一个自然问题：对于更小的伸展值，是否也能求解 NC^0_k-Avoid 呢 m。

开放问题 2：设计一个多项式时间算法（带有一个 NP 预言机），用 n 个输入和伸展 $m = o(n^{k-1})$ 求解 NC^0_k-Avoid，其中 $k \geq 3$。

1.1 我们的结果

Shannon 的经典结果[Sha49] 表明，大多数具有 n 个变量的布尔函数都需要规模呈指数增长的布尔电路。尽管如此，目前已知的电路规模下界（甚至是对数深度电路，即 NC^1）对于属于 P （甚至属于 E^{NP}）的函数而言，仍然是 $3.1n - o(n$，由 Li 和 Yang 证明 [LY22]）。电路复杂性中的一个核心问题，是证明计算某个显式函数的 NC^1 电路的门数量具有超线性下界，该问题见 [Val77、AB09 和 Frontier 3]。

类似地，对于 线性 NC^1 电路——NC^1 电路中的每个门都计算其两个输入的异或（或其否定）——目前尚无针对显式线性映射 $M \in \mathbb{F}_2^{n\times n}$复杂度的超线性下界。针对线性电路所知的最佳下界是 $3n - o(n$由 Chashkin 证明的 [Cha94]。

在我们的第一个结果（定理 3.2）中，我们通过证明：若存在一个多项式时间算法来处理 NC^0_3-Avoid，则可显式构造一个需要线性规模的 NC^1电路且规模超线性的映射（由此证明 NC^0_3-Avoid 的困难性）。

定理 1. 一个用于 NC^0_3-Avoid （伸展为 $m = n + O(n^{2/3})$ 的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法，意味着可以在 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）中显式构造一个线性映射，该映射无法由规模为 $o(n\log\log(n))$ 的线性 NC^1电路计算。

我们的证明定理 1 首先将刚性矩阵的显式构造归约为 NC^0_3-Avoid(定理 3.2）。如果一个矩阵 $M \in \mathbb{F}_2^{n\times n}$不能写成以下形式，则称其为（$r$， s）-刚性矩阵：一个秩为 $M =$

[3] GLW 的算法[22] 并未用尽 $\mathsf{FP}^{\mathsf{NP}}$ 的全部能力：它输出一个击中集 $H \subseteq \{0,1\}^m$，使得对于每个 NC^0_k 函数 C，至少有一个点 $y \in H$ 不在 C 的值域内。只有这时，算法才会查看输入函数，并利用 NP 预言机找到一个解。

$L+S$ 的矩阵 r 与一个矩阵 L 之和，且后者 S 每行至多含有 s 个非零元素。在一项开创性工作中，Valiant [Val77] 提出了通过矩阵刚性证明线性 NC^1 电路规模的超线性下界的方法。Valiant 证明了，对于任意常数 εn，n^{ε}-刚性矩阵 $M \in \mathbb{F}_2^{n\times n}$要求线性 $\varepsilon > 0$ 规模的 NC^1 电路，其规模为 $\Omega(n\log\log(n))$。定理 1 现在可直接由 定理 3.2作为推论得到。

目前已知的刚性矩阵构造尚未达到 Valiant 的电路下界所需的参数。[Fri93、PR94、SSS97] 在多项式时间内构造出一个 $\left(r\Omega\left(\frac{n}{r}\log(\frac{n}{r})\right)\right)$-刚性矩阵，[GT16] 给出了一个 $\left(r,\Omega\left(\frac{n^2}{r^2}\log(n)\right)\right)$-刚性矩阵，所需时间为 $2^{O(n)}$（针对 $r \geq \sqrt{n}$），而[AC19、BHPT20] 在带 NP 预言机的多项式时间内给出了 $(2^{\varepsilon\log(n)/\log\log(n)}, \Omega(n))$-刚性矩阵。然而，即使是一个能对任意常数$\varepsilon > 0$ 实现伸展 $m = n + n^{12/17-\varepsilon}$ 的 NC^0_3-Avoid 的 $\mathsf{FP}^{\mathsf{NP}}$ 算法，也已经可以改进这些已知的刚性矩阵构造。

事实上，我们将显式构造刚性矩阵的问题归约为一个我们称为二次多项式-Avoid的问题，其中每个输出都计算输入的一个二次多项式。遵循 Ren、Santhanam 和 Wang 的方法 [RSW22], ，该问题可以利用 Applebaum、Ishai 和 Kushilevitz 的完美编码方案 NC^0_3-Avoid 归约到 [AIK06]。[4]

在算法方面，我们在解决开放问题 2 上取得了部分进展。首先，我们给出了一个简单的确定性多项式时间算法，用于解决伸展为 $m \geq \binom{n}{2}/3 + 2n$ 的 NC^0_3-Avoid 问题（见附录 A）。该算法已经优于目前已知的 NC^0_3, 问题的最佳算法，因为它不使用 NP 预言机。随后，在定理 4.5 中，我们将该算法扩展到求解所有常数 k 下的 NC^0_k-Avoid 问题。回想一下，目前该问题的最佳算法使用 NP 预言机 [GLW22]，并在多项式时间内解决 $m \geq \Omega(n^{k-1})$ 的情形。我们从两个方面改进了这一结果：我们的算法不使用 NP 预言机，而且对于伸展 $m \geq n^{k-1}/\log(n)$，也能在多项式时间内运行。

定理 2。存在一个确定性多项式时间算法，对于每个 $k \geq 3$ 和 $m \geq n^{k-1}/\log(n)$，都能求解具有 n 个输入、伸展为 m 的 NC^0_k-Avoid 问题。

1.2 证明概述

困难性 NC^0_3-Avoid。Valiant [Val77] 证明了线性 NC^1 电路若只有线性数量的门，则只能计算非刚性线性映射 $M \in \mathbb{F}_2^{n\times n}$，即可以写成如下和的映射 M，其中秩 $\mathrm{rank}(Q) \leq \varepsilon n$，且 S 的每一行至多含有 n^{δ} 个 1。在本节余下部分，我们的非刚性矩阵可以写成一个秩为 $\leq n/10$ 的矩阵与一个行稀疏度至多为 $n^{0.1}$的矩阵之和。因此，构造一个刚性矩阵将意味着其线性 NC^1 电路需要超线性规模才能计算该矩阵。

为了将显式构造一个 $n\times n$刚性矩阵的问题归约为求解 NC^0_3-Avoid 的一个实例，我们设计一个 NC^0_3函数 f: $\{0,1\}^{p(n)} \to \{0,1\}^{n^2}$，其中某个多项式 $p(n) < n^2$，使得对每个非刚性矩阵 $M \in \{0,1\}^{n\times n}$，都存在 $x \in \{0,1\}^{p(n)}$ 满足 $f(x) = M$。现在，函数 f 的 NC^0_3-Avoid 问题的任何解 $M' \in \{0,1\}^{n\times n}$ 都必须是一个刚性矩阵。

在构造这样的NC^0_3函数 f之前，我们首先设计一个函数 g： $\mathbb{F}_2^{n^2/2} \to \mathbb{F}_2^{n^2}$，其中 g 的每个输出位都是输入的二次多项式，而 g 的值域包含所有非刚性矩阵。对于函数Avoid 的二次 g 问题的解将为我们提供一个刚性矩阵。遵循 [RSW22], ，我们接着可以将完美编码方案 [IK00、IK02、AIK06] 应用于 g，并且

[4]Ren、Santhanam 和 Wang 使用以下定义来描述完美编码：函数 $\widehat{f}$是函数 f的完美编码，当且仅当存在一个多项式时间算法 Dec，使得对于所有 x, y: $\mathrm{Dec}(y) = f(x) \iff \exists r, y = \widehat{f}(x, r)$。

得到一个NC^0_3 函数 f，如所需（见引理 2.7）。实际上，这将把在 AVOID上求解 g 的问题归约为在 AVOID 上求解 f。

现在我们构造一个二次函数 g: $\mathbb{F}_2^{n^2/2} \to \mathbb{F}_2^{n^2}$ 其输入编码所有非刚性矩阵，即对于每个非刚性矩阵 M，都存在一个 $x \in \{0,1\}^{n^2/2}$使得 $f(x) = M$。非刚性矩阵 M可以写成 $M = LR + S$，其中 L、$R^T \in \mathbb{F}_2^{n \times n/10}$，并且 S的每一行至多包含 $n^{0.1}$ 个 1。函数 g的前 $n^2/5$ 个输入将对应于 L和 R的元素。注意， LR 中的每个元素都是 L 和 R 中元素的二次函数，因为它只需计算 L 中的一行与 R中的一列的内积。现在，对于 S的每个 $n^{0.1}$-稀疏行，我们展示如何使用 $n^{0.6}$ 个输入和一个二次函数对其进行编码。对 S 的每一行重复这一过程，即可完成证明。

我们将含有 n 个元素的 $n^{0.1}$-稀疏行视为矩阵 A。由于 A 至多有 $n^{0.1}$ 个非零元素，因此 $\mathrm{rank}(A) \le n^{0.1}$。它可以写成乘积 $A = BC$，其中 B、$C^T \in \mathbb{F}_2^{\sqrt{n} \times n^{0.1}}$。因此，存在一个二次函数 h，以 B、大小为 $2n^{0.6}$ 的 C 作为输入，并输出稀疏矩阵 A。

