用于包含亚稳态加法的编码

Johannes Bund*、Christoph Lenzen†、Moti Medina‡

*巴黎-萨克雷大学 ´、法国国家科学研究中心、巴黎-萨克雷高等师范学院、形式化方法实验室91940 吉夫叙尔伊韦特，法国 jobund@lmf.cnrs.fr

†*CISPA* 亥姆霍兹信息安全中心66123 萨尔布吕肯¨，德国 lenzen@cispa.de

‡亚历山大·科夫金工程学院，巴伊兰大学5290002 拉马特甘，以色列 moti.medina@biu.ac.il

arXiv:2602.06467v1 [cs.IT] 6 Feb 2026

摘要—

我们研究不确定性下加法这一基本任务，即加数以数值区间而非单个值表示。此类不确定性的一个潜在来源，是获取模拟值的离散值测量时可能出现的亚稳态。自然，不稳定位会影响门级计算，进而影响电路级计算。对于此类加法，采用二进制编码会产生不精确度被放大的和。

因此，挑战在于设计一种编码，使不稳定位导致的不精确度不会被放大。我们称这类编码为可恢复。虽然一元编码很容易满足这一要求，但目前尚无适用的高码率编码。在这项工作中，对于给定的加数合并不确定性上界，我们证明了保持与可恢复编码码率的上界。随后，我们设计了一种渐近最优码，可保持加数的合并不确定性。

随后，我们讨论如何为这种编码构造加法器。该方法可与任何已知或未来的输入亚稳态容限构造方法结合使用。我们推测，基于现有技术进行精心设计，有望显著降低延迟。

索引词—区间加法、亚稳态遏制、可恢复码

1 引言

乍看之下，对带有不确定性的输入进行加法很简单：将输入表示为区间后，只需分别将这些输入区间的下界和上界相加，即可得到一个表示可能输出的区间。然而，当输入的不稳定性干扰计算时，情况就会大不相同。例如，亚稳态就可能造成此类干扰。亚稳态信号会破坏输入的二进制表示，使得输入区间的下界和上界无法表示。

本文已被 *IEEE Transactions on Computers* 接收发表。资助：本研究得到以色列科学基金会第 *867/19* 和 *554/23* 号资助。本研究还部分获得法国国家研究署通过 *DREAMY* 项目（*ANR-21-CE48-0003*）提供的资助。*Johannes Bund* 在参与本文技术部分的研究期间，任职于 *CISPA* 亥姆霍兹信息安全中心和巴伊兰大学。*Christoph Lenzen* 在本研究的部分阶段任职于阿尔托大学。

表1 使用串行加法器的标准 VHDL 实现对二进制码字进行加法，输入信号为(X，Y)，输出信号为（和）。亚稳态信号用值 x表示。

X	0001 1001 ({25})	0001 101x ({26, 27})
Y	0010 0101 ({37})	0010 0101 ({37})
SUM	0011 1110 ({62})	0xxx xxxx ({0, . . . , 127})

先前的研究结果表明，亚稳态容纳电路可以减轻亚稳态信号造成的信息丢失 [1]。如果组合电路对于带有亚稳态位的输入，不会偏离其所能作出的最佳响应，我们就称该电路为 亚稳态容纳 *(mc)*（正式定义见第 2.1 节）。在 [2], 中，我们发现，面对亚稳态输入时，编码方式的选择至关重要。在这项工作中，我们表明，通过精心设计编码和亚稳态容纳加法电路，可以保留输入区间隐含的下界和上界，从而防止信息丢失。

在以下示例中，我们表明，面对亚稳态位时编码至关重要，也就是说，即使使用亚稳态包含二进制加法器，信息仍会丢失。

不确定性下的加法：第一个示例

由于亚稳态造成的信息丢失，一个经典例子是在标准二进制编码下进行加法。表1中，我们模拟了一个用 VHDL 实现的经典组合加法器。对于稳定输入（25 和37的二进制编码），电路会产生正确的输出（62的二进制编码）。然而，对于单个亚稳态输入位，电路会放大不确定性。在26和27的编码之间切换时，最低有效位可能变为亚稳态。此例会导致7 个8 输出位变为亚稳态。将每个x 解析为0 或1，可以得到从0 到127的任意码字。

让我们考虑一种使用格雷码的朴素方案，其中每个码字至多包含一个亚稳态位，表示大小为2的区间。这些情况出现在使用时间数字转换器（TDCs）生成加数时，因为它们可以

表2
VHDL 中混合码字的加法，采用亚稳态包含第 7 节加法方案的实现。码字的长度为8，其中 $n = 5$ 和 $k = 3$。亚稳态信号表示为值 X。

X	00101 1X0 ($\{25, 26\}$)	01X10 X00 ($\{47, 48, 49\}$)
Y	01101 011 ($\{37\}$)	00111 011 ($\{21\}$)
SUM	01000 00X ($\{62, 63\}$)	11001 XX1 ($\{68, 69, 70\}$)

被设计为确定性地确保格雷码输出 [3]中至多包含一个亚稳态位。在这种情况下，将每个仅含一个亚稳态位的两个或更多值相加，必然会累积相应的不确定性。反过来，不确定性会再次增加，例如，将0X1 与11X 相加，对应于 $\{1, 2\}$和 $\{4, 5\}$，得到1XX，对应于 $\{4, \ldots, 7\}$，而不是所需的 $\{5, 6, 7\}$。

非正式地说，我们使用术语 精度 来描述编码值的确定程度：它可能表示的连续整数值越少，其精度就越高。我们通过不精确度 $p \geq 0$来参数化有限精度：不精确度为 p 的码字解析后最多可能对应 $p + 1$个连续整数值。在接下来的章节中，我们将逐步使这一概念更加形式化。

我们的主要研究问题

可以说，我们克服的主要挑战在于，通过引入“精度”这一概念，并将其与编码的码率进行比较，从而使不确定性下的加法这一概念形式化。直观而言，输入的精度就是它所表示的数的值域；例如，值域 $\{7, 8, 9\}$ 的精度为 2。合适的编码必须保留这一精度，也就是说，在亚稳态消除后，我们仍能可靠地从这个值域中恢复出一个数。

在先前的工作中（参见 [4] 和 [2]），我们发现，可容忍的不精确度大小受到输入编码的限制。在布尔世界中，编码的选择可能会影响某项运算的复杂度，但不会影响其精度。然而，面对亚稳态时，编码可能会导致不必要的信息丢失（参见表1）。在经典设计中，其结果是同步器必须确保亚稳态不会传播到加法电路中。

注意，一种“低效”的保精度编码以及相应的加法电路是可以实现的。为此，考虑基于一元温度计码与mc排序电路相结合的简单方案。一元温度计码能够保持精度，而对输入字各比特的并集进行mc排序即可实现加法。最优mc排序电路是已知的 [4]。因此，这种朴素的mc方案在 n上是高效的。然而，这具有误导性：使用 n 比特时，它只能对值域 $\{0, \ldots, n\}$中的数进行加法，而传统加法器在指数级更小的编码上工作，即使用相同数量的比特处理值域 $\{0, \ldots, 2^n - 1\}$。

事实上，正如我们稍后将证明的（参见定理4.2），若不对亚稳态位的数量加以限制，就无法改善这一状况，因为保持精度 n的编码需要 n 比特。相反，格雷码具有最优码率，并且能够处理对应于相邻数值的输入，例如 $\{2, 3\}$ 或 $\{62, 63\}$（参见定理5.10）。由此引出如下问题。

> 是否存在一种高码率编码，能够（*i*）在对包含数量有界的亚稳态位的码字进行加法时保持精度，并且（*ii*）允许通过电路实现高效运算？

1.1 我们的贡献

在本文中，我们建立以下结果。

保持与可恢复编码的形式化。我们通过引入编码的属性保持性和可恢复性，将不确定性下的加法概念形式化。非正式地说，在加法的语境下，如果通过解析 mc 加法电路输出中的亚稳态，可以得到的码字值域与加数不确定性的和相匹配，则称该编码具有保持性。此外，如果能够确定一个从所有字符串到码字的映射，使得在解析亚稳态后应用该映射所得到的是上述值域中的码字，则称具有保持性的编码是可恢复的。

可恢复码码率的上界。我们证明，保持 ℓ 精度的 n 位编码的码率以 $\mathcal{O}(2^{n-\ell}\ell/n)$ 为上界，其中 $\ell \in \mathbb{N}$ 是不精确度的一个界。

一种渐近最优码。 我们给出一种 $(n + \ell)$ 位编码，其精度为 ℓ。事实上，该编码是 ℓ 保持且 $\lceil \ell/2 \rceil$ 可恢复的，这表明大小为 ℓ 的区间可由一个码字表示，并且我们可以从亚稳态信号中恢复大小为 $\lceil \ell/2 \rceil$ 的区间（见定义 3.3 和定义 3.4）。

面向我们编码的 MC 加法电路。 对于该 ℓ 精度编码，我们给出一个规模为 $\mathcal{O}(n+\ell)$、深度为 $\mathcal{O}(\log n+\log \ell)$ 的加法电路。然而，该电路仅对稳定码字实现加法。还需要两个步骤：（i）将（稳定的）非码字映射为码字，这很直接；（ii）构造一个在稳定输入上具有相同行为的 mc电路。利用 [5] 中的构造，可得到一个规模为 $(n + \ell)^{\mathcal{O}(\ell)}$、深度为 $\mathcal{O}(\ell \log\, n)$ 的纯组合 mc 电路。借助所谓的屏蔽寄存器， [1] 中的构造可得到一个规模为 $\mathcal{O}(\ell($ $n + \ell))$、深度为 $\mathcal{O}(\log n + \log\, \ell)$ 的电路。

以下示例展示了来自我们编码的码字的 mc 加法器如何不会损失精度。

不确定性下的加法：第二个示例

在表2中，我们展示了来自我们的“混合”编码的码字相加（参见第 6 节）。在表2的示例中，我们使用一种8位混合编码，其中包含一个5位二进制反射格雷码（BRGC）部分和一个3位一元变体。我们从全零字00000000 开始，并按如下方式定义递增计数序列。当一元部分既不是全0 ，也不是全1时，向上计数只改变一元部分，遵循以下模式

$$000 \to 100 \to 110 \to 111 \to 011 \to 001 \to 000,$$

从而相邻码字仅在一个一元位上有所不同。每当一元部分为 111 或000时，下一次向上计数只翻转BRGC部分（在5位BRGC中前进一个步长），并保持一元部分不变；此后，一元模式继续如上所述进行。这就在8位码字上形成了全局格雷序列。

对于表2中出现的具体整数值，相应的混合码字（BRGC 和一元部分之间以空格分隔）为：

$$\begin{aligned} &21 \mapsto 00111\ 011, 25 \mapsto 00101\ 100, 26 \mapsto 00101\ 110,\\ &37 \mapsto 01101\ 011, 47 \mapsto 01110\ 000, 48 \mapsto 01010\ 000,\\ &49 \mapsto 01010\ 100, 62 \mapsto 01000\ 001, 63 \mapsto 01000\ 000,\\ &68 \mapsto 11001\ 111, 69 \mapsto 11001\ 011, 70 \mapsto 11001\ 001. \end{aligned}$$