所提出的 s-稀疏向量在 $\mathbb{F}_2^n$ 中的编码只有在 $s < \sqrt{n}$ 时才是非平凡的（否则， h 的输入数量会超过输出数量）。因此，我们无法将整个矩阵 S 编码为 $n^{1.1}$-稀疏向量，置于 $\mathbb{F}_2^{n^2}$ 中。然而，对于 Valiant 的方法——证明电路下界，我们可以假设 S 是 $n^{0.1}$-行 稀疏的。[5] 因此，我们可以分别对 S 的每一行进行编码，仅使用 $O(n^{0.6})$ 个输入，从而得到一个 S 在 $O(n^{1.6})$ 位中的编码。在引理 3.1中，我们将说明，当编码函数允许具有更高（但仍为常数）的度 d 时，如何采用稍优的稀疏度参数。

NC^0_3-Avoid 的简单算法。 我们先简要介绍一种针对 NC^0_3-AVOID 、伸展为 $m \ge \binom{n}{2}/3 + 2n$ 的简单确定性多项式时间算法（见 附录 A）。该算法已经优于目前已知的最佳 NC^0_3-AVOID算法，因为我们的算法不使用 NP 预言机。

如果我们拥有一个 #SAT 预言机，那么即使伸展为 $m = n + 1$，我们也可以解决 NC^0_3-AVOID 。我们的算法会迭代地为前 n 个输出确定常量赋值，以最小化映射到当前（部分）输出赋值的输入数量。在全文中，我们称与这些（部分）输出赋值一致的输入为一致输入。在描述算法之前，需要特别指出的是，我们始终将电路输出固定为常量赋值。在每次迭代中，我们都会使用 #SAT 预言机来寻找一个输出赋值，使输入集合的规模至少减少一半。固定前 n 个输出后，仍有至少 $m - n \ge 1$个未赋值输出，并且只有一个输入点 $x \in \mathbb{F}_2^n$ 与之前指定的输出位一致。这使我们能够找到第（$n+1$）个输出位的赋值，使输出位所确定的字符串位于电路的值域之外。

遗憾的是，对于这类多输出电路，即使是近似求解 #SAT 也很困难。由于缺乏高效的 #SAT 算法，我们的算法会维护一个仿射子空间 $\mathcal{S}$，其中包含 $\mathbb{F}_2^n$ 中所有与当前部分赋值一致的输入（$\mathcal{S}$ 也可能包含与当前部分赋值不一致的输入）。我们会谨慎设置输出值，确保每次迭代都至少将 $\mathcal{S}$的维数降低一。这样，经过 $n+1$ 步后，我们就能找到 NC^0_3-AVOID 问题的一个解。不过，我们的算法只有在

[5]或者，也可以通过马尔可夫不等式论证：如果一个矩阵 M 无法表示为秩为-r 的矩阵与 $n^{0.1}$-*row* 稀疏矩阵之和，那么 M 也无法表示为秩为-$2r$ 的矩阵与 $rn^{0.1}$-*globally* 稀疏矩阵之和。

伸展为 $m \geq \Omega(n^2)$ 时才能运行。

不失一般性，我们假设每个输出恰好读取三个输入位。在每次迭代中，当前尚未赋值的输出数量为 $> \binom{n}{2}/3$。这使我们能够找到一对输出 y_1 和 y_2，它们共享一对输入变量。[6] 例如，$y_1 = f_1(x_1, x_2, x_3)$ 和$y_2 = f_2(x_2, x_3, x_4$。我们将找到对 y_1 和 y_2 的一个常量赋值，使仿射子空间 $\mathcal{S}$的维数降低。

注意，(x_1, x_2, x_3, x_4) 有 16 种赋值，而 (y_1, y_2) 有四种不同的取值。因此，存在一种对 $y_1 = c_1$, $y_2 = c_2$ 赋值的方式，使得至多四个点 (x_1, x_2, x_3, x_4) 映射到这些输出值。注意，总存在一个超平面 H 包含 $\mathbb{F}_2^4$ 中任意四个点。令 $\mathcal{H}$ 为将 H 扩展到所有 n 输入后得到的仿射子空间。那么，$\mathcal{S} \cap \mathcal{H}$ 给出了一个仿射子空间，其中包含所有与赋值 $y_1 = c_1, y_2 = c_2$ 一致的输入。

若 $\mathcal{S} \nsubseteq \mathcal{H}$，那么我们可以赋值两个输出，使仿射子空间的维数降低为 $\dim(\mathcal{S} \cap \mathcal{H}) < \dim(\mathcal{S})$。否则，若 $\mathcal{S} \subseteq \mathcal{H}$，那么我们无需考虑输入的全部 16 种赋值（$x_1$, x_2, x_3, x_4），而只需关注其中至多 8 种属于 H的赋值，这会使问题变得更简单（我们将在定理 A.1中证明这一点）。

更好的NC^0_k-Avoid 算法。这个简单算法的主要瓶颈在于，它维护一个必须包含所有一致输入的仿射子空间。虽然仿射子空间易于处理，但它们的表达能力不足以准确描述与任意部分赋值一致的所有输入。为了改进先前的算法，我们将维护比仿射子空间表达能力更强的结构——仿射子空间的并。下面我们概述解决NC^0_3-Avoid 、伸展为 $m \geq \Omega(n^2/\log(n))$ 的方法。定理 4.5 将其推广到求解NC^0_k-Avoid 问题，伸展为 $m \geq \Omega(n^{k-1}/\log(n))$，适用于所有 k的取值。

同样，不失一般性，我们假设每个输出恰好依赖三个输入。考虑一个二分图，其中左侧顶点对应 n 个输入，右侧顶点对应 m 个输出；如果输出 y_j 依赖输入 x_i，则输入与输出构成的顶点对 (x_i, y_j) 之间有一条边。首先，我们选取 $t = 3n^2/m$ 个度最高的输入 $I = \{x_1, \ldots, x_t\}$。它们的邻域必须包含至少 $3n$ 个不同的输出 $O = \{y_1, \ldots, y_{3n}\}$。[7] 令 C 为由 O 中的输出及其对应输入定义的子电路。现在，我们将找到一个不属于 Range(C) 的 $y \in \mathbb{F}_2^{3n}$。

首先，考虑对 $I = \{x_1, \ldots, x_t\}$ 中的输入进行的所有 2^t 种赋值，由此得到电路 C_1、.....、C_{2^t}。由于 O 中的每个输出都与 I 中至少一个输入相连，将赋值固定到输入 I 后，每个 C_i 都会化简为一个 NC^0_2 电路。换言之，我们将 NC^0_3-Avoid 化简为 2^t 个 NC^0_2-Avoid 实例的或：我们需要找到一个不属于所有 C_i 值域的 $y \in \{0,1\}^{3n}$。对于每个电路 C_i，我们将维护一个仿射子空间 $\mathcal{S}_i$，其中包含与当前输出部分赋值相一致的所有输入。

我们的算法通过迭代地固定输出位来实现 $\{y_1, \ldots, y_{3n}\}$，使得每一步中，（互不相交的）仿射子空间并集中的点总数 $\mathcal{S}_i$ 都按一个常数因子减少，最终使所有子空间都变为空集。我们注意到（见引理 4.3），对于任意仿射子空间 $\mathcal{S}_i$，赋值 $y_i = 0$ 或 $y_i = 1$ 总能使 $\mathcal{S}_i$的维数减少一。因此，通过在所有子空间中选取“最佳”赋值 $y_i = c$，我们可以 $\mathcal{S}_i$

[6]~~每个输出都关联 3 对输入位，总计有 $3m >$~~ $\binom{n}{2}$ 对。根据鸽巢原理，至少有一对输入出现在两个输出中。[7]由于输出数量为 m，且每个输出的度为 3，因此图中的边数为 $3m$，输入的平均度为 $3m/n$。那么，度最高的 t 个输入的总度至少为 $3mt/n$，并且必须连接到至少 $mt/n = 3n$ 个不同的输出。

将这类仿射子空间并集的规模减少至原来的常数因子 $4/3$。重复此过程 $\log_{4/3}(2^n)+1<3n$ 步，即可完成证明。

1.3 开放问题

我们的工作引出了关于 NC^0_3-Avoid 和度-2-Avoid复杂性的若干自然问题。我们将刚性矩阵的显式构造归约为求解度-2-Avoid，进而归约为 NC^0_3-Avoid，并采用适当的伸展。

开放问题 3：其他显式构造问题能否归约为 NC^0_3-Avoid 或度-2-Avoid?

特别是，我们猜想，具有最优参数的线性码和列表可译码码的构造 [GLW22] 可能是这类归约的理想候选。

在从度-2-[AIK06] 归约到 Avoid NC^0_3-Avoid 时使用 $m=n+o(n)$ 的编码，几乎总会将所需的伸展降低到 $m=n+o(n)$（如 第 3 节 所强调的）。如果能找到一种更高效的编码，或找到从度-2-Avoid 到 NC^0_3-Avoid的归约，也会很有意义。这有可能提高 NC^0_3-Avoid 所需的伸展，从而使显式构造更容易获得。

开放问题 4。我们能否构造一种更高效的从二次-Avoid 到 NC^3_0-Avoid 的归约?

我们认为，二次-Avoid 可能本身就很有研究价值，因为它允许更大的伸展。例如，要改进刚性矩阵的构造，只需针对常数 $\varepsilon>0$，解决超线性伸展 $m\geq n^{12/11-\varepsilon}$ 下的二次-Avoid 。事实上，当伸展为 $m\geq n^2$ 时，二次-Avoid 很容易解决。注意，在 n 个变量上至多有 $\binom{n}{2}$ 个不同的二次单项式。如果 $m\geq n^2$，那么我们可以用一个新变量替换每个不同的单项式。因此，我们将得到 m 个关于 $<m$ 个变量的线性函数。对于这个线性函数实例，我们可以采用与上一节所述类似的降维策略来解决 Avoid 。

开放问题 5。是否存在无需归约到 NC^0_3-Avoid 的算法技术来解决二次-Avoid ？

对于 NC^0_3-Avoid 问题，我们的算法对任意伸展$m\geq n+1$都在确定性时间 $2^{O(n^2/m)}$ 内运行。特别地，这恢复了 $m=n+1$最困难情形下的指数时间暴力算法。对于 NC^0_3-Avoid，如果能为不使用 NP 预言机的确定性算法获得匹配的条件下界，将会非常有趣。

开放问题 6。对于不使用 NP 预言机的 NC^0_3-Avoid 确定性算法，其复杂度是否存在 $2^{\Omega(n^2/m)}$ 的条件下界?

最后，自然会产生这样的问题：带有 NP 预言机的算法能否更高效地解决 NC^0_3-Avoid ？

开放问题 7. 是否存在带 NP 预言机的多项式时间算法，能够在伸展为 $m=o(n^2/\log(n))$ 时求解 NC^0_3-Avoid ？

1.4 结构

本文其余部分安排如下。第 2 节介绍所有必要的背景材料，包括第 2.3 节中从度-d-Avoid 到 NC^0_{d+1}-Avoid 的归约。第 3 节将构造显式刚性矩阵的问题归约到 NC^0_3-Avoid。第 4 节给出可在多项式时间内求解 NC^0_k-Avoid 的确定性算法，其伸展为 $m \geq n^{k-1}/\log(n)$。最后，附录 A 给出了另一种确定性多项式时间算法，用于在伸展为 $m \geq \Omega(n^2)$ 时求解 NC^0_3 。

2 预备知识

对于每个 $a \in \mathbb{F}_2^n$ 以及 $\mathbb{F}_2^n$的子空间 L ，我们都可以定义一个仿射子空间 $\mathcal{A} \subseteq \mathbb{F}_2^n$ ，其中 $\mathcal{A} = \{a + v \mid v \in L\}$。仿射子空间的维数 $\dim(\mathcal{A})$ 与定义它的线性子空间 L 的维数相同。等价地， $\mathbb{F}_2^n$ 上指定的一组超平面所包含的点集也刻画了 $\mathbb{F}_2^n$的一个仿射子空间。这些超平面可以写成一个线性方程组 $Ax = b$，相应仿射子空间 $\mathcal{A}$ 的维数可计算为 $\dim(\mathcal{A}) = n - \mathrm{rank}(A)$。

本文中的电路和算法通常在布尔超立方体 $\{0,1\}^n$上运行。我们研究多输出电路 $C : \{0,1\}^n \to \{0,1\}^m$ ，其中 $m > n$ ，且 m 称为电路的伸展。部分输出赋值 $y \in \{0,1,*\}^m$是指将电路的一部分输出位置为常数。对于电路的输入 $x \in \{0,1\}^n$ ，如果 $C(x)$在固定的位上与 y 一致，我们就称 x 与部分输出赋值 $y \in \{0,1,*\}^m$相容。在有需要时，电路的输入（分别为输出）空间也可以视为有限域 $\mathbb{F}_2$上的向量空间 $\mathbb{F}_2^n$ （分别为 $\mathbb{F}_2^m$）。

复杂性类 FP、$\mathrm{FP}^{\mathrm{NP}}$、FE 和 $\mathrm{FE}^{\mathrm{NP}}$ 是搜索问题类，类似于判定问题类 P、P^{NP}、E 和 E^{NP}。例如，类 FP 包含所有可由确定性多项式时间图灵机计算的函数。

2.1 电路与矩阵刚性

本文研究电路类 NC^0_k 和 NC^1, ，其定义如下。

定义 2.1。（NC 电路）电路类 NC^i 包含以 n为输入、深度为 $O(\log^i(n))$ 的多输出布尔电路，其中每个门的扇入为 2。我们特别关注以下几类电路：

- 对于每个常数 $k \geq 1$，NC^0_k 是这样一类电路：每个输出至多依赖k 个输入。
- NC^1 是深度为 $O(\log(n))$且所有门的扇入均为 2 的电路类。
- 线性 NC^1 电路是深度为 $O(\log(n))$ 的电路，其中每个门的扇入为 2，并计算一个仿射函数，即对其两个输入进行异或或取其反。

证明规模超线性下界一直是电路复杂性领域一个长期悬而未决的问题，所讨论的是计算一个来自 NC^1 电路的（线性） n输出函数，该函数来自 FP 甚至是 $\mathrm{FE}^{\mathrm{NP}}$ [Val77、AB09、Frontier 3]。Valiant [Val77] 提出了一种利用矩阵刚性概念证明线性NC^1 电路超线性下界的方法。

定义 2.2（矩阵刚性）。对于 $r, s \in \mathbb{Z}^+$，矩阵 $M \in \mathbb{F}_2^{n\times n}$是 (r, s)-刚性的，如果 M 不能写成若干项之和

$$M = L + S\,,$$

其中 $L, S \in \mathbb{F}_2^{n\times n}$, L 是低秩的，即 $\mathrm{rank}(L) \le r$，而 S 是行稀疏的，即 S 的每一行至多有 s个非零项。

Valiant [Val77] 证明，若线性算子由足够刚性的矩阵给出，则至少需要规模为 $\Omega(n\log\log(n))$ 的线性 NC^1 电路，但即便在 $\mathsf{FE}^{\mathsf{NP}}$ 中，目前仍没有已知的此类刚性矩阵构造。

定理 2.3 ([Val77]). 若矩阵族 $(M_n)_{n\ge 1}$, $M_n \in \mathbb{F}^{n\times n}$ 是 $(\varepsilon n, n^{\delta})$-刚性的，且深度为常数$\varepsilon, \delta > 0$，则线性映射 $x \mapsto Mx$ 至少需要规模为线性级别的 NC^1 电路，规模为 $\Omega(n\log\log(n))$。

2.2 电路的值域规避

在值域回避问题中，给定一个有 n 个输入和 m 个输出的电路 C， $m > n$，目标是找到一个位于 C 的值域之外的 m 位字符串。

定义 2.4（Avoid）。在 Avoid 问题中，给定一个电路 $C: \{0,1\}^n \to \{0,1\}^m$ 的描述，其中 $m > n$，任务是找到一个 $y \in \{0,1\}^m$，使得 $\forall x \in \{0,1\}^n: C(x) \ne y$。

函数 $m = m(n)$ 称为多输出电路 C 的伸展。注意， Avoid是一个全搜索问题，即总是存在这样的 $y \in \{0,1\}^m$，因为 $m > n$。我们关注一个限制更严格的问题，其中附加承诺输入电路 C 属于固定的电路类 $\mathcal{C}$。

定义 2.5（$\mathcal{C}$-Avoid）。在 $\mathcal{C}$-Avoid 问题中，给定一个电路 $C: \{0,1\}^n \to \{0,1\}^m$ 的描述，其中 $m > n$，且 $C \in \mathcal{C}$，任务是找到一个 $y \in \{0,1\}^m$，使得 $\forall x \in \{0,1\}^n: C(x) \ne y$。

特别地，我们关注常数 $k \ge 1$ 下的 NC^1-Avoid 和 NC^0_k-Avoid 。我们还将考虑这样一类函数：其中每个输出都是输入的一个次数至多为 d 的多项式，定义在 $\mathbb{F}_2$ 上。

定义 2.6（次数-d-Avoid）。在次数-d-Avoid 问题中，给定函数 $C: \mathbb{F}_2^n \to \mathbb{F}_2^m$（其中 $m > n$）的描述，且每个输出都可以通过一个关于 n 个输入的次数为 $\le d$ 的多项式计算，任务是找到一个 $y \in \mathbb{F}_2^m$，使得 $\forall x \in \mathbb{F}_2^n$: $C(x) \ne y$。

2.3 低次数与低局部性

完美随机编码由 [AIK06] 为各种密码学应用而引入。我们关注完美编码的以下性质：对于一个布尔函数f: $\{0,1\}^n \to \{0,1\}^m$及其编码 $\widehat{f}$: $\{0,1\}^{n+\ell} \to \{0,1\}^{m+\ell}$，存在一个多项式时间解码算法 $\mathrm{Dec}: \{0,1\}^{m+\ell} \to \{0,1\}^m$，使得对于所有满足 $\mathrm{Dec}(y) = f(x)$ 的 $y \in \{0,1\}^{m+\ell}$ 和 $x \in \{0,1\}^n$，都存在 $r \in \{0,1\}^\ell$ 使得 $y = \widehat{f}(x, r)$。此性质可用于多项式时间归约，具体如下。给定函数 f 的布尔可满足性问题的一个解，即 $\sim y \notin$ 值域(f)，只需计算 $\mathrm{Dec}(y) \notin$ 值域(f)，便能在多项式时间内找到函数 f 的布尔可满足性问题的一个解。

[AIK06] 首先将 NC^1 函数编码为 3 次函数。然后，他们将每个度为 d 的函数编码为 NC^0_{d+1} 函数。复合这两种编码，即可得到将 NC^1 函数编码为 NC^0_4 的编码。利用这种编码，[RSW22] 给出了从 NC^1-Avoid 到 NC^0_4-Avoid 的多项式时间归约。我们只使用 [AIK06] 结果中的一部分：存在从度为 d-Avoid 到 NC^0_{d+1}-Avoid 的多项式时间归约。为完整起见，我们在此给出证明。