在这里，BRGC部分确定粗略位置，而一元部分则在较小值域内对其进行细化；整数区间

是通过混合编码对这些码字进行解释而得到的，如表2所示。

亚稳态信号用值 x 表示。所模拟的电路是对第 7 节所介绍电路的 VHDL 亚稳态包含实现。加入一个可能表示25 或26 的输入，并对37 进行编码，正确运算后得到的码字可能表示62 或63。我们的混合码以 k 为参数。对于 $k=3$，该编码允许的不精确度最高为2，即可以正确相加的码字可表示3个连续数字。例如，将表示47、48和49 的码字与21的编码相加，正确运算后得到的码字表示68、69和70。

1.2 应对亚稳态：从同步器到亚稳态包含电路

亚稳态是数字电路中的一种异常工作模式。任何时钟域跨越、模数转换或时间数字转换都不可避免地存在亚稳态信号的风险 [6]。这种现象涉及处于不稳定状态的双稳态存储元件。亚稳态存储元件的输出可能停留在逻辑0 与逻辑1之间的某个电压电平，也可能发生振荡，或出现延迟跳变。亚稳态信号可能违反门电路和存储元件的时序约束与输入规范。因此，它们会在数字电路中传播，并可能影响其他信号。电路发生故障的概率通常用平均故障间隔时间（MTBF）表示。

亚稳态信号本质上会造成软错误，并可能导致严重后果。在任意时刻，亚稳态信号都可能被读作逻辑0 或逻辑1。因此，亚稳态会导致人们无法确定编码后的值是0 还是1。任何数字电路都无法以确定性的方式避免、解析或检测亚稳态 [7]。

超大规模集成电路技术的发展趋势进一步加剧了存储元件发生亚稳态扰动的风险。复杂度增加、时钟频率提高、未同步时钟域增多、工作电压降低以及晶体管技术尺寸缩小等因素，都会降低平均故障间隔时间 [8], [9]。

降低亚稳态传播风险的一种常见方法是使用同步器 [10]。同步器是一种器件，能够促使亚稳态信号解析为逻辑0或逻辑1。从本质上说，同步器以时间换取可靠性。分配给亚稳态信号解析的时间越长，由亚稳态引起软错误的概率就越低。尽管可能实现很长的平均故障间隔时间，但错误永远无法以确定性的方式避免。通过增加同步器来应对风险会增加同步延迟，进而增加延迟。因此，理想的做法是采用不会带来这些缺点的技术来应对亚稳态扰动的威胁。

有趣的是，对开关网络的研究表明，精心设计的电路可以确定性地 避免软错误 [11]，似乎违背了 Marino 的不可能性结果 [7]。关键在于，电路可以设计成在不放大亚稳态信号所引入的不确定性的情况下加以容纳 [1]；我们将此类电路称为亚稳态包含（*mc*）电路。mc电路能够确定性地最大限度地减少亚稳态信号的传播。也就是说，在假设亚稳态按最坏情况传播的前提下，它能产生精度最高的输出。

尽管在某些应用中可以避免 [3], [4], ，但一般而言，构建 mc 电路会导致电路规模呈指数级膨胀 [5]。然而，借助所谓的屏蔽存储元件，可将这一开销降至线性；这些元件将内部亚稳态转化为延迟的输出转换 [1]。近来，已设计出实用的高质量屏蔽锁存器 [12]。

1.3 应用：容错时钟同步

亚稳态包含加法自然出现的一个具体应用，是 Lynch 与 Welch 风格的容错硬件时钟同步 [13], ，其以亚稳态包含的方式在 [1]中实现。其中，时间数字转换器（时间数字转换器）以精度-1编码（通常为格雷码或一元码）产生相位测量值，即每次测量与真实整数的偏差至多为一，且至多包含一个亚稳态位。在Lynch-Welch算法的一个基本步骤中，每个节点收集若干此类相位测量值，丢弃离群值，对剩余值求和，然后将和除以2 ，从而得到施加于其本地时钟的微小校正。

因此，计算核心需要将两个编码后的相位测量值相加，每个测量值至多包含一个亚稳态位。加法的总输入不精确度至多为 $k=2$，我们可以使用相应的混合码（在第 6 节中称为$\gamma^{\mathrm{h}}_{n,\ 3}$ ），该编码仅使用2个冗余位；也就是说，与串接一元编码变体的标准 n位格雷码相比，所需的额外开销很小。

Lynch-Welch更新中剩余的运算是除以2。同一模板——设计一个为我们的编码实现该运算的布尔电路、将非码字映射为码字，并推导该电路的mc变体——也可用于此任务。然而，时钟的模拟校正极有可能可以按其构造直接按比例缩小，而不能想当然地认为加法在模拟域中是简单直接的。总之，控制回路中的所有非平凡计算都可以以完全亚稳态包含的数字方式完成，而无需退回到效率极低的一元加法器。

1.4 概述

在第 2 节中，我们定义亚稳态容纳电路的基本概念。关于精度保持码的关键定义在第 3 节给出。我们在第 4 节中形式地证明，不确定性需要冗余。也就是说，给定不确定性的一个（上）界，我们证明冗余位数量的下界。我们给出一种简单而自然的$(n+k)$位编码，它使用 k 个冗余位，称为“混合”编码$\gamma^{\mathrm{h}}_{n,k}$ （上标 h 表示“混合”）。$\gamma^{\mathrm{h}}_{n,k}$ 编码是 k保持的且 $\lceil k/2 \rceil$可恢复的。研究该编码具有广泛意义，因为它在mc 电路 [3]的背景下自然出现。常见编码方案的初步定义在第 5 节给出。$\gamma^{\mathrm{h}}_{n,k}$ 编码及相应证明在第 6 节给出。非正式地说，该编码的第一部分是 n 比特的二进制反射格雷码，第二部分是 k 比特的一元温度计码。所得编码对任意常数 k的码率为$1-\mathcal{O}\left(\frac{1}{n}\right)$。同一编码$\gamma^{\mathrm{h}}_{n,k}$ 可通过使用 $2k-1$ 个冗余位变为 k可恢复的。我们建立 k可恢复加法的概念，并在第 7 节中证明它可以在$\gamma^{\mathrm{h}}_{n,k}$ 编码上执行。接下来，在第 8 节中，我们讨论我们的编码与 [3] 以及纠错码的联系。

2 预备知识

2.1 三值逻辑与含亚稳态的电路

在本研究中，我们采用了 Kleene [14, §64], 提出的布尔逻辑经典扩展，该扩展允许存在未指定信号。下文中，我们将布尔值 $\mathbb{B} := \{0,1\}$ 称为稳定值，而额外的第三个逻辑值$\mathtt{M}$ 则称为亚稳态值。由此得到的三值逻辑值集合表示为 $\mathbb{T} := \{0,1,\mathtt{M}\}$。直观地说，我们将 $\mathtt{M}$ 视为一种

表 3 与、或和非基本逻辑门的行为。

and	0	1	M
0	0	0	0
1	0	1	M
M	0	M	M

or	0	1	M
0	0	1	M
1	1	1	1
M	M	1	M

not	
0	1
1	0
M	M

“叠加”，即0和1的“叠加”。采用克莱尼逻辑是对不稳定事件导致的故障进行建模的一种经典方法。

在克莱尼逻辑中，当且仅当稳定输入已经能够确定输出时，基本逻辑门[1] 才会输出稳定值。表 3 给出了与、或和非基本逻辑门的自然扩展。

电路 C 是一个有向无环图，其中每个节点对应一个基本逻辑门。根据电路结构进行归纳，基本逻辑门的行为为任意电路定义了其实现的函数 $C\colon\ \mathbb{T}^n \to \mathbb{T}^m$。我们的目标是设计这样的电路：在接收不确定输入时，其行为与基本逻辑门相似。也就是说，给定所有输入，如果将一个输入位从0 改为 1 不会影响输出，那么将该输入位设为M 也不应影响输出。满足这一概念的电路称为亚稳态包含（*mc*）电路。

这一基本概念的重要性体现在它出现在数字电路设计 [1], [11],[15], 逻辑 [14], [16], 和网络安全 [17], [18]等迥异领域中。因此，多年来 mc 电路的设计一直受到广泛关注。早期，Huffman 证明所有布尔函数都存在 mc 电路 [11]。然而，近期已有研究证明，对于显式函数，即使允许电路出现亚稳态扰动 [5], ，也存在无条件的指数下界，包括多项式规模的电路。此外，除非（电路变体的）P = NP 成立，否则 NP 完全语言的验证器函数不存在多项式规模的 mc 电路。这些下界与 Ikenmeyer 等人提出的一般构造 [5] 相辅相成；该构造可生成规模为 $n^{\mathcal{O}(k)}|C|$的电路，其中 k 是其输入中亚稳态位数量的上界，而 C 是实现所需函数的任意电路。

2.2 叠加与解析

为了讨论上述精度概念（即一个编码字可能表示的值域；参见第 1.1 节），我们使用 [1] 中所定义的解析 与叠加 运算。扩展码字的不精确度的精确概念在定义 3.2 中予以形式化。

当其输入在稳定输入值上不一致时，叠加会得到不确定值 M。

定义 2.1（叠加）. 用运算符 $*\colon\ \mathbb{T}\times\mathbb{T}\to\mathbb{T}$ 表示两个比特的叠加 。对于 $x, y \in \mathbb{T}$，$x * y = x$ 若 $x = y$，否则为 $x * y = \mathtt{M}$。我们将 $*$运算扩展到 $x, y \in \mathbb{T}^n$，通过对每个下标 $i \in \{1, \ldots, n\}$ 应用该运算，使得

$$(x * y)_i = \begin{cases} x_i & \textit{if } x_i = y_i, \\ \mathtt{M} & \textit{otherwise.} \end{cases}$$

$*$运算满足结合律与交换律，因此对于 $X \subseteq \mathbb{T}^n$，我们定义 $*X := x_1 * \ldots * x_n$，其中 $x_1, \ldots, x_n$ 是 X 中元素的任意枚举。第二种运算，即解析，通过将每个M 替换为两个稳定值，把三元字映射为布尔字的集合。

1. 基本逻辑门的具体选择并不重要，参见 [5]；因此，我们坚持使用 and、or 和 not。

定义 2.2（解析）。将 $\mathcal{P}\,\mathbb{B}$ 解析 记为 $x \in \mathbb{T}$ res: $x = \{0,1\}$ ()。对于 $x = \{x\}$，若 M 则 res()，否则 res()。我们以自然的方式将此推广到长度为 n 的字，通过设定 $\mathbb{T} \to$。

$$\mathrm{res}(x) := \{y \in \mathbb{B}^n \mid \forall i \in \{1, \ldots, n\} \colon x_i \neq \mathtt{M} \Rightarrow y_i = x_i\},$$