引理 2.7。设 $d \geq 2$ 是一个常数，且 $f\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ 是一个多输出函数，其中每个输出都计算 k 个度为 $\leq d$ 的单项式之和。那么存在一个由 NC^0_{d+1} 电路计算的函数 $\widehat{f}\colon \{0,1\}^{n+(2k-1)m} \to \{0,1\}^{2km}$，以及一个多项式时间算法 Dec：$\{0,1\}^{2km} \to \{0,1\}^m$，使得对于所有

$$x\ y \qquad y\ =f\ x \qquad r \in \{0,1\}^{(2k-1)m}$$

，若 Dec( ) ( )，则存在一个 使得 $f(x,r)\ =y$。

证明。我们沿用 [AIK06] 中构造的编码。首先，我们为 f 中的每个单输出函数 g 构造一种编码 $\widehat{g}$。设 $g(x)\ =T_1(x)\ +T_2(x)\ +\cdots+T_k(x)$ 是一个度为 d 的单输出函数，其中每个 $T_i(x)$ 都是一个度至多为 d 的单项式。考虑 $g,\widehat{g}\colon \{0,1\}^n \times \{0,1\}^k \times \{0,1\}^{k-1} \to \{0,1\}^{2k}$ 的编码，定义如下

$$\begin{aligned}\widehat{g}(x,r,s) = (&T_1(x)-r_1, && T_2(x)-r_2, && \ldots && T_{k-1}(x)-r_{k-1}(x), && T_k(x)-r_k,\\ &r_1-s_1, && s_1+r_2-s_2, && \ldots && s_{k-2}+r_{k-1}-s_{k-1}, && s_{k-1}+r_k)\,.\end{aligned}$$

显然，$\widehat{g}$ 的每个输出位都可以由一个 NC^0_{d+1} 电路计算。我们定义一个多项式时间算法 $\mathrm{Dec}_{\widehat{g}}\colon \{0,1\}^{2k} \to \{0,1\}$，使得如果 $\mathrm{Dec}_{\widehat{g}}(y)\ =g(x)$，那么存在 r 和 s 满足 $\widehat{g}(x,r,s)\ =y$。给定 $y \in \{0,1\}^{2k}$，$\mathrm{Dec}_{\widehat{g}}(y)$ 将 y 的各位相加并对 2 取模。

假设对于某个 $x \in \{0,1\}^n$，$\mathrm{Dec}_{\widehat{g}}(y)\ =g(x)$，也就是说，

$$\mathrm{Dec}_{\widehat{g}}(y) = \sum_{j=1}^{2k} y_j = g(x) = \sum_{j=1}^{k} T_j(x)\,. \tag{1}$$

现在我们将证明，存在 r 和 s 使得 $\widehat{g}(x,r,s)\ =y$。对于每个 $j \in [k]$，我们设定$r_j\ =T_j(x)-y_j$。我们还设定 $s_1 = r_1 - y_{k+1}$，并依次为每个$j \in \{2,\ldots,k-1\}$设定 $s_j = s_{j-1} + r_j - y_{k+j}$。根据定义，$\widehat{g}(x,r,s)$ 的前 $2k-1$ 位等于 y 的前 $2k-1$ 位。对于最后一位，注意：

$$\begin{aligned}s_{k-1}+r_k &= \sum_{i=1}^{k} r_i - \sum_{i=k+1}^{2k-1} y_i && \text{（根据 } s\text{的定义）}\\ &= \sum_{i=1}^{k} T_i(x) - \sum_{i=1}^{2k-1} y_i && \text{（根据 } r \text{ 的定义）}\\ &= y_{2k}\,. && \text{（根据方程（1））}\end{aligned}$$

因此，对于构造出的 r 和 s，$\widehat{g}$（x、r、s）$=y$，符合要求。

假设 $f(x)\ =(f_1(x)\ f_2(x)\ldots,f_m(x))$，其中每个 $f_i(x)$ 都是至多包含 k 个度为 $\leq d$ 的单项式 ^ 之和。令 f_i 为上述定义的 f_i 的编码。那么，我们对 f 的编码就是其各个输出的编码的串接，$\widehat{f}$：$\{0,1\}^{n+(2k-1)m} \to \{0,1\}^{2km}$，其中

$$\widehat{f}(x,r^{(1)},r^{(2)},\ldots,r^{(m)},s^{(1)},s^{(2)},\ldots,s^{(m)}) = (\widehat{f}_1(x,r^{(1)},s^{(1)}),\ldots,\widehat{f}_m(x,r^{(m)},s^{(m)}))\,. \tag{2}$$

输入 $y=(y_1,y_2,\ldots,y_m)\in\{0,1\}^{2km}$ 时，解码算法返回

$$\mathrm{Dec}(y)=(\mathrm{Dec}_{\widehat{f_1}}(y_1),\ldots,\mathrm{Dec}_{\widehat{f_m}}(y_m))\,. \tag{3}$$

假设对于某个 $x\in\{0,1\}^n$，$\mathrm{Dec}(y)=f(x)$。那么对于所有 $i\in[m]$，$\mathrm{Dec}\,\widehat{f_i}(y_i)=f_i(x)$。根据上面的证明，存在 $r^{(i)}$ 和 $s^{(i)}$，使得 $y_i=\widehat{f_i}(x,r^{(i)},s^{(i)})$，因此，

$$y=(y_1,\ldots,y_m)=(\widehat{f_1}(x,r^{(1)},s^{(1)}),\ldots,\widehat{f_m}(x,r^{(m)},s^{(m)}))=\widehat{f}(x,r^{(1)},s^{(1)},\ldots,r^{(m)},s^{(m)})\,.$$

最后，Dec 的运行时间为 $O(mk)$，因为对于每个 i，它会运行 $\mathrm{Dec}^{\widehat{f_i}}$ 迭代 m 次，每次只需计算 $2k$ 位的和。由于每个 $\widehat{f_i}$ 都属于 NC^0_{d+1}，因此 $\widehat{f}$ 也属于 NC^0_{d+1}。 □

现在，根据 [RSW22], ，我们得出结论：存在一个从度-d-Avoid到 NC^0_{d+1}-Avoid的多项式时间归约。

推论 2.8. 对于每个 $d\geq 1$，如果存在一个针对 NC^0_{d+1}-Avoid 的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法，那么就存在一个针对度-$d$-Avoid 的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法。

证明。设 f 是一个度为 d-Avoid 问题的输入，该问题有 m 个输出位。那么，f 的每个输出位都是至多 $k=O(n^d)$ 个度为 d 的单项式之和。设 $\widehat{f}$ 是由引理 2.7 保证的 f 在 NC^0_{d+1} 中的编码。注意到 $\widehat{f}\colon\ \{0,1\}^{n+(2k-1)m}\to\{0,1\}^{2km}$。根据推论的假设，存在一个 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法，可返回一个 $y\notin\mathrm{Range}(\widehat{f})$。于是，根据引理 2.7，$\mathrm{Dec}(y)\notin\mathrm{Range}(f)$，且 Dec 可在多项式时间内运行。因此，存在一个用于度为 d-Avoid 问题的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法。 □

3 NC^0_3-Avoid 的困难性

在本节中，我们将构造显式刚性矩阵的问题归约为求解 NC^0_3-Avoid 的算法任务。首先，在 引理 3.1 中，我们给出一个显式二次函数 $f\colon\{0,1\}^k\to\{0,1\}^n$，$k\ll n$，其值域包含所有长度为 n 的稀疏向量。注意，这样的函数 f的度必定至少为 2。事实上，若 f 是仿射函数，且其值域包含所有稀疏度至多为 1 的向量，那么其值域的维数必定为 n，而 f 的输入数量将为 $k\geq n$。

接下来，在定理 3.2中，我们将此引理与来自引理 2.7 的从 2 度Avoid 到NC^0_3-Avoid 的约化结合起来，从而得出结论：针对NC^0_3-Avoid 的高效算法将能显式构造刚性矩阵。

引理 3.1。对于每个 $d\geq 1$ 以及每个多项式时间可计算的 $s:=s(n)<\frac{n^{1-1/d}}{d}$，都存在一个多项式时间可计算的函数 $f\colon\mathbb{F}_2^{dsn^{1/d}}\to\mathbb{F}_2^n$，其值域包含所有稀疏度至多为 s的向量，并且 f 的每个输出都是一个 d 度多项式。

证明。设 G 是一个任意的 d-均匀超图，包含 $\ell=dn^{1/d}$ 个顶点和 n 条超边（这样的图存在，因为 $\binom{\ell}{d}\geq(\ell/d)^d=n$）。固定顶点的一个顺序 $\{1,\ldots,\ell\}$ 和边的一个顺序 $\{1,\ldots,n\}$。G 的每个顶点都将由 $\mathbb{F}_2^s$ 中的一个向量标记。我们的函数 $f\colon\mathbb{F}_2^{s\ell}\to\mathbb{F}_2^n$ 将以 G 的顶点标记作为输入，并输出 G 中对应于 n 条超边的 n 个元素：第 i 个输出是第 i 条超边中 d 个顶点标记的广义内积。我们将输入视为矩阵 $X\in\mathbb{F}_2^{s\times\ell}$，其中第 j 列 $X_j\in\mathbb{F}_2^s$ 是对应于第 j 个顶点的标记。假设超边 i 包含顶点 $\{j_1,\ldots,j_d\}$

那么第 i 个输出为 $f_i(X) = \sum_{k=1}^{s} X_{k,\ j_1} \cdots X_{k,\ j_d}$。显然， f 是一个度为 d 的函数，接下来只需证明其输出包含所有稀疏度为 $\leq s$ 的向量。为此，我们证明，对于每个稀疏度为 $\leq s$ 的向量 $y \in \mathbb{F}_2^n$，都存在一个输入，即对 G 的一种顶点标记，使得 f 输出 y。令 y 的非零元素对应于 G 中互不相同的边 i_1、 ..... 、 i_s。对于 G 中的每个顶点 j，我们将其标记 $X_j \in \mathbb{F}_2^s$ 设为满足以下条件：若 $j \in i_k$，则 $(X_j)_k = 1$；否则 $(X_j)_k = 0$。

考虑任意一条边 $i = \{j_1, \ldots, j_d\}$，以及子矩阵 $X_{j_1,\ldots,j_d}$，该子矩阵由这些通过 i 相连的顶点的标签组成。

- 若对于某个 $k \in [s]$， $i = i_k$，则 $y_i = 1$。$X_{j_1,\ldots,j_d}$ 的第 i_k 行全为 1。此外，其他每一行都至少包含一个 0。因此， $f_i(X) = 1$。
- 若对于所有 $k \in [s]$， $i \neq i_k$，则 $y_i = 0$，且 $X_{j_1,\ldots,j_d}$ 的每一行都至少包含一个 0。因此， $f_i(X) = 0$。 □

有了引理 3.1，我们现在可以证明，针对度为 2 的AVOID或NC^0_3-AVOID 问题的高效算法将意味着刚性矩阵的显式构造。

定理 3.2。对于每个常数 $1/2 \leq \delta \leq 1$，若存在一个针对伸展为 $m = 2n^{2/(1+\delta)}$ 的度为 2-AVOID 问题的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法，则可得到一个用于寻找 $(n^\delta/10, n^{\delta-1/2}/10)$-刚性矩阵的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法。

此外，对于每个 $1/2 \leq \delta \leq 1$，一个用于 NC^0_3-AVOID 、伸展为$m = n + O(n^{2/(2+\delta)})$ 的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法，将提供一个用于寻找 $(n^\delta/10, n^{\delta-1/2}/10)$-刚性矩阵的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法。

证明。令 $r = n^\delta/10$ 和 $s = n^{\delta-1/2}/10$。首先，我们在确定性多项式时间内将寻找 (r, s)-刚性矩阵的问题归约为求解关于函数 g: $\mathbb{F}_2^{4rn} \to \mathbb{F}_2^{n^2}$ 的度-2-AVOID 问题。

假设 $M \in \mathbb{F}_2^{n\times n}$不是 (r, s)-刚性的。那么， M 可以写成和 $M = Q + S$，其中 $\mathsf{rank}(Q) \leq r$，且 S 是 s-行稀疏的。此外，对于某些矩阵 L、$R^T \in \mathbb{F}_2^{n\times r}$， $Q = L \cdot R$。

我们将 g 的输入视为矩阵 L 和 R的 $2rn$ 个条目，以及 n 个 f （引理 3.1 中的二次函数）副本的输入，这些副本用于编码 S的 n 个稀疏行的条目。然后，函数 g 只需输出 $M = L \cdot R + S$的所有 n^2 个条目。注意， g 的每个输出都会计算 L 的一行与 R的一列的点积，并加上一个二次函数输出 f。因此，我们构造了一个二次函数 g，其值域包含所有非刚性矩阵。于是，在输入 g 上求解二次函数-AVOID，便可得到一个 (r, s)-刚性矩阵。 g 的输入数量为 $n' = 2rn + 2sn^{3/2} = 4rn = 2n^{1+\delta}/5$，而函数 g 的伸展至少为$m'(n') \geq 2(n')^{2/(1+\delta)}$。至此，定理第一部分的证明完成。

对于第二部分，我们使用引理 2.7 中从二次-AVOID 到 NC^0_3-AVOID 的多项式时间归约。根据上述构造，我们有一个二次函数 g: $\mathbb{F}_2^{4rn} \to \mathbb{F}_2^{n^2}$，其中每个输出位都是至多 $t = r + s \leq 2r$ 个二次单项式之和。我们应用引理 2.7，将 AVOID针对 g归约为 AVOID针对一个 NC^0_3函数 $\widehat{g}$: $\{0,1\}^{\widehat{n}} \to \{0,1\}^{\widehat{m}}$，其中 $\widehat{n} = n' + (2t-1)n^2$且 $\widehat{m} = 2tn^2$。由此可得，函数 $\widehat{g}$的伸展为 $\widehat{m}(\widehat{n}) = \widehat{n} + O(\widehat{n}^{2/(2+\delta)})$。因此，对于伸展为 $\widehat{m}(n)$ 的 NC^0_3-AVOID 问题，若存在一个算法，就能得到一个 (r, s)-刚性矩阵。 □

我们指出，在 $\delta > 1/2$的范围内，定理 3.2 将给出这样的矩阵：对于秩为 n^δ 的情形，其刚性高于所有已知的刚性矩阵构造（见FP，$\mathsf{FP}^{\mathsf{NP}}$ 和$\mathsf{FE}^{\mathsf{NP}}$）。因此，对每个 $\varepsilon > 0$，一个具有$\mathsf{FP}^{\mathsf{NP}}$ 度数 2-AVOID 且伸展为 $n^{12/11-\varepsilon}$ 的算法，或一个具有$\mathsf{FP}^{\mathsf{NP}}$ 的NC^0_3-AVOID 且伸展为 $n + n^{12/17+\varepsilon}$ 的算法，都将导出新的刚性下界。由于范围

的 $\delta = 1$ 在定理 3.2 中足以支持 Valiant 关于证明线性NC^1 电路规模超线性下界的研究纲领（见定理 2.3），因此我们得到以下推论。

推论 3.3. 若存在一个用于度为 2 的 Avoid 、伸展为 $m = 2n$ ，或用于 NC^0_3-Avoid 、伸展为 $m = n + O(n^{2/3})$ 的 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）算法，则可得到一个属于 FP（分别为 $\mathsf{FP}^{\mathsf{NP}}$）的线性函数，该函数无法由规模为 $o(n\log\log(n))$ 的线性 NC^1 电路计算。

4 用于 NC^0_k-Avoid 的算法

本节介绍用于求解具有非平凡伸展的 NC^0_k-Avoid 的多项式时间算法。具体而言，当输入电路的伸展至少为 $m \geq \Omega(n^{k-2})$ 时，我们给出一个运行时间为 $2^{O(n^{k-1}/m)} \cdot \mathrm{poly}(n)$ 的算法。首先，我们介绍 NC^0_k 电路的一个有用结构性质，该性质由以下简单的图论结果推出。

引理 4.1. 对任意常数 $c \geq 1$ 和 $k \geq 3$，每个具有 n 个顶点和 $m \geq cn^{k-2}$ 条超边的 k-均匀超图 $G = (V, E)$ 都包含一个顶点子集 $V' \subseteq V, |V'| \leq cn^{k-1}/m$ 和一个超边子集 $E' \subseteq E, |E'| \geq cn$，使得 E' 中的每条超边都包含至少 $k-2$ 个来自 V' 的顶点。此外，存在一个多项式时间算法，可以找到这样的 V' 和 E'。

证明。考虑以下二分图 H，其顶点集为 $A \sqcup B$，其中 $A = \{u_S \mid S \subseteq V,\ |S| = k-2\}$ 是由 V的 $k-2$ 元子集索引的顶点集，而 $B = \{v_e \mid e \in E\}$由 G的边索引。此外，若 $S \subseteq e$，则 H 中存在一条边 (u_S, v_e)，也就是说，若 S 中的所有顶点都包含在超边 $e \in E$中。由于每条超边 e 都包含 $\binom{k}{k-2}$ 个顶点， B 中每个顶点的度恰好为 $\binom{k}{k-2}$。因此， A 中顶点的平均度为$\frac{|B|\binom{k}{k-2}}{|A|} = \binom{k}{k-2}\binom{n}{k-2}$。令 $A' \subseteq A$ 为 A 中度最高的 t 个顶点组成的子集，并令 $N(A')$为它们在 B中的邻居。那么， A' 中顶点的总度至少为 $tm\binom{k}{k-2}\binom{n}{k-2}$。由于 B中的每个顶点的度为 $\binom{k}{k-2}$，$|N(A')| \geq tm\binom{k}{k-2}\binom{n}{k-2}\binom{k}{k-2} = tm\binom{n}{k-2}$。因此，令 $t = \frac{cn}{m}\binom{n}{k-2}$，取 ~~$V'$为~~ G 中包含在 A'所对应顶点子集之并中的 $t(k-2) = \frac{cn}{m}(k-2)\binom{n}{k-2} \leq cn^{k-1}/m$ 个顶点组成的集合，并令 $E' = N(A')$，即可完成证明。

为了找到 V' 和 E'，我们首先构造规模为多项式的图 H，然后通过在 A 中寻找度最大的 t 个顶点，找出 A' 中的顶点。现在可以直接根据 A' 构造 V'，并找到 $E' = N(A')$。 □

推论 4.2。对于任意常数 $c \geq 1$ 和 $k \geq 3$，给定一个具有 n 个输入位和 $m \geq cn^{k-2}$ 个输出位的 NC^0_k 电路 C，存在一个规模为 $|O| \geq cn$ 的输出子集 O，以及一个规模为 $|I| \leq cn^{k-1}/m$ 的输入子集 I，使得对于每个输出位 $\mathsf{C}_i \in O$，输入到 C_i 的输入位中至少有 $k-2$ 个来自 I。此外，存在一个多项式时间算法，可以找到这样的集合 I 和 O。