对于 $x \in \mathbb{T}^n$。

根据这一直觉，叠加可能会降低精度，因为对叠加进行解析可以增加新的字。例如，字100 与111的叠加为1MM。对该三值字进行解析得到res(1MM) ={100, 101, 110, 111}，因此可能取值的集合已从 {100, 111} 增长到四个元素。

2.3 精确编码

在我们于第 3 节正式定义编码所需性质之前，我们通过描述期望行为来加以说明。假设给定两个精度为 k 的 n位一元编码输入，这意味着输入 $x \in \mathbb{T}^n$ 满足res(x) 至多包含 $k+1$ 个码字，且这些码字全部连续。换言之，由x 表示的可能取值构成一个大小至多为 $k+1$的区间。此类输入具有形式 $1^*\mathtt{M}^*0^*$，其中M的个数至多为 k，且该字的总长度为 n。用 $\gamma^{\mathrm{u}}\colon\ [n+1] \to \mathbb{T}^n$ 表示一元温度计码的编码函数。全文中，对于 $M \in \mathbb{N}$，我们记 $[M]\colon = \{0, \ldots, M-1\}$；特别地，$[n+1] = \{0, \ldots, n\}$。由此可得，输入具有如下形式

$$\overset{j}{\underset{\ell=i}{*}} \gamma^{\mathrm{u}}(\ell) = \overset{j}{\underset{\ell=i}{*}} 1^\ell 0^{n-\ell} = 1^i \mathtt{M}^{j-i} 0^{n-j},$$

其中 i，$j \in [n+1]$ 和 $0 \le j - i \le k$。

两个 n 位一元码的和可以用一个 $(2n)$ 位一元码表示。将输入 $*_{\ell=i}^{j} \gamma^{\mathrm{u}}(\ell) = 1^i \mathtt{M}^{j-i} 0^{n-j}$和 $*_{\ell=i'}^{j'} \gamma^{\mathrm{u}}(\ell) = 1^{i'} \mathtt{M}^{j'-i'} 0^{n-j'}$（其中 i、i'、j、$j' \in [n+1]$）送入一个保持精度的加法器，输出为 $*_{\ell=i+i'}^{j+j'} \gamma^{\mathrm{u}}(\ell)\, 1^{i+i'}\ {}^{j+j'-i-i'} 0^{2n-j-j'}$

M。这种加法以尽可能高的程度保留信息：我们用 $\langle i, j\rangle$ 表示集合 $\{i, \ldots, j\}$。这些词的解析结果所对应的码字分别表示集合 $\langle i, j\rangle$、$\langle i', j'\rangle$ 和 $\langle i+i', j+j'\rangle$。[2] 换言之，输出中关于所表示值的不确定性与输入的不确定性相符。读者可以将其视为对区间加法的忠实表示。

上述一元温度计编码还有另一个关键性质。我们可以始终将输出的任意解析映射回一个码字。我们将解码函数 $(\gamma^{\mathrm{u}})^{-1}\colon\ \gamma^{\mathrm{u}}([n+1]) \to [n+1]$ 扩展定义为 $(\tilde{\gamma}^{\mathrm{u}})^{-1}\colon \mathbb{B}^n \to [n+1]$，如下所示：[3] 对于 $x \in \mathbb{B}^n$，令 $i \in [n+1]$ 为满足对所有 $j \in [i+1]$均有 $x_j = 1$ 的最大值。然后 $(\tilde{\gamma}^{\mathrm{u}})^{-1}(x) := i$，接着 $\gamma^{\mathrm{u}}((\tilde{\gamma}^{\mathrm{u}})^{-1}(x)) = 1^i 0^{n-i}$。请注意，这保证了 $(\tilde{\gamma}^{\mathrm{u}})^{-1}$(res($*_{\ell=i}^{j} \gamma(\ell)$)) $= \{i, \ldots, j\}$，也就是说，稳定化后（当每个亚稳态位都已解析为 0 或 1 时），我们可以使用 $(\tilde{\gamma}^{\mathrm{u}})^{-1}$，从字 $1^i\mathtt{M}^{j-i}0^{n-j}$ 中恢复其所表示值域中的一个值。这一性质与纠错码在发生位翻转时恢复正确码字的能力非常相似；在这里，我们恢复一个“正确”的值，其含义是该值与现有信息一致。我们称这种编码具有可恢复性。我们将看到，这一性质非常有用，而且是我们所能期待的最佳结果。

2. 并非所有 $1^*\mathtt{M}^k0^*$ 解析都是一元码字。2^k 个解析中只有 $k+1$ 个是码字。

3. 函数 $f\colon\ X \to Y$通过将 f 应用于 X'中的每个元素，作用于集合 $X' \subseteq X$；$f(X')\colon = \{f(x) | x \in X'\}$。

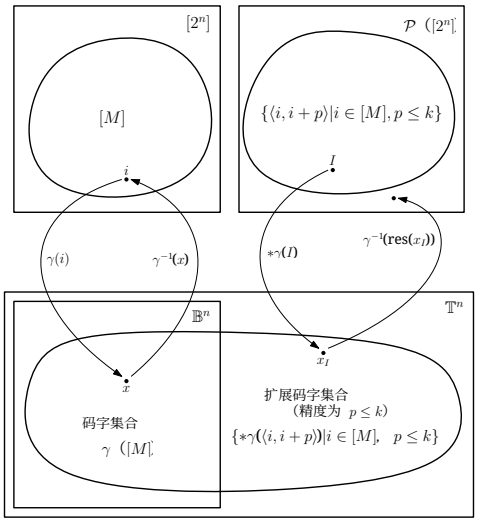

图1. 编码与解码函数及其向三元世界扩展的概述。

自然地，上述正面示例的意义有限，因为一元编码带来的指数级开销使其效率极低。那么，如果我们使用非冗余编码会怎样？不难看出，标准二进制编码会导致灾难。例如，设 $\gamma_n^{\mathrm{b}}\colon\ [2^n] \to \mathbb{B}^n$ 为一个[4] $\gamma_4^{\mathrm{b}}(1) = 000$ 位二进制码，[4]则有 $\gamma_4^{\mathrm{b}}(7) = 0111$，而 $\gamma_4^{\mathrm{b}}(0)\,()\,\mathtt{M}$；对于这对词，其和究竟应为 7 还是 8，并不明确。相比之下，$\gamma_4^{\mathrm{b}}(7+0) * \gamma_4^{\mathrm{b}}(7+1) = 0111{*}1000 = \mathtt{MMMM}$，也就是说，所得词完全不含任何信息。这是一个极端例子：即使已知所有其他位，单个输入位也会影响所有输出位。在克莱尼逻辑中，当输入存在不确定性时，该编码无法提供任何非平凡的保证。

上述示例表明，我们应致力于设计这样的编码：面对不稳定输入时，其表现类似于一元温度计编码，同时又具有高码率。然而，与这一目标相矛盾的是，输入中的不确定性要求编码具有冗余。例如，不含冗余位的编码无法从超过一个 M 中恢复（见定理 3.7）。更一般地说，输入中的每个不确定位都需要增加一个冗余位，也就是说，对于 k-可恢复性，我们需要 k 个冗余位。我们在定理4.2中证明这一结论。

3 三元逻辑中的编码

本节将正式定义三值逻辑中的编码及相关术语。本节的第二部分将正式阐述 k-保持和 k-可恢复编码的概念。所有定义均配有示意图。

3.1 编码的定义

编码是有限数集与定长二进制字之间的映射。任何编码都由其编码函数定义。编码函数定义码字集合和解码函数。

定义 3.1（编码函数）。一个编码由一个单射函数 $\gamma\colon\ [M] \to \mathbb{B}^n$ 给出，该函数将集合 $[M]$ 中的整数映射为 n位字。我们称 γ 为编码函数，其像 $\gamma([M])$ 为码字集合。γ 在码字集合上的逆函数，记为 $\gamma^{-1}\colon\ \mathbb{B}^n \to [M]$，称为解码函数。每个编码函数都定义了一个解码函数。解码函数仅在定义域 $\gamma([M])$上有定义。

通过对所有码字进行逐位叠加，可以将定义域的一个子集映射为三值字。我们将其称为该子集的扩展码字。在本文中，我们始终考虑由连续整数构成的子集，并将其称为区间。对于整数 $i \le j$，我们用 $\langle i,\ j\rangle$ 表示区间 $\{i, \ldots, j\}$。其大小为通常意义上的基数 $|\langle i,\ j\rangle| = j - i + 1$。例如，对于 $I = \langle i,\ i+p\rangle$，有 $|I| = p + 1$。图 1 展示了由编码定义的集合和映射概览。

定义 3.2. 对于编码 $\gamma\colon\ [M] \to \mathbb{B}^n$ 以及区间 $I = \langle i, i+p\rangle \subseteq [M]$ 我们定义

the extended codeword x_I *by* $x_I \coloneqq *\gamma(I)$,
the range r_{x_I} *of* x_I *by* $r_{x_I} \coloneqq I$, *and*
the imprecision *of* x_I *by* $p = |I| - 1$.

取决于编码函数，这样的扩展码字可能无法保留关于原始集合的信息。它并非在所有情况下都能被映射回去。例如，标准二进制编码由 $\gamma_n^{\mathrm{b}}\colon\ [2^n] \to \mathbb{B}^n$ 给出。考虑区间 $\langle 3, 4\rangle = \{3, 4\}$，其中码字由 $\gamma_n^{\mathrm{b}}(3) = 0011$ 和 $\gamma_n^{\mathrm{b}}(4) = 0100$ 给出。扩展码字是所有码字的叠加，即为 0MMM。对扩展码字的解析会产生更多码字，使得 $\mathrm{res}(\mathtt{0MMM}) \neq \{\gamma_n^{\mathrm{b}}(3)\ \gamma_n^{\mathrm{b}}(4)\}$。

叠加一般不可逆，因为它不是单射函数。对于编码而言，这意味着可能存在多个区间映射到同一个扩展码字。因此，将区间映射到三元字会丢失关于原始区间的信息。

二进制编码 γ_n^{b} 将区间 $\langle 5,\ 6\rangle$ 和 $\langle 4,\ 7\rangle$ 映射为同一个三值字。由二进制编码，我们得到以下码字：$\gamma_n^{\mathrm{b}}(4) = 0100$,$\gamma_n^{\mathrm{b}}(5) = 0101$,$\gamma_n^{\mathrm{b}}(6) = 0110$ $\gamma_n^{\mathrm{b}}\ 7 = 0111$，以及() 。因此，区间 $\langle 5,\ 6\rangle$ 和 $\langle 4,\ 7\rangle$ 的扩展码字分别为 $\langle 5, 6\rangle = *\{0101, 0110\} = \mathtt{01MM}$和 $\langle 4,\ 7\rangle = *\{0100, 0101, 0110, 0111\} = \mathtt{01MM}$。