证明。 不失一般性，假设 C 的每个输出恰好读取 k 个输入（如果它读取 $\ell < k$ 个输入，我们就让它额外读取任意 $k-\ell$ 个输入并忽略它们）。考虑这样一个超图：每个顶点对应 n 个输入 $\{x_1, \ldots, x_n\}$ 之一，来自 C。超图的每条边对应一个输出 C_i， $e_i = \{j \mid \mathsf{C}_i \text{ 读取 } x_j\}$。现在，我们对这个超图应用 引理 4.1 ，并设定 $I = V'$ 和 $O = E'$。注意， $|I| = |V'| \leq cn^{k-1}/m$ 并且 $|O| = |E'| \geq cn$。 □

该推论找到了电路中线性数量的输出位 O，这些输出位主要依赖于少量共同的输入位 I。我们的 NC^0_k-AVOID 算法将对来自 I的输入的所有可能赋值进行"分支"。每个这样的赋值都对应于输入空间的一个仿射子空间$\mathcal{S} \subseteq \mathbb{F}_2^n$。接着，我们的算法通过固定 O 中的输出位来运行，从而使这些仿射子空间的维数之和在每一步都显著减少，最终使所有子空间变为空集。注意，根据 推论 4.2的保证，在固定输入 I后， O 中的每个输出最多依赖两个输入。因此，对于输出函数最多依赖两个输入的情形，我们需要一种高效的方法来降低包含相容输入的仿射子空间的维数。在 引理 4.3中，我们提供了这样一个子程序AFFINEREDUCE(算法 1)。

引理 4.3. 设 $\mathcal{S} \subseteq \mathbb{F}_2^n$ 是一个仿射子空间，且 $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ 是一个至多依赖两个输入的函数。算法 AFFINEREDUCE 在确定性多项式时间内找到两个仿射子空间（或空集） $\mathcal{S}_0, \mathcal{S}_1 \subseteq \mathcal{S}$，使得

(1) $\forall x \in \mathcal{S}, b \in \mathbb{F}_2$, *if* $f(x) = b$, *then* $x \in \mathcal{S}_b$;

(2) $|\mathcal{S}_0| + |\mathcal{S}_1| \leq 3|\mathcal{S}|/2$.

证明. 不失一般性，我们假设 $f(x)$ 依赖于 x_1 和 x_2（的一个子集）。我们将根据 f 的度分三种情况讨论，并且在每种情况下，我们都会找到仿射子空间（或空集） $\mathcal{S}_0, \mathcal{S}_1 \subseteq \mathcal{S}$，使得其中至少一个的维数严格小于 $\mathcal{S}$ 的维数（或者其中至少一个是空集）。这将确保 $|\mathcal{S}_0| + |\mathcal{S}_1| \leq 3|\mathcal{S}|/2$。

- 如果对于某个 $c \in \mathbb{F}_2$， $f(x) = c$ 是一个常函数，那么令 $\mathcal{S}_c = \mathcal{S}$ 和 $\mathcal{S}_{1-c} = \emptyset$。显然，$\mathcal{S}_c = \mathcal{S}$ 和 $\mathcal{S}_{1-c}$ 分别包含所有与 $f(x) = c$ 和 $f(x) = 1 - c$ 相容的点 $x \in \mathcal{S}$。

- 如果对于某些常数 $a_1, a_2, c \in \mathbb{F}_2$， $f(x) = a_1x_1 + a_2x_2 + c$是仿射函数，那么对于每个 $b \in \mathbb{F}_2$，令 H_b 为由 $a_1x_1 + a_2x_2 + c = b$定义的超平面，并令 $\mathcal{S}_b = \mathcal{S} \cap H_b$。同样， $\mathcal{S} \cap H_b$ 包含 $\mathcal{S}$中与 $f(x) = b$一致的所有输入。此外，如果 $\dim(\mathcal{S}_0) = \dim(\mathcal{S})$，那么 $\mathcal{S} \subseteq H_0$ 且 $\mathcal{S}_1 = \mathcal{S} \cap H_1 = \emptyset$。因此，要么 $\dim(\mathcal{S}_0) < \dim(\mathcal{S})$，要么 $\mathcal{S}_1 = \emptyset$。

- 如果对于某些常数 $a_1, a_2, c \in \mathbb{F}_2$， $f(x) = (x_1 + a_1)(x_2 + a_2) + c$是二次函数，那么令 $\mathcal{H}$ 为由 $\mathcal{H} = \{x \in \mathbb{F}_2^n \mid x_1 = 1 + a_1, x_2 = 1 + a_2\}$ 定义的仿射子空间。考虑仿射子空间 $\mathcal{S}_{1-c} = \mathcal{S} \cap \mathcal{H}$，它包含所有满足 $f(x) = 1 - c$ 的点 $x \in \mathcal{S}$。– 如果 $\dim(\mathcal{S}_{1-c}) < \dim(\mathcal{S})$，那么取 $\mathcal{S}_c = \mathcal{S}$即可完成。– 如果 $\dim(\mathcal{S}_{1-c}) = \dim(\mathcal{S})$，那么 $\mathcal{S} \subseteq \mathcal{H}$。于是， $\mathcal{S}$中的每个点都满足 $f(x) = 1 - c$，因此，令 $\mathcal{S}_{1-c} = \mathcal{S}$ 和 $\mathcal{S}_c = \emptyset$ 即可完成构造。

在每种情况下，要么是 $|\mathcal{S}_0| \leq \frac{|\mathcal{S}|}{2}$，要么是 $|\mathcal{S}_1| \leq \frac{|\mathcal{S}|}{2}$。因此， $|\mathcal{S}_0| + |\mathcal{S}_1| \leq 3|\mathcal{S}|/2$。

AFFINEREDUCE唯一进行的计算是求出明确给定的仿射子空间的维数，这可以在多项式时间内完成。□

一个简单的应用 AFFINEREDUCE恢复了NC^0_2-AVOID中 [GLW22]的多项式时间算法。

算法 1 AffineReduce($\mathcal{S}, f$)

输入：仿射子空间 $\mathcal{S} \subseteq \mathbb{F}_2^n$, f： $\mathbb{F}_2^n \to \mathbb{F}_2$ 可能仅依赖于 x_1 和 x_2

输出： $\mathcal{S}_0, \mathcal{S}_1 \subseteq \mathcal{S}$

如果 $f(x) = c$，则

返回 $\mathcal{S}_c = \mathcal{S}$ 和 $\mathcal{S}_{1-c} = \emptyset$

如果 $f(x) = a_1x_1 + a_2x_2 + c$，则

对于 $b \in \mathbb{F}_2$，令 $H_b = \{x \in \mathbb{F}_2^n : a_1x_1 + a_2x_2 + c = b\}$

返回 $\mathcal{S}_0 = \mathcal{S} \cap H_0$ 和 $\mathcal{S}_1 = \mathcal{S} \cap H_1$

如果 $f(x) = (x_1 + a_1)(x_2 + a_2) + c$，则

令 $\mathcal{H} = \{x \in \mathbb{F}_2^n : x_1 = 1 + a_1, x_2 = 1 + a_2\}$

令 $\mathcal{S}_{1-c} = \mathcal{S} \cap \mathcal{H}$

如果 $\dim(\mathcal{S}_{1-c}) < \dim(\mathcal{S})$，则

返回 $\mathcal{S}_{1-c}$ 和 $\mathcal{S}_c = \mathcal{S}$

else

返回 $\mathcal{S}_{1-c}$ 和 $\mathcal{S}_c = \emptyset$

推论 4.4：存在一个确定性多项式时间算法，给定一个 NC_2^0 电路 C: $\{0,1\}^n \to \{0,1\}^m$, $m \geq n+1$，可以找到一个元素 $y \in \{0,1\}^m, y \notin \text{Range(C)}$。

证明。在第 $1 \leq i \leq n+1$ 次迭代中，我们的算法将固定第 i 个输出位 y_i 的值。算法还维护一个仿射子空间 $\mathcal{S} \subseteq \mathbb{F}_2^n$，其中包含所有与 $y_1, \ldots, y_i$ 的部分输出赋值一致的输入 $x \in \mathbb{F}_2^n$。根据引理 4.3，存在一个赋值 $y_i = b$，使得要么 $\mathcal{S}_b$ 中没有输入与 y 一致，要么 $\mathcal{S} = \mathcal{S}_b$ 的维数至少减少一。在前一种情况下，我们已经找到了所需的输出 y（只需将 y 中尚未赋值的位设为任意值即可）。否则，在固定前 n 个输出后，我们有 $\dim(\mathcal{S}) = 0$，也就是说，对于某个 $x \in \mathbb{F}_2^n$，有 $S = \{x\}$。令 $b \in \{0,1\}$ 为 C(x) 的第 $(n+1)$ 个输出位的值。接着，设定 $y_{n+1} = 1 - b$ 即可得到所需的输出 y。该算法运行时间为多项式时间，因为它最多调用 AffineReduce 共 n 次，并调用 C(x) 一次。

□

最后，有了引理 4.3的帮助，我们准备好给出用于NC_k^0-Avoid的主要算法。

定理 4.5. 给定一个 NC_k^0 电路 C： $\{0,1\}^n \to \{0,1\}^m$，其中 $m \geq 3n^{k-2}$，算法 Sub-spaceUnion 会在确定性时间 $2^{O(n^{k-1}/m)} \cdot \text{poly}(n)$ 内找到一个元素 $y \in \{0,1\}^m, y \notin \text{Range(C)}$。

证明。首先，我们对电路 C 应用推论 4.2，其中 $c = 3$，并在多项式时间内选取一组输入子集 $I = \{x_1, \ldots, x_t\}$ 和一组输出 $O = \{y_1, y_2, \ldots, y_{3n}\}$，用于 $t \leq 3n^{k-1}/m$。这确保每个 y_i 在 I之外至多有两个输入。对于输入 I的每种 2^t 赋值，我们考虑一个电路，其中这些 t 输入的值是固定的。具体来说，对于$j \in \{0, \ldots, 2^t - 1\}$，我们将 I中的输入固定为 j的二进制表示中的位。然后，将电路 C 限制为输出 $y_1, \ldots, y_{3n}$ 以及为这些输出提供输入的所有输入，从而得到一个电路 Cj，其中每个输出至多依赖于 2 个输入。我们将找到一个 Cj 不会输出的值 $y \in \{0,1\}^{3n}$，这就为原始的 NC_k^0-Avoid 实例提供了一个解。

我们的算法将保持以下不变量。在算法的第 i 次迭代中，固定输出 y_1、……、y_i 的值后，我们保持 $\mathcal{U} = \bigcup_{j=0}^{2^t-1} \mathcal{U}_j$ 为 2^t 个仿射子空间的不交并，使得所有与 y_1、……、y_i 一致的输入 $x \in \mathbb{F}_2^n$ 都属于 $\mathcal{U}$（而且 $\mathcal{U}$ 也可能包含与 y_1、……、y_i 不一致的点）。

在算法开始时，对于每个 $0 \le j < 2^t$，我们令 $\mathcal{U}_j$ 为这样一个仿射子空间：其中 I 中的输入被固定为 j 的二进制表示中的位。于是，$\mathcal{U} = \bigcup_j \mathcal{U}_j = \mathbb{F}_2^n$ 就是与我们的初始空部分赋值一致的所有输入的集合。

在每一步 i，我们都会说明如何找到一个常数 $b \in \{0,1\}$，使得固定 $y_i = b$ 后，我们的不交并 $|\mathcal{U}|$ 的规模缩小为原来的 $4/3$。因此，对来自 O 的 $3n$ 个输出重复此过程后，我们将得到一个空的 $\mathcal{U}$，而构造出的部分赋值将为我们提供 NC^0_k-AVOID 问题的一个解。

在算法的第 i 次迭代中，我们已有输出 y_1、$\dots y_{i-1}$ 的取值固定，现在要固定 y_i 的取值。我们有两种选择：要么令 $y_i = 0$，要么令 $y_i = 1$。根据 引理 4.3，我们得到两个仿射子空间（或空集）$\mathcal{U}_{j,0}, \mathcal{U}_{j,1} \subseteq \mathcal{U}_j$，分别包含所有映射到 $y_i = 0$ 和 $y_i = 1$ 的输入 $x \in \mathcal{U}_j$。此外，引理 4.3 保证 $|\mathcal{U}_{j,0}| + |\mathcal{U}_{j,1}| \le 3|\mathcal{U}_j|/2$。对所有 $0 \le j < 2^t$ 求和，得到

$$\sum_j |\mathcal{U}_{j,0}| + \sum_j |\mathcal{U}_{j,1}| \le \sum_j 3|\mathcal{U}_j|/2 = 3|\mathcal{U}|/2\,.$$

令 $b \in \{0,1\}$ 为使 $\sum_j |\mathcal{U}_{j,b}|$ 取值最小的值。特别地，我们有 $\sum_j |\mathcal{U}_{j,b}| \le 3|\mathcal{U}|/4$。因此，设置 $y_i = b$ 至少能将 $\mathcal{U}$ 的规模缩小 $4/3$ 倍。重复此过程 $\log_{4/3}(2^n) + 1 \le 3n$ 次后，将得到对输出位 O 的一个部分赋值，且没有任何输入会映射到该赋值。

算法 SUBSPACEUNION 维护 2^t 个 $\mathbb{F}_2^n$ 的仿射子空间，计算它们的维数，并多次调用确定性多项式时间 AFFINEREDUCE 过程。因此，该算法的运行时间为 $2^t \cdot \mathrm{poly}(n) = 2^{O(n^{k-1}/m)} \cdot \mathrm{poly}(n)$。□

算法 2 SUBSPACEUNION(C)

输入：NC^0_k 电路 $C : \{0,1\}^n \to \{0,1\}^m$，其中 $m \ge 3n^{k-2}$

输出：$y \in \{0,1\}^m$，$y \notin \mathrm{Range}(C)$

查找 $x_1, \dots, x_t$ 和 $y_1, \dots, y_{3n}$ 通过 推论 4.2 对于 $t \le 3n^{k-1}/m$

对于 $0 \le j < 2^t$，设置 $\mathcal{U}_j = \{x \in \{0,1\}^n : \sum_{i=1}^t x_i 2^{i-1} = j\}$

对于 i=1 到 3n，执行

在 y_i 处查找函数 f

对于 $0 \le j < 2^t$，设置 $\mathcal{U}_{j,0}, \mathcal{U}_{j,1} \leftarrow \mathrm{AFFINEREDUCE}(\mathcal{U}_j, f)$

找到使 $b \in \{0,1\}$ 最小化的 $\sum_j |\mathcal{U}_{j,b}|$

设置 $y_i = b$

对于 $0 \le j < 2^t$，设置 $\mathcal{U}_j = \mathcal{U}_{j,b}$

设置所有剩余的 $y_k = 0$

返回 y

我们以一个推论结束本节，该推论指出，对于伸展 m 的某些非平凡取值， SUBSPACEUNION 能够高效地求解 NC^0_k-AVOID。

推论 4.6. 对任意常数 $k \ge 3$ 和 $\varepsilon > 0$，算法 SUBSPACEUNION 可分别在确定性多项式时间和确定性亚指数时间 $2^{O(n^{1-\varepsilon})}$ 内，针对 $m \ge n^{k-1}/\log(n)$ 和 $m \ge n^{k-2+\varepsilon}$ 求解具有 n 个输入和 m 个输出的 NC^0_k-AVOID 。

致谢
我们感谢 Justin Thaler、Sam King 以及匿名审稿人对本文提出的宝贵意见。

参考文献

[AB09] Sanjeev Arora 和 Boaz Barak。《计算复杂性：一种现代方法》。剑桥大学出版社，2009年。2, 7[AC19] Josh Alman 和 Lijie Chen。使用 NP 预言机构造刚性矩阵的高效方法。载于 FOCS，2019年。3[AIK06] Benny Applebaum、Yuval Ishai 和 Eyal Kushilevitz。NC^0中的密码学。《SIAM 计算杂志》，36(4):845–888，2006年。1, 3, 6, 8, 9[BHPT20] Amey Bhangale、Prahladh Harsha、Orr Paradise 和 Avishay Tal。来自矩形 PCP 的刚性矩阵。载于 FOCS，2020年。3[Cha94] Aleksandr V. Chashkin。论布尔矩阵、图及其对应布尔函数的复杂性。《离散数学与应用》，4(3):229–257，1994年。2[CHLR23] Yeyuan Chen、Yizhi Huang、Jiatu Li 和 Hanlin Ren。通过满足对算法实现值域规避、远点和困难的部分真值表。载于 STOC，2023年。1[Fri93] Joel Friedman。关于矩阵刚性的一则注记。Combinatorica，13(2):235–239，1993年。3[GLW22] Venkatesan Guruswami、Xin Lyu 和 Xiuhan Wang。低深度电路的值域规避及其与伪随机性的联系。载于 RANDOM，2022年。1, 2, 3, 6, 13[GT16] Oded Goldreich 和 Avishay Tal。随机托普利茨矩阵的矩阵刚性。载于 STOC，2016年。3[IK00] Yuval Ishai 和 Eyal Kushilevitz。随机化多项式：一种具有轮次高效安全计算应用的新表示。载于 FOCS，2000年。1, 3[IK02] Yuval Ishai 和 Eyal Kushilevitz。通过完美随机化多项式实现完美常数轮安全计算。载于 ICALP，2002年。1, 3[ILW23] Rahul Ilango、Jiatu Li 和 Ryan Williams。不可区分混淆、值域规避与有界算术。载于 STOC，2023年。1[IP99] Russell Impagliazzo 和 Ramamohan Paturi。 k-布尔可满足性问题的复杂性。载于 CCC，1999年。1[IPZ98] Russell Impagliazzo、Ramamohan Paturi 和 Francis Zane。哪些问题具有强指数复杂度？载于 FOCS，1998年。1[KKMP21] Robert Kleinberg、Oliver Korten、Daniel Mitropolsky 和 Christos Papadimitriou。多项式层次中的全函数。载于 ITCS，2021年。1

[Kor21] Oliver Korten。最困难的显式构造。载于 FOCS，2021年。1, 2[LY22] Jiatu Li 和 Tianqi Yang。 $3.1n - o(n)$ 显式函数的电路下界。载于 STOC，2022年。2[PR94] Pavel Pudl´ak 和 Vojtech R¨odl。复杂性理论中的一些组合代数问题。《离散数学》，1(136):253–279，1994年。3[RSW22] Hanlin Ren、Rahul Santhanam 和 Zhikun Wang。论电路的值域回避问题。载于 FOCS，2022年。1, 2, 3, 9, 10[Sha49] 克劳德·E·香农。双端开关电路的综合。《贝尔系统技术期刊》，28:59–98，1949年。2[SSS97] Mohammad A. Shokrollahi、Daniel A. Spielman 和 Volker Stemann。关于矩阵刚性的一则注记。《信息处理快报》，64(6):283–285，1997年。3[Val77] Leslie G. Valiant。低层复杂性中的图论论证。载于 MFCS，1977年。2, 3, 7, 8