我们注意到，值域（如定理 3.2 中所述）定义得并不明确。对于一个扩展码字，可能有多个值域映射到该码字，如上例所示。只有当区间可由上下文明确判断时，我们才使用范围记法来表示该区间。

3.2 保持性与可恢复编码

现在我们定义编码的两个重要性质，它们可防止将区间映射为扩展码字时发生信息丢失。参数 k 描述了在不丢失信息的情况下可映射为扩展码字的区间最大长度。

首先，保持性确保原始区间得到保留，即对扩展码字进行解析时，不会在原始区间之外添加新的码字。不过，解析可能会添加非码字。图2给出了正面和负面的示例。

定义 3.3 (k-保持编码)。定义域为 $[M]$ 的编码 γ 是k-保持的，当且仅当对于每个区间 $I = \langle i, i+p\rangle$，其中 $p \le k$，解析$\mathrm{res}(x_I)$不包含不属于 $\gamma(I)$的码字。形式化地，

$$\gamma([M]) \cap \mathrm{res}(x_I) = \gamma(I)\,.$$

其次，可恢复性确保扩展码字解析中的每个非码字都能映射回原始区间。图 3 展示了这一性质，其定义如下。

4. 二进制码在第 5 节中定义。

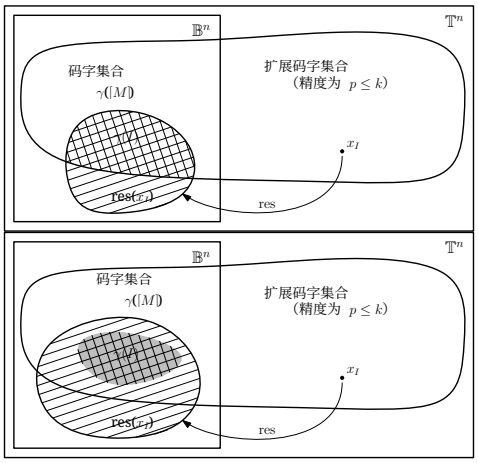

图 2. 保持性编码（上）与非保持性编码（下）的示意图。阴影区域表示 $\mathrm{res}(x_I)$，棋盘格区域表示 $\gamma(I)$。对于保持性编码，扩展码字 x_I 的解析不得产生新的码字，但可以产生非码字的二进制字。

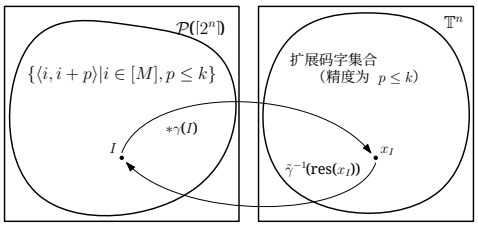

图 3. 可恢复性性质的示意图。

定义 3.4 (k-可恢复编码)。 设 x 是一个不精确度为 $p_x \le k$的扩展码字。如果存在一个将 γ^{-1} 从码字扩展到所有二进制字的扩展，使得 x 的每个解析都映射到 x的值域，则编码 γ 是 k-可恢复的。形式化地，将γ^{-1}: $\gamma([M]) \to [M]$ 的扩展记为 $\widetilde{\gamma}^{-1}$: $\mathbb{B}^n \to [M]$。那么，如果对于 $I = \langle i,\ i+p\rangle$ 和 $p \le k$，γ是k-可恢复的，

$$\widetilde{\gamma}^{-1}(\mathrm{res}(x_I)) \subseteq I\,.$$

保持性是可恢复性的必要条件，也就是说，每个可恢复码也是保持性的。以下证明了这一点。

引理 3.5. 每个可 k恢复的编码也具有 k保持性。

证明： 我们证明其逆否命题，即如果一个编码不具有 k保持性，那么它就不可 k恢复。假设编码 γ 不具有 k保持性。根据定理 3.3，存在一个区间 I，使得有某个 $x \in \gamma([M]) \cap \mathrm{res}(x_I)$ 不是 $\gamma(I)$ 中的元素。根据定义 3.1，解码函数满足 $\gamma^{-1}(\gamma(I)) = I$。因此，由于 $x \notin \gamma(I)$，可得 $\gamma^{-1}(x) \notin I$。所以，对 γ^{-1} 的任何扩展都无法满足要求 $\widetilde{\gamma}^{-1}(\mathrm{res}(x_I)) \subseteq I$。

□

3.3 编码效率

编码的效率通常用其码率来衡量。码率的倒数可以衡量编码中添加了多少冗余位。在本研究中，我们使用冗余 来衡量编码的效率。编码的冗余度是所用位数与编码相同信息所需最少位数之比 [19]。形式上，冗余度和码率定义如下。

定义 3.6 （冗余度和码率）。 设 γ: $[2^m] \to \mathbb{B}^\ell$ 是一个编码。γ 的冗余度 为 $\ell/m \ge 1$，其码率 定义为 $m/\ell \le 1$。

冗余为1 的编码是一个双射。每个二进制字都是一个码字，可以映射到一个自然数。由定理 3.3 可推知，冗余为1 的编码，其保持性不可能超过1。考虑一个包含三个元素的任意区间，相应的扩展码字至少有2个不稳定比特。因此，该扩展码字的解析包含四个元素，它们都是码字，因为不存在冗余的非码字。由于保持性是可恢复的必要条件，这类编码的可恢复性不可能超过1。

推论 3.7. 任何冗余度为1 （最优）的编码，至多是1-保持且1-可恢复的。

一类特别值得我们关注的编码是格雷码。如果两个连续码字的汉明距离1，则该编码就是格雷码。两个字的汉明距离是指两者比特不同的位置数。

定义 3.8 （格雷码）。 若编码 γ: $[M] \to \mathbb{B}^n$ 中的码字 γ（i） 和 γ（$i+1$ 模 M） 的汉明距离为1，则该编码为格雷码。

给定一个格雷码，对于大小为二的区间，其每个扩展码字都恰好包含一个M 位。扩展码字的解析结果仅包含原区间中的两个数。由此可得，每个格雷码至少是1-保持且1-可恢复的。

4 可恢复性需要冗余

我们的主要贡献之一是证明，更高的可恢复性需要更高的冗余。本节给出 k-可恢复性所需冗余的下界[5]。

我们证明，一个 n 位编码若是 k-可恢复的，则最多可以编码 $2^{n-k}(k+1)$ 个值。 k-可恢复码的冗余下界为 $n/(n-k+\log(k+1))$。我们首先证明，不精确度为 $p \le k$ 的扩展码字至少包含 p 个 M。回顾一下， M 表示值域的大小，也就是说，我们对 $[M] = \{0, \ldots, M-1\}$ 中的值进行编码。

引理 4.1. 对于 $k, n \in \mathbb{N}$，设 γ 是一个 n 位、保持 k 的编码。设 $p \le k$ 和 $i \in [M-p]$，则 $*\{\gamma(i), \ldots, \gamma(i+p)\}$ 至少包含 p 个 M。[6]

证明： 我们对 p 使用归纳法来证明该命题。基例 $p = 0$ 显然成立，因为码字 $\gamma(i)$和 $\gamma(i+p)$ 完全相同。它们的汉明距离为0，且没有M。

接下来，我们证明从 $p-1$ 到 p的归纳步骤，其中$0 < p \le k$。直观地说，从 $\gamma(i+p-1)$ 变为 $\gamma(i+p)$时，一个保持 k的编码必须翻转一个位，而这个位在从 $\gamma(i)$ 变为 $\gamma(i+p-1)$时保持不变。

设 x_I 为一个扩展码字，且 $I = \langle i, i+p-1\rangle$。编码 γ 是保持 k的，因此码字 $\gamma(i+p)$ 不能属于$\mathrm{res}(x_I)$。对于 x_I 和 $\gamma(i+p)$，至少有一个稳定比特不同。因此，叠加 $*\{x_I, \gamma(i+p)\}$ 比 x_I至少多一个M 。归纳步骤至此完成。

□

5. 冗余的下界就是编码码率的上界。

6. 人们可能会倾向于将其理解为汉明距离 p。考虑编码 100、010、001。它是3-保持的，所有码字彼此之间的汉明距离均为 2，但 $*\{100, 010, 001\} = \mathtt{MMM}$。

有了定理 4.1，我们可以证明，任何 k- 可恢复码最多只能有 $2^{n-k}(k+1)$ 个码字。

定理4.2. 对于 k，$n \in \mathbb{N}$，如果 γ 是 k-可恢复的，那么定义域为 $\gamma\colon [M] \to \mathbb{B}^n$ 的编码的大小至多为 $M \le 2^{n-k}(k+1)$。

证明：考虑定义域为 $[M]$、满足某个 $M \in \mathbb{N}$ 的 k-可恢复编码 γ。我们证明，对于固定的 $M = 2^{n-k}(k+1)$，陪域中至少必须包含 2^n 个不同的码字。考虑一个大小为 $k+1$ 的区间族 I_ℓ。对于 $\ell \in [2^{n-k}]$，定义 $I_\ell = \langle \ell(k+1)\ \ell(k+1)+k \rangle$。区间族 I_ℓ 是 $[M]$ 的一个划分。由于 γ 是 k-可恢复的，存在一个扩展 $\tilde{\gamma}^{-1}$，使得对所有 ℓ，都有 $\tilde{\gamma}^{-1}(\mathsf{res}(x_{I_\ell})) \subseteq I_\ell$。由于 $I_\ell \cap I_{\ell'} = \emptyset$，对于 ℓ、$\ell' \in [2^{n-k}]$ 和 $\ell \neq \ell'$，我们得到 $\mathsf{res}(x_{I_\ell}) \cap \mathsf{res}(x_{I_{\ell'}}) = \emptyset$。因此，存在 2^{n-k} 个不同的区间，它们对应的扩展码字具有互不重叠的解析集。

我们证明，对于每个 I_ℓ，解析 $\mathrm{res}(x_{I_\ell})$的大小至少为 2^k。解析 $\mathrm{res}(x_{I_\ell})$包含码字 $\gamma(i)$和 $\gamma(i+k)$。根据定理 4.1，x_I至少有 k 个 $\mathtt{M}$。由此可得 $|\mathrm{res}(x_I)| \ge 2^k$。因此，存在 $[2^{n-k}]$ 个不同区间 I_ℓ，其解析 $\mathrm{res}(x_{I_\ell})$ 的大小至少为 2^k。相应地，陪域中至少有 $2^{n-k} \cdot 2^k = 2^n$ 个可能的词。因此，若 $M = 2^{n-k}(k+1)$，陪域的大小至少为 2^n。

最后，固定陪域 $\mathbb{B}^n$，即大小为 2^n 的陪域。反设定义域为 $M > 2^{n-k}(k+1)$。我们将 M 划分为大小为 $k+1$ 的区间。区间数超过 2^{n-k}，其中一个区间可能小于 $k+1$。根据上述结论，我们为前 2^{n-k} 个区间使用陪域中的 2^n 个词。剩余区间还需要更多词，但陪域中的所有词都已分配完毕。

□

定理4.2表明，对于 k可恢复性，n位编码γ最多可编码 $2^{n-k}(k+1)$ 个码字。因此我们得出，定义域为$[2^{n-k}(k+1)]$ 且陪域为 $\mathbb{B}^n$ 的任何编码，其可恢复性至多为k。

推论 4 3 k $n \in \mathbb{N}$ γ $[M] \to \mathbb{B}^n$ $|M| >$ .. 对于，，2^{n-k} $k+1$ k 不存在编码：满足() 且具有-可恢复性。

定理 4.3的直接结果是，任何具有 $M = 2^n$的编码，其可恢复性都不可能超过1。在下一节中，我们将介绍两种最优码率编码：一种是不具备1可恢复性的编码（标准二进制码），另一种是具备1可恢复性的编码（二进制反射格雷码）。

5种常见编码方案

本节将介绍以下几种（众所周知的）编码：标准二进制码、一元温度计码和二进制反射格雷码（BRGC）。它们在冗余、保持性和可恢复性方面各不相同。每种

将讨论相应编码的这一性质。标准二进制码是一个众所周知但不适用于 k可恢复码的编码示例。它将用于加法；参见第 7 节。一元温度计码和二进制反射格雷码将在第 6 节中用于构造混合码。

在符号表示方面，我们使用上标来区分特定的编码族，同时将 γ 保持为通用编码符号：γ_n^{b}、γ_n^{g} 和 γ_n^{u} 分别表示二进制、二进制反射格雷码和一元编码族，而 $\gamma_{n,\ k}^{\mathrm{h}}$ 表示我们的混合码。

5.1 标准二进制码

我们用 γ_n^{b} 表示标准二进制码的编码函数。标准二进制码也称为以 2 为底的数制系统。

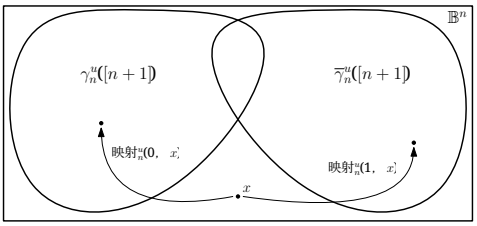

图 4. 从字到一元编码的映射 $\tilde{u}$示意图。

定义 5.1。对于 $n \in \mathbb{N}_{>0}$ 和 $i \in [2^n]$，我们定义二进制编码函数 γ_n^{b}：$[2^n] \to \mathbb{B}^n$，如下所示：

$$\gamma_n^{\mathrm{b}}(i) \coloneqq \gamma_{n-1}^{\mathrm{b}}(\lfloor i/2 \rfloor) \circ x_n\,,$$

其中 $x_n :=$（i 模 2），且“∘”表示比特串的串接。

标准二进制码具有（最优）冗余1，$[2^n]$ 中的每个数都可映射为 $\mathbb{B}^n$ 中的一个二进制字，反之亦然。从第 1 节和第 3 节的示例中可以看出，该编码既不是1保持的，也不是1可恢复的。

5.2 一元温度计码

接下来，我们定义一元温度计码。该编码有两种不同的类型，我们将其命名为 γ_n^{u} 和γ_n^{u}。编码 γ_n^{u}通过连续1的个数来表示所编码的值。反之，γ_n^{u} 通过连续0的个数来表示该值。编码 γ_n^{u} 由 γ_n^{u} 翻转所有比特得到。编码 γ_n^{u} 和 γ_n^{u} 的冗余均为 $n/\log(n+1)$。

定义 5.2（一元温度计码）。对于 $n \in \mathbb{N}$和$i \in [n+1]$，我们定义编码函数

$$\gamma_n^{\mathrm{u}}(i) \coloneqq 1^i 0^{n-i}\,, \textit{and}$$
$$\overline{\gamma}_n^{\mathrm{u}}(i) \coloneqq \overline{\gamma_n^{\mathrm{u}}(i)} = 0^i 1^{n-i}\,.$$

这些编码的两个示例字为 γ_4^{u}（3）$=1110$、γ_4^{u}（3）$=0001$。

我们观察到，不精确度为 p的每个扩展码字恰好有 p 个$\mathtt{M}$ 位。这一观察表明，在这种情况下，每个扩展码字的前部有 i个稳定比特，后部有（$n-i-p$）个稳定比特。

观察 5.3. 设 n，$p \in \mathbb{N}$ 和 $i \in [n+1]$。对于任意 $I = \langle i,\ i+p \rangle$，其中 $i+p \le n$，有

$$*\gamma_n^{\mathrm{u}}(I) = \gamma_n^{\mathrm{u}}(i) * \gamma_n^{\mathrm{u}}(i+p) = 1^i \mathtt{M}^p 0^{n-i-p}\,, \textit{and}$$
$$*\overline{\gamma}_n^{\mathrm{u}}(I) = \overline{\gamma}_n^{\mathrm{u}}(i) * \overline{\gamma}_n^{\mathrm{u}}(i+p) = 0^i \mathtt{M}^p 1^{n-i-p}\,.$$

特别地，每个解析的前部和后部都与扩展码字的相同。因此，区间的解析不会增加新的码字。因此，一元温度计码的两种类型都是 n保持的。

推论 5.4. 编码 γ_n^{u} 和 γ_n^{u} 是 n-保持的。

接下来，我们证明一元温度计码的两种类型都是 n-可恢复的。为此，我们首先定义映射函数$\mathrm{map}_n^{\mathrm{u}}$：$\mathbb{B} \times \mathbb{B}^n \to \mathbb{B}^n$。给定指示比特 π，该映射将二进制字 x 映射为一个一元温度计码字。若 $\pi = 0$，则映射到γ_k^{u}，即形如$1^\ell 0^{k-\ell}$的码字。尚需决定是根据 x 中的第一个0 还是最后一个1来选择 ℓ。类似地，若 $\pi = 1$，则映射到 γ_k^{u}，并需要决定是根据 x中的第一个1还是最后一个0 来选择 ℓ。

对于二进制字 $x \in \mathbb{B}^n$ $\ell_b^{\max}$ $\ell_b^{\min} \in$，我们 $[n+2] = \{0, \ldots, n+1\}$ $\ell_b^{\max}$ 定义索引,如下。索引是最大的

索引，使得 $x_{\ell_b^{\max}} = b$ 且 $\ell_b^{\max} = 0$；如果不存在这样的索引，则为。索引 $\ell_b^{\min}$是最小的索引，使得 $x_{\ell_b^{\min}} = b$ 且 $\ell_b^{\min} = n+1$；如果不存在这样的索引，则为。举例来说，给定二进制字00110011，我们得到 $\ell_0^{min} = 1$、$\ell_1^{min} = 3$、$\ell_0^{max} = 6$ 和$\ell_1^{max} = 8$。

定义 5.5（一元温度计映射）。设 $\pi \in \mathbb{B}$ 和 $x \in \mathbb{B}^n$、索引 $\ell_b^{\max}$、$\ell_b^{\min}$是 x的索引。映射 $\mathrm{map}_n^{\mathrm{u}}\colon \mathbb{B}\times\mathbb{B}^n \to \mathbb{B}^n$ 定义如下：

$$\mathrm{map}_n^{\mathrm{u}}(\pi, x) \coloneqq \begin{cases} 1^{\ell_0^{\min}-1} \circ 0^{k-\ell_0^{\min}+1} & \text{, if } \pi = 0 \text{ and } x_{\lceil n/2\rceil} = 0 \\ 1^{\ell_1^{\max}} \circ 0^{k-\ell_1^{\max}} & \text{, if } \pi = 0 \text{ and } x_{\lceil n/2\rceil} = 1 \\ 0^{\ell_0^{\max}} \circ 1^{k-\ell_0^{\max}} & \text{, if } \pi = 1 \text{ and } x_{\lceil n/2\rceil} = 0 \\ 0^{\ell_1^{\min}-1} \circ 1^{k-\ell_1^{\min}+1} & \text{, if } \pi = 1 \text{ and } x_{\lceil n/2\rceil} = 1\,. \end{cases}$$

备注。这里无需根据 $x_{\lceil k/2\rceil}$ 选择索引 ℓ。当在 $\langle \ell_0^{\min}-1, \ell_1^{\max}\rangle$ 中任意选择 ℓ 时，对于 $\mathrm{map}_n^{\mathrm{u}}(0,x)$，以及对于 $\mathrm{map}_n^{\mathrm{u}}(1,x)$，分别可以获得一元温度计码的可恢复性。然而，映射 $\mathrm{map}_n^{\mathrm{u}}$ 将在下一章中用于将二进制字映射为混合编码中的字。

在证明一元温度计码可恢复之前，我们先证明对扩展码字的解析进行映射是正确的。具体而言，对于区间 I和 $x \in \mathrm{res}(x_I)$，我们证明 $\mathrm{map}_n^{\mathrm{u}}(0,x)$（相应地，$\mathrm{map}_n^{\mathrm{u}}(1,x)$）将 x 映射为 $\gamma_n^{\mathrm{u}}(I)$（相应地，$\overline{\gamma}_n^{\mathrm{u}}(I)$）中的一个码字。

引理 5.6. 设 n，$p \in \mathbb{N}$、$i \in [n+1]$ 以及 $I = \langle i,\ i+p\rangle$，且 $i+p \le n$。对于 $x_0 \in \mathrm{res}(*\gamma_n^{\mathrm{u}}(I))$ 与 $x_1 \in \mathrm{res}(*\overline{\gamma}_n^{\mathrm{u}}(I))$，成立

$$\mathrm{map}_n^{\mathrm{u}}(0,x_0) \in \gamma_n^{\mathrm{u}}(I)\text{, and}$$
$$\mathrm{map}_n^{\mathrm{u}}(1,x_1) \in \overline{\gamma}_n^{\mathrm{u}}(I)$$

在证明该命题之前，我们首先观察到，该映射的按位取反等于对取反后的自变量进行映射。

观察 5.7. 设 $\pi \in \mathbb{B}$ 与 $x_u \in \mathbb{B}^k$，则

$$\overline{\tilde{u}(\pi, x_u)} = \tilde{u}(\overline{\pi}, \overline{x_u})\,.$$

证明：该观察可通过定义中四种情形的分情况讨论加以验证。每种情形均源于如下关键洞见：x_u 的 $\ell_b^{\max}$（相应地，$\ell_b^{\min}$）等于 $\overline{x_u}$ 的 $\ell_b^{\max}$（相应地，$\ell_b^{\min}$）。 □