A NC^0_3-Avoid 的另一种算法

定理 A.1. 给定一个 NC^0_3 电路 C：$\{0,1\}^n \to \{0,1\}^m$，其中 $m \geq \frac{1}{3}\binom{n}{2} + 2n$，算法OneSubspace可在确定性多项式时间内找到一个元素 $y \in \{0,1\}^m, y \notin$ Range(C)。

证明。该算法维护一个关于输入的仿射子空间 $\mathcal{S} \subseteq \mathbb{F}_2^n$，以及一个部分输出赋值 $y \in \{0,1,*\}^m$，使得 $\mathcal{S}$ 包含所有与 y一致的输入 $x \in \mathbb{F}_2^n$。初始时，$y = (*, \ldots, *)$ 且 $\mathcal{S} = \mathbb{F}_2^n$。在每次迭代中，OneSubspace至多赋值两个输出，并使 $\mathcal{S}$ 的维数至少减少 1。经过 n 步后，$\mathcal{S}$ 的维数必为 0。然后，算法再赋值一个输出位，并以元素 $y \notin$Range(C) 结束。

现在，我们只需论证该算法能够在每次迭代中降低 $\mathcal{S}$ 的维数，并且每一步都能在多项式时间内完成。

首先，如果存在一个输出 y_1 至多依赖于 2 个输入 x_1，x_2，令 f 为该输出处计算的函数：$y_1 = f(x_1, x_2)$。根据 引理 4.3，AffineReduce($\mathcal{S}$，f) 输出一个维数更低的仿射子空间 $\mathcal{S}_b$ $\dim(\mathcal{S}_b) < \dim(\mathcal{S}$，其中包含所有与 $y_1 = b$一致的输入。因此，接下来我们假设每个输出都恰好依赖于 3 个输入。

由于每次迭代中我们至多固定输出的 2 个比特，未赋值的输出数量m 始终大于 $\frac{1}{3}\binom{n}{2}$。因此，存在一对输出 y_1、y_2，它们都依赖于同一对输入 x_2、x_3。由于每个输出依赖于三对输入，而这类输入对的数量为 $\binom{n}{2} < 3m$，必定存在一对输入同时连接到两个输出。令 x_1、x_4 分别为连接到输出 y_1、y_2 的其余输入：$y_1 = f_1(x_1, x_2, x_3)$ $y_2 = f_2(x_2, x_3, x_4)$。

(y_1, y_2) 有 4 种可能取值，而出现在 $\mathcal{S}$ 中的输入位 (x_1, x_2, x_3, x_4) 至多有 16 种可能取值。因此，存在一对常数 $(b_1, b_2) \in \{0,1\}^2$，使得至多有 4 种不同的赋值 $A \subseteq \{0,1\}^4$，它们与部分赋值 $(y_1, y_2) = (b_1, b_2)$ 一致，并且赋给 (x_1, x_2, x_3, x_4)。由于 $|A| \leq 4$，$\mathbb{F}_2^4$ 中存在一个 3 维仿射子空间，包含来自 A 的所有点。因此，存在一个超平面 $\mathcal{H} \subseteq \mathbb{F}_2^4$ 来定义这个 3 维仿射子空间。将 $\mathcal{H}$ 扩展到所有 n 输入后，可得到一个仿射子空间 $\mathcal{H}' = \{x \in \mathbb{F}_2^n : (x_1, x_2, x_3, x_4) \in \mathcal{H}\} \subseteq \mathbb{F}_2^n$，其中包含 $\mathbb{F}_2^n$ 中所有与部分赋值 $(y_1, y_2) = (b_1, b_2)$ 一致的输入。

如果 $\mathcal{S} \not\subseteq \mathcal{H}'$，那么设置 $(y_1, y_2) = (b_1, b_2)$ $\mathcal{S} = \mathcal{S} \cap \mathcal{H}'$ 会降低仿射子空间 $\mathcal{S}$ 的维数。下文假设 $\mathcal{S} \subseteq \mathcal{H}'$。

- 假设存在 $(c_1, c_2) \in \{0,1\}^2$，使得对于 $\mathcal{H}'$ 中的所有点 (x_1, x_2, x_3, x_4)，都有 $(f_1(x_1, x_2, x_3)$ $f_2(x_2, x_3, x_4)) \neq (c_1, c_2)$。那么我们可以设置 $(y_1, y_2) = (c_1, c_2)$，并且 $\mathcal{S} = \emptyset$，因为 $\mathcal{S} \subseteq \mathcal{H}'$ 中没有点能输出 (c_1, c_2)。在这种情况下，我们找到了一个 $y \notin$ Range(C)。
- 如果不存在这样的赋值，那么由于 $|\mathcal{H}| = 8$，必定存在一个赋值 $(c_1, c_2) \in \{0,1\}^2$，使得 $\mathcal{H}$ 中至多有两个点与 $(y_1, y_2) = (c_1, c_2)$ 相容。这至多两个点构成一个 0 维或 1维仿射子空间 $\mathcal{U} \subseteq \mathbb{F}_2^4$，我们将其扩展到所有$n$输入 $\mathcal{U}' = \{x \in \mathbb{F}_2^n : (x_1, x_2, x_3, x_4) \in \mathcal{U}\} \subseteq \mathbb{F}_2^n$。

  – 若 $\mathcal{S} \not\subseteq \mathcal{U}'$，我们可以设置 $(y_1, y_2) = (c_1, c_2)$ 并 $\mathcal{S} = \mathcal{S} \cap \mathcal{U}'$，从而降低 $\mathcal{S}$ 的维数。– 否则，$\mathcal{S} \subseteq \mathcal{U}'$ 中的所有输入都具有 $(y_1, y_2) = (c_1, c_2)$，我们可以设置 $(y_1, y_2) = (1 - c_1, c_2)$ 以得到 $\mathcal{S} = \emptyset$。

该算法执行 n 次迭代，每次迭代都在多项式时间内计算常数个显式给出的仿射子空间的维数。 □

算法 3 OneSubspace(C)

输入：$\mathbf{NC}^0_3$ 电路 $\mathbf{C}: \{0,1\}^n \to \{0,1\}^m$，其中 $m \geq \frac{1}{3}\binom{n}{2} + 2n$

输出：$y \in \{0,1\}^m$，$y \notin$ Range(C)

令 $\mathcal{S} = \mathbb{F}_2^n$

对于 i=1 到 n，执行

如果 $\mathcal{S} = \emptyset$，则

设置所有剩余的 $y_k = 0$

返回 y

如果 $\exists y_1, x_1, x_2$ 满足 $y_1 = f(x_1, x_2)$，则

$\mathcal{S}_0, \mathcal{S}_1 =$ AffineReduce$(\mathcal{S}, f)$

找到使 $|\mathcal{S}_b|$ 最小化的 $b \in \{0,1\}$，设置 $y_1 = b, \mathcal{S} = \mathcal{S}_b$

else

找到满足以下条件的 $y_1, y_2, x_1, x_2, x_3, x_4$：$y_1 = f_1(x_1, x_2, x_3)$，$y_2 = f_2(x_2, x_3, x_4)$

求 $b_1, b_2 \in \{0,1\}$，使得

$A = \{(x_1, x_2, x_3, x_4) \in \mathbb{F}_2^4 : (f_1(x_1, x_2, x_3)\ f_2(x_2, x_3, x_4)) = (b_1, b_2)\}$ 和 $|A| \leq 4$

设 $\mathcal{H} \subseteq \mathbb{F}_2^4$ 为由 A中的点定义的超平面

设 $\mathcal{H}' = \{x \in \mathbb{F}_2^n : (x_1, x_2, x_3, x_4) \in \mathcal{H}\}$

若 $\mathcal{S} \nsubseteq \mathcal{H}$，则

设 $(y_1, y_2) = (b_1, b_2)$，$\mathcal{S} = \mathcal{S} \cap \mathcal{H}'$

else

如果 $\exists (c_1, c_2)$，使得 $\forall (x_1, x_2, x_3, x_4) \in \mathcal{H}\ (f_1(x_1, x_2, x_3)\ f_2(x_2, x_3, x_4)) \neq (c_1, c_2)$，那么

令 $\mathcal{S} = \emptyset$，$(y_1, y_2) = (c_1, c_2)$

else

求 (c_1, c_2)，使得

$\mathcal{U} = \{(x_1, x_2, x_3, x_4) \in \mathcal{H} : (f_1(x_1, x_2, x_3)\ f_2(x_1, x_2, x_3)) = (c_1, c_2)\}, |\mathcal{U}| \leq 2$

设 $\mathcal{U}' = \{x \in \mathbb{F}_2^n | (x_1, x_2, x_3, x_4) \in \mathcal{U}\}$

如果 $\mathcal{S} \nsubseteq \mathcal{U}'$，那么

集合 $\mathcal{S} = S \cap \mathcal{U}'$，$(y_1, y_2) = (c_1, c_2)$

else

$\mathcal{S} = \emptyset, (y_1, y_2) = (1 - c_1, c_2)$

返回 y