现在我们可以给出定理 5.6 的证明。

证明：我们仅证明情形 $\mathrm{map}_n^{\mathrm{u}}(0,\ x_0) \in \gamma_n^{\mathrm{u}}(I)$，因为另一情形可由定理 5.7 得出。假设 $x_0 \in \mathrm{res}(*\gamma_n^{\mathrm{u}}(I))$，由定理 5.3 可知 $\ell_0^{\min}-1,\ \ell_1^{\max} \in \langle i,\ i+p\rangle$。因此，$\mathrm{map}_n^{\mathrm{u}}(0,\ x_0) = 1^{\ell}0^{n-\ell}$，对某个 $\ell \in \langle i,\ i+p\rangle = I$。该结论由一元温度计码的定义得出。 □

如果我们通过映射$\mathrm{map}_n^{\mathrm{u}}$ 来定义解码函数的扩展，就能立即得到一元温度计编码的可恢复性，这由定理 5.6 得出。

引理 5.8. 编码 γ_n^{u} 和 $\overline{\gamma}_n^{\mathrm{u}}$ 是n-可恢复的。

证明：我们证明 n的可恢复性。对于 γ_n^{u}，证明过程相同，故略去。

将解码函数 $(\widetilde{\gamma}_n^{u})^{-1}\colon \mathbb{B}^n \to [n+1]$ 对 $x \in \mathbb{B}^n$ 的扩展定义为

$$(\widetilde{\gamma}_n^{u})^{-1}(x) \coloneqq (\gamma_n^{u})^{-1}(\mathrm{map}_n^{\mathrm{u}}(0,x))\,,$$

该结论由定理 5.6 的第一种情形得出。 □

备注。从$\gamma_n^{\mathrm{u}}(n)$ 切换到$\gamma_n^{\mathrm{u}}(0)$时，以及从 $\gamma_n^{\mathrm{u}}(n)$ 切换到 $\gamma_n^{\mathrm{u}}(0)$时，每一位都会翻转。这在使用一元温度计码构造我们的混合编码时会造成问题。将γ_n^{u} 的码字附加到 γ_n^{u}的码字后，我们就能解决这个问题。这种组合编码也是一种蛇盒编码。

表4 二进制反射格雷码 γ_4^{g}。

i	$\gamma_4^{\mathrm{g}}(i)$	i	$\gamma_4^{\mathrm{g}}(i)$	i	$\gamma_4^{\mathrm{g}}(i)$	i	$\gamma_4^{\mathrm{g}}(i)$
0	0000	4	0110	8	1100	12	1010
1	0001	5	0111	9	1101	13	1011
2	0011	6	0101	10	1111	14	1001
3	0010	7	0100	11	1110	15	1000

5.3 二进制反射格雷码（二进制反射格雷码）

顾名思义，这种编码是通过反射过程从标准二进制码构造而成的。直观地说，对于一个 n 位编码，我们依次遍历首位固定为 0 的所有长度为 $n-1$ 的码字。下一次向上计数时，将首位翻转为 1，同时固定其余 $n-1$ 位。我们将计数序列中的单次递增步骤称为 向上计数，也就是从 i 对应的码字移至 $i+1$ 对应的码字。随后，我们反转计数顺序。首位固定为 1 时，我们按逆序遍历所有长度为 $n-1$ 的码字。例如，表4的前两列清楚地展示了这一点。该编码先遍历第三位和第四位，然后翻转第二位并反转顺序。形式上，我们通过递归定义长度为 n 的二进制反射格雷码 γ_n^{g}。

定义5.9（二进制反射格雷码）。对于 $n \in \mathbb{N}$，我们递归地定义编码函数 $\gamma_n^{\mathrm{g}}\colon [2^n] \to \mathbb{B}^n$，如下所示。1 位二进制反射格雷码定义为 $\gamma_1^{\mathrm{g}}(0) \coloneqq 0$和 $\gamma_1^{\mathrm{g}}(1) \coloneqq 1$。对于 $n>1$ 和 $i \in [2^n]$，编码函数 γ_n^{g} 定义为

$$\gamma_n^{\mathrm{g}}(i) \coloneqq \begin{cases} 0 \circ \gamma_{n-1}^{\mathrm{g}}(i) & \text{, if } i \in [2^{n-1}] \\ 1 \circ \gamma_{n-1}^{\mathrm{g}}(2^n - i - 1) & \text{, if } i \in [2^n] \setminus [2^{n-1}]\,. \end{cases}$$

例如，编码γ_4^{g} 列于表4中。二进制反射格雷码的（最优）冗余为1；$[2^n]$ 中的每个数都可以映射为 $\mathbb{B}^n$ 中的一个二进制字，反之亦然。我们注意到，根据定理 3.8，编码γ_n^{g} 确实是格雷码。因此，γ_n^{g}是1保持的。

观察5.10。对于 $n \in \mathbb{N}$，给定两个数 $i \in [2^n]$ 和$j = i+1$ 模 2^n，码字 $\gamma_n^{\mathrm{g}}(i)$ 和 $\gamma_n^{\mathrm{g}}(j)$ 的汉明距离为1。

证明：我们通过对编码长度n进行归纳论证来证明该命题。基例 $n=1$ 很简单，所有可能的码字（即0 和1）的汉明距离均为1。对于归纳步骤（$n>1$），我们对 i进行情形区分，其中在情形 $i \in [2^n] \setminus \{2^{n-1}-1, 2^n-1\}$ 中，该命题由 γ_n^{g}的递归定义得出。在情形 $i = 2^{n-1}-1$中，我们得到

$$\gamma_n^{\mathrm{g}}(i) = 0\gamma_{n-1}^{\mathrm{g}}(2^{n-1}-1) \text{ and}$$
$$\gamma_n^{\mathrm{g}}(j) = 1\gamma_{n-1}^{\mathrm{g}}(2^{n-1}-1)\,.$$

因此，这些码字在第一位上不同，该命题成立。在情形 $i = 2^n-1$中，我们得到

$$\gamma_n^{\mathrm{g}}(i) = 1\gamma_{n-1}^{\mathrm{g}}(0) \text{ and}$$
$$\gamma_n^{\mathrm{g}}(j) = 0\gamma_{n-1}^{\mathrm{g}}(0)\,.$$

因此，这些码字在第一位上不同，该命题成立。 □

由于二进制反射格雷码是格雷码，因此二进制反射格雷码具有1保持性。

观察 5.11. 二进制反射格雷码 γ_n^{g} 恰好具有1保持性。

证明：考虑任意区间 $I = \langle i, i+1\rangle$，满足 $i \in [M]$。根据定理5.10，我们知道 $\gamma_n^{\mathrm{g}}(i)$ 和 $\gamma_n^{\mathrm{g}}(i+1)$ 恰好相差一位。因此，x_I 恰好有一个 M 位，从而$\mathrm{res}(x_I) = \{\gamma_n^{\mathrm{g}}(i)\ \gamma_n^{\mathrm{g}}(i+1)\}$。

对于 γ_n^{g}，我们有$|M| = 2^n > 2^{n-2}(2+1)$。因此，根据定理 4.3，我们可以推出 γ_n^{g} 不是2-保持的。 □

6 个 A k-保持的 $\lceil k/2\rceil$-可恢复码

本节中，我们定义一种具有 k-保持性且可 $\lceil k/2\rceil$-恢复的编码。我们使用一种混合编码方案，其中采用 BRGC码以及两种类型的一元温度计码。

在定义混合编码之前，我们需要说明奇偶性的概念。在混合码的定义中，我们利用奇偶性在两种一元编码之间进行选择。一个词的奇偶性表示该词中1的个数是偶数还是奇数。

定义 6.1（奇偶性）。 n 位字的奇偶性记为 par：$\mathbb{B}^n \to \mathbb{B}$。其定义为所有位之和对 2 取模，

$$\mathrm{par}(x) \coloneqq \sum_{i=1}^{n} x_i \bmod 2\,.$$

6.1 混合码

（ $n+k$ $\gamma_{n\,k}^{\mathrm{h}}\ [2^n\ k+1\ ]\to$ ）
$\mathbb{B}^{n+k}$
位混合码记为 . : ()。它由一个 位的二进制反射格雷码部分和一个 位的一元温度计部分组成。如果 BRGC部分具有偶校验，我们将使用 γ_k^{u} 编码进行 $k+1$ 次向上计数。然后将 BRGC部分增加一次向上计数。因此，BRGC部分的奇偶性发生翻转。递增后，我们对 γ_k^{u} 部分进行 $k+1$次向上计数。随后再对 BRGC部分进行一次向上计数，并重复该过程。每次增加 BRGC部分时，我们都会在两种一元编码类型之间切换。因此，使用哪种一元码由 BRGC部分的奇偶性决定。例如，混合码的一部分 $\gamma_{4,\,4}^{\mathrm{h}}$ 如表 5 所示。形式化地，该编码定义如下。

定义 6 2 $\gamma_{n\,k}^{\mathrm{h}}$ $n\ k \in \mathbb{N}$ $n \geq k$ $i \in$ 。(编码 .)。令 ，且，$[2^n\ k+1\ ]$ $x_g \in \mathbb{B}^n$ $x_u \in \mathbb{B}^k$
对于()，定义 和 为

$$x_g \coloneqq \gamma_n^{\mathrm{g}}(\lfloor i/(k+1)\rfloor)\,,$$
$$x_u \coloneqq \begin{cases} \gamma_k^{\mathrm{u}}(i \bmod (k+1)) & \text{, if } \mathrm{par}(x_g) = 0 \\ \overline{\gamma}_k^{\mathrm{u}}(i \bmod (k+1)) & \text{, if } \mathrm{par}(x_g) = 1\,. \end{cases}$$

混合码由 x_g 与 x_u串接而成

$$\gamma_{n,k}^{\mathrm{h}}(i) \coloneqq x_g \circ x_u\,,$$

我们用下标 g 和 u分别表示二进制反射格雷码部分和一元部分，

$$\gamma_{n,k}^{\mathrm{h}}(i)_g \coloneqq \gamma_{n,k}^{\mathrm{h}}(i)_{1,n} = x_g\,,$$
$$\gamma_{n,k}^{\mathrm{h}}(i)_u \coloneqq \gamma_{n,k}^{\mathrm{h}}(i)_{n+1,n+k} = x_u\,.$$

每个编码函数也定义一个解码函数，因为编码是单射的。解码函数$(\gamma_{n,k}^{\mathrm{h}})^{-1}$: $\mathbb{B}^{n+k} \to [2^n(k+1)]$ 可通过以下方式计算

$$(\gamma_{n,k}^{\mathrm{h}})^{-1}(x) = \tag{1}$$
$$(\gamma_n^{\mathrm{g}})^{-1}(x_g)\cdot(k+1) + \begin{cases} (\gamma_k^{\mathrm{u}})^{-1}(x_u) & \text{, if } \mathrm{par}(x_g) = 0 \\ (\overline{\gamma}_k^{\mathrm{u}})^{-1}(x_u) & \text{, if } \mathrm{par}(x_g) = 1 \end{cases}.$$

在此处及全文中，“+”和“·”表示标准整数加法和乘法，而“∘”表示比特串的串接。

我们观察混合编码中扩展码字的两个性质。首先，考虑一个不涉及二进制反射格雷码部分向上计数的区间，即 i 是$(k+1)$的倍数，或者 I 中没有元素是$(k+1)$的倍数。此时，扩展码字的二进制反射格雷码部分没有M，且一元部分[7] 具有稳定前缀和后缀。

观察 6.3。对于 $i \in [2^n(k+1)]$ 和 $p \leq k$，令 $I = \langle i,\ i+p\rangle$。如果存在 α，$\ell \in \mathbb{N}$ 使得 $i = \ell(k+1) + \alpha$ 且 $\alpha + p \leq k$，那么

$$(x_I)_g = \gamma_{n,k}^{\mathrm{h}}(i)_g = \gamma_n^{\mathrm{g}}(\ell)\,, \textit{and}$$
$$(x_I)_u = *\,\gamma_k^{\mathrm{u}}(\langle\alpha, \alpha+p\rangle) = b^{\alpha}\mathtt{M}^{p}(\bar{b})^{k-\alpha-p}\,,$$

其中 $b = \overline{\mathrm{par}((x_I)_g)}$。

证明：假设存在 α，$\ell \in \mathbb{N}$ 使得 $i = \ell(k+1)+\alpha$且 $\alpha + p \leq k$。因此，

$$\lfloor i'/(k+1)\rfloor = \ell \text{ for each } i' \in \langle i, i+p\rangle\,.$$

这已经证明了第一个结论，即区间中的每个元素都有相同的二进制反射格雷码部分。此外，我们得到 $\mathrm{par}(\gamma_{n,\ k}^{\mathrm{h}}(i')_g) = \mathrm{par}(\gamma_{n,\ k}^{\mathrm{h}}(i)_g)$，即，一元部分使用相同类型的一元编码。第二部分由定理 5.3 推出。 □

例如，考虑

$n = 4\ \ k = p = 4$ $i = 25$ =、和 ()，使得
5 4 + 1 $I = \{25, \ldots, 29\}$ $\alpha = 0$
and 。我们在表 5 中将其标记为灰色区域。叠加得到扩展码字 $x_I = 0111\mathtt{MMMM}$。

其次，对于二进制反射格雷码部分中每次向上计数的区间，扩展一元部分至少有一个稳定比特。

观察 6.4。设 $I = \langle i, i+p\rangle$ 和 $p \leq k$ 满足存在一个元素 $j \in \langle i+1,\ i+p\rangle$，它是$(k+1)$的倍数。那么，

$$(x_I)_g = \gamma_{n,k}^{\mathrm{h}}(i)_g * \gamma_{n,k}^{\mathrm{h}}(j)_g\,, \textit{and}$$
$$(x_I)_u = \mathtt{M}^{\beta} b^{k-\alpha-\beta}\mathtt{M}^{\alpha}\,,$$

其中 $b = \overline{\gamma_{n\,k}^{\mathrm{h}}\ i\ _g}$ $\alpha = |\langle i\ j-1\rangle| - 1$ $\beta = |\langle j\ i+p\rangle| - 1$ $p \leq \lceil k/2\rceil$ $\alpha \leq \lceil k/2\rceil$ $\mathrm{par}((,())),$,和,。特别地，如果我们得到 和 $\beta \leq \lceil k/2\rceil$，使得$(x_u)_{\lceil k/2\rceil} = b$。此外，根据定理 *3.8*，$(x_I)_g$ 恰好有一个 M 位。[8]

证明：</style}至多有一个元素 可以是$(k+1)$的倍数。根据混合码的定义可知，所有小于 j 的元素共享相同的二进制反射格雷码部分，所有大于或等于 j 的元素也共享相同的二进制反射格雷码部分。因此，命题的第一个等式得证。

接下来，我们证明 $j-1$ 和 j的码字的一元部分相等。由定理 5.10可知，$\mathrm{par}(\gamma_{n,k}^{\mathrm{h}}(j-1)_g) = \mathrm{par}(\gamma_{n,k}^{\mathrm{h}}(j)_g)$，也就是说，从 $j-1$变为 j时，奇偶性会翻转。因此，我们会在一元编码的两种类型之间切换。我们注意到，一元部分编码的数分别为 $j-1 \bmod (k+1) = k$和 $j \bmod (k+1) = 0$。根据一元码的定义，即 $\gamma_k^{\mathrm{u}}(k) = \gamma_k^{\mathrm{u}}(0)$ 且 $\gamma_k^{\mathrm{u}}(k) = \gamma_k^{\mathrm{u}}(0)$，可得 $\gamma_{n,k}^{\mathrm{h}}(j-1)_u = \gamma_{n,k}^{\mathrm{h}}(j)_u$。

最后，由定理 5.3 可知，$(x_{\langle i,j-1\rangle})$ $b^{k-\alpha}\ \alpha$ $\beta b^{k-\beta}$ $_u$ = M和 $(x_{\langle j,i+p\rangle})$ $_u$ =M。因此，两者的叠加结果为 $\mathtt{M}^{\beta}b^{k-\alpha-\beta}\mathtt{M}^{\alpha}$。由此，第二个等式得证。 □

例如，考虑 $n = 4$、$k = p = 4$ 和 $i = 18$，使得 $I = \{18, \ldots, 22\}$、$j = 20$、$\alpha = 1$ 和 $\beta = 2$。我们将其标记为

7. 为方便起见，在本文余下部分中，我们将一元温度计部分称为一元部分。
8. F or a bi t $x \in \mathbb{B}$, we d enote its i nvers ion, i.e., $1-x$, by $\bar{x}$.

表 5（混合码的一部分）
$\gamma^{\mathrm{h}}_{4,\ 4}$。

i	$\gamma^{\mathrm{h}}_{4,4}(i)$	i	$\gamma^{\mathrm{h}}_{4,4}(i)$	i	$\gamma^{\mathrm{h}}_{4,4}(i)$	i	$\gamma^{\mathrm{h}}_{4,4}(i)$	i	$\gamma^{\mathrm{h}}_{4,4}(i)$	i	$\gamma^{\mathrm{h}}_{4,4}(i)$
	⋮	15	0010 1111	20	0110 0000	25	0111 1111	30	0101 0000	35	0100 1111
	⋮	16	0010 0111	21	0110 1000	26	0111 0111	31	0101 1000	36	0100 0111
12	0011 1100	17	0010 0011	22	0110 1100	27	0111 0011	32	0101 1100	37	0100 0011
13	0011 1110	18	0010 0001	23	0110 1110	28	0111 0001	33	0101 1110		
14	0011 1111	19	0010 0000	24	0110 1111	29	0111 0000	34	0101 1111	⋮	

表 5 中的灰色区域。叠加得到扩展码字 $x = \mathtt{0M10MM0M}$。

6.2 混合码的性质

在本节中，我们证明混合码 $\gamma^{\mathrm{h}}_{n,\ k}$ 是 k-保持的，并且是 $\lceil k/2 \rceil$-可恢复的。因此，混合码能够保持大小最多为 k 的区间的精度。然而，该编码只能从值域最多为 $\lceil k/2 \rceil$ 的扩展码字中恢复码字。

引理 6.5。对于 $n,\ k \in \mathbb{N}$，编码 $\gamma^{\mathrm{h}}_{n,\ k}$ 是 k-保持的。

证明：令 $I = \langle i,\ i+p \rangle$，其中 $i \in [2^n(k+1)]$，且 $p \le k$ 满足 $i+p \le 2^n(k+1)$。我们分两种情况讨论，分别对应定理 6.3 和定理 6.4 的条件。

首先，假设 i 是 $k+1$ 的倍数，或者 I 中没有元素是 $k+1$的倍数。根据定理 6.3，BRGC部分中没有M。所有M都出现在一元部分中。结论由 γ^{u}_k 和 γ^{u}_k 均为 k-保持这一事实得出；见定理 5.4。

其次，假设存在一个 $j \in I$，使得对于某个 $\ell \in \mathbb{N}$，$j = \ell(k+1)$。根据定理 6.4，BRGC部分等于 i 和 j的编码的叠加；

$$(x_I)_g = \gamma^{\mathrm{h}}_{n,k}(i)_g * \gamma^{\mathrm{h}}_{n,k}(j)_g\,.$$

从 i 到 j，BRGC部分进行一次向上计数。根据格雷码性质，$(x_I)_g$ 的解析仅包含 i 和 j各自对应的部分；

$$\mathrm{res}(*(\gamma^{\mathrm{h}}_{n,k}(I)_g)) = \{\gamma^{\mathrm{h}}_{n,k}(i)_g, \gamma^{\mathrm{h}}_{n,k}(j)_g\}\,.$$

定理 6.4 还指出，一元部分具有如下形式

$$(x_I)_u = \mathtt{M}^{\beta} b^{k-\alpha-\beta} \mathtt{M}^{\alpha}\,,$$

其中 $\alpha = |\langle i, j-1\rangle| - 1$ 且 $\beta = |\langle j, i+p\rangle| - 1$。$(x_I)_u$ 的解析包含将每个 M 替换为 0 或 1 的所有字。并非每个解析都是码字。如果 BRGC部分解析为 $\gamma^{\mathrm{h}}_{n,k}(i)_g$，则 $(x_I)_u$ 的解析中的码字是 $\langle k-\alpha, k\rangle$的码字。否则，如果 BRGC部分解析为 $\gamma^{\mathrm{h}}_{n,k}(j)_g$，则 $(x_I)_u$ 的解析中的码字是 $\langle 0, \beta\rangle$的码字。我们可以得出结论：x_I的解析不会添加$\gamma^{\mathrm{h}}_{n,k}(I)$之外的任何码字。 □

我们证明该混合码是 $\lceil k/2 \rceil$-可恢复的。我们通过表明存在解码函数的一个扩展，将扩展码字的所有解析映射回原始值域，来证明可恢复性。我们通过根据 BRGC部分的奇偶性映射一元部分来定义该扩展。一元码的映射由定理 5.5 中定义的函数 映射$^{\mathrm{u}}_k$ 完成。

引理 6.6。$\gamma^{\mathrm{h}}_{n,\ k}$ 是 $\lceil k/2 \rceil$-可恢复的。

证明：设 $I = \langle i,\ i+p \rangle$，对于 $i \in [2^n(k+1)]$ 以及满足 $i+p \le 2^n(k+1)$ 的 $p \le k$。我们证明，解码函数存在一个扩展，可将 x_I 的每个解析映射回区间 I。

扩展 $(\widetilde{\gamma}^{\mathrm{h}})^{-1}$：$\mathbb{B}^{n+k} \to [M]$ 定义如下：

$$(\widetilde{\gamma}^{\mathrm{h}})^{-1}(x) \coloneqq (\gamma^{\mathrm{h}}_{n,k})^{-1}(x_g \circ \mathrm{map}^{\mathrm{u}}_k(\mathrm{par}(x_g), x_u))\,.$$

我们考虑与定理 6.3 和定理 6.4 的条件相对应的两种情况。

首先，假设存在 $\alpha,\ \ell \in \mathbb{N}$ 使得 $i = \ell(k+1) + \alpha$且 $\alpha + p \le k$。根据定理 6.3，$(x_I)_g = \gamma^{\mathrm{g}}_n(\ell)$ 不含 M。根据解码函数（参见式 (1)），只要一元部分映射到 $\langle \alpha,\ \alpha+p\rangle$中的一个数，结论便成立。一元部分的映射取决于 BRGC部分的奇偶性。令 π 表示$(x_I)_g$的奇偶性。如果能够证明一元部分映射到 $\gamma^{\mathrm{u}}_k(\langle \alpha,\ \alpha+p\rangle)$中的一个码字，那么第一种情况下结论便成立。同样，根据定理 6.3，$(x_I)_u = *\gamma^{\mathrm{u}}_k(\langle \alpha,\ \alpha+p\rangle)$。因此，由定理 5.3 可得结论。

其次，我们假设存在 $j \in \langle i+1, i+p\rangle$ 使得 $j = \ell(k+1)$。根据定理 6.4，$(x_I)_g$ 的解析包含 $\gamma^{\mathrm{h}}_{n,k}(i)_g$ 和 $\gamma^{\mathrm{h}}_{n,k}(i+1)_g$。不失一般性，假设 $\gamma^{\mathrm{h}}_{n,k}(i)_g$ 具有偶校验，并且根据定理 3.8，$\gamma^{\mathrm{h}}_{n,k}(i+1)_g$ 具有奇校验。对于 $\gamma^{\mathrm{h}}_{n,k}(i)_g$ 具有奇校验的情形，证明思路相同。由定理 6.4 和 $p \le \lceil k/2 \rceil$，我们得到 $(y_u)_{\lceil k/2 \rceil} = 1$。根据定理 5.5，我们推出 (i) $\mathrm{map}^{\mathrm{u}}_k(0, y_u) = 1^{\ell^{\max}_1} 0^{k-\ell^{\max}_1}$，其中 $\ell^{\max}_1 \in \{k-(j-i)\ldots, k\}$；以及 (ii) $\mathrm{map}^{\mathrm{u}}_k(1, y_u) = 0^{\ell^{\min}_0} 1^{k-\ell^{\min}_0}$，其中 $\ell^{\min}_0 \in \{0, \ldots, p-(j-i)\}$。

由此可知，对于 x_I 的所有解析，其中 $(x_I)_g$ 等于 $\gamma^{\mathrm{h}}_{n,k}(i)_g$，一元部分映射为 $\gamma^{\mathrm{u}}_k(\ell)$，其中$\ell \in \langle i \bmod (k+1)\ j-1 \bmod (k+1)\rangle$。类似地，对于 x_I 的所有解析，其中 $(x_I)_g$ 等于 $\gamma^{\mathrm{h}}_{n,k}(j)_g$，一元部分映射为 $\gamma^{\mathrm{u}}_k(\ell)$；其中 $\ell \in \langle j \bmod (k+1)\ i+p \bmod (k+1)\rangle$。因此，借助扩展解码函数，$x_I$ 的每个解析都会得到 I 中的一个数。 □

7 k-可恢复加法

本节介绍我们实现的电路，该电路可在不精确度有界于 k 的情况下执行加法。为此，我们首先形式化定义 k 位亚稳态容纳和k-可恢复加法，即对不确定输入所导致的区间加法的忠实表示。

7.1 亚稳态容纳电路

这个概念的一种简便形式化表述如下。对于一个具有 C 个 n 输入和 m 个输出的电路 $C(x) \in \mathbb{T}^m$，用 $x \in \mathbb{T}^n$表示它在输入 $x \in \mathbb{T}^n$时计算出的输出。若对所有 $x \in \mathbb{B}^n$，都有 $C(x) = f(x)$，则称 C实现了布尔函数 $f\colon \mathbb{B}^n \to \mathbb{B}^m$。电路所需的行为由 f的亚稳态闭包给出。

定义 7.1 (亚稳态闭包 [1])。对于函数 $f\colon \mathbb{B}^n \to \mathbb{B}^m$，用 $f_{\mathtt{M}}\colon \mathbb{T}^n \to \mathbb{T}^m$ 表示其亚稳态闭包，其定义为 $f_{\mathtt{M}}(x)\colon = *_{y \in \mathrm{res}(x)} f(y)$。

亚稳态闭包是通过组合逻辑可计算的f 的“最精确”扩展。容易证明，对于实现 f的任意电路 C，$(f_{\mathtt{M}}(x))_i = \mathtt{M}$ 蕴含 $C_i(x) = \mathtt{M}$：限制为布尔输入时，C 和 $f_{\mathtt{M}}$ 是相同的；

将输入位更改为M只能将输出位更改为M，并且如果 $f_{\mathtt{M}}(x)$ 在位置 i 处为M，那么 $C(x)$ 在位置 i处为M。

定义 7.2（k 位亚稳态包含）。设电路 C 实现布尔函数 $f\colon \mathbb{B}^n \to \mathbb{B}^m$。若 C 计算出 f的亚稳态闭包，即对每个 $x \in \mathbb{T}^n$，都有 $C(x) = f_{\mathtt{M}}(x)$，则称 C 具有亚稳态包含（*mc*）性质。如果对于至多含有 $k \in \mathbb{N}$ 个M的每个 x，C 都具有 *mc* 性质，则称 C 具有 k位 *mc* 性质。

请注意，若 C 不具有 mc 性质，则等价于：对于某个 x，$C(x)$ 输出的 M 比必要的更多。一个未能实现亚稳态闭包的电路的最小非平凡示例，是对多路复用器电路的朴素实现。多路复用器的布尔规范为 MUX$(a, b, s) = a$当 $s = 0$时，以及 MUX$(a, b, s) = b$当 $s = 1$时。它可以通过对应于布尔公式 or(and(a, not(s)), and(b, s)) 的电路来实现，但这会导致输入 (1, 1, M) 处出现一个不必要的 M，参见 [5]。

备注。请注意，我们同时使用下标 u 和 M，其中x_u 表示字 x 中的某一部分，而 $f_{\mathtt{M}}$ 表示函数 f的亚稳态闭包。

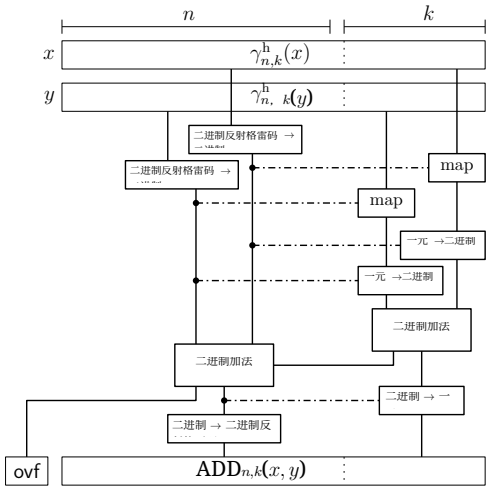

图5. 混合编码的稳定码字与非码字的加法。只有在对非码字进行加法时，才需要映射电路。点划线表示仅复制最低有效位。

7.2 k-可恢复加法

k-可恢复加法计算两个码字之和的编码，其中输入和输出使用相同的编码，且两个输入的不精确度均不超过k。我们将关于编码 γ 的加法记为 $+_\gamma$。实现 k-可恢复加法的电路必须实现亚稳态闭包$(+_\gamma)_{\mathtt{M}}$。如果发生溢出，我们允许输出任意值 $s \in \mathbb{T}^n$。

定义 7.3。给定一个 k-可恢复编码 $\gamma\colon [M] \to \mathbb{B}^n$。设$x, y \in \mathbb{T}^n$ 为 $I_x = \langle i, j\rangle$ 和$I_y = \langle i', j'\rangle$的扩展码字。假设 x 和 y 的不精确度分别为 p_x 和 p_y，且满足 $p_x + p_y \le k$。用运算符 $+_\gamma\colon \mathbb{B}^n \times \mathbb{B}^n \to \mathbb{B}^{n+1}$ 表示两个编码的加法。我们将 $x(+_\gamma)_{\mathtt{M}}y = s$ 的 k-可恢复 γ-加法以及溢出标志 *ovf* 定义如下；若没有溢出，即 $j + j' < M$，则

1) s 是 $\gamma([M])$ 中的扩展码字，2) s 的值域为$\langle i + i',\ j + j'\rangle$，3) s 的不精确度为 $p_s = p_x + p_y$，并且4) $ovf = 0$。

如果发生溢出，即 $j + j' \ge M$，那么 $ovf = 1$。

为了在电路中实现两个混合编码的 k-可恢复加法，我们首先讨论如何在布尔世界中高效地对混合编码进行加法。我们将二进制反射格雷码和一元部分转换为二进制码字，从而对两个混合码字进行加法；任何现成的二进制加法器都可以将这些码字相加。相加后，我们再将和的二进制表示转换回混合码字。最后，我们使用 [5] 中的构造来得到一个 mc电路。请注意，使用这一构造需要

我们来处理电路输入中的非码字。

7.3 稳定码字的加法

我们通过分别对 BRGC 部分和一元部分进行相加，来对稳定码字求和。相加之前，需要先将非码字映射为码字。通常会将 BRGC 码字转换为二进制码字，然后使用广为人知的二进制加法器。我们对一元码字也采用相同的技术。若能有一种无需转换为二进制码字、直接对混合码字进行相加的 mc 加法算法，将十分理想，因为这样可以减小电路的规模和深度。由于这两个部分之间存在双向依赖，无法采用简单的 mc 技术，因此这种算法并非易事。

图 5 展示了我们针对稳定码字和非码字的加法方案，记作 ADD$_{n,\,k}$。回顾一下，x_g、y_g 表示 BRGC 部分，x_u、y_u 表示一元温度计部分。

为了在二进制反射格雷码和二进制码之间相互转换，我们首先观察以下性质。

观察7.4。设 $a \in [2^n]$和 $i \in [n]$，则对于编码 γ_n^{g} 和 γ_n^{b}，以下性质成立：

$$(\gamma_n^{\mathrm{b}}(a))_i = \mathrm{par}((\gamma_n^{\mathrm{g}}(a))_{1,i}),\ \textit{and}$$
$$(\gamma_n^{\mathrm{g}}(a))_i = (\gamma_n^{\mathrm{b}}(a))_{i-1}\ \textit{xor}\ (\gamma_n^{\mathrm{b}}(a))_i$$

利用定理 7.4，我们正式证明存在一个电路，可将 BRGC 转换为二进制编码。

引理7.5 (BRGC $\to$ 二进制)。存在一个大小为 $\mathcal{O}(n)$ 、深度为 $\mathcal{O}(\log n)$ 的电路，它以 $x \in \mathbb{B}^n$ 为输入，计算转换后的二进制表示。也就是说，它计算输出 $o = \gamma_n^{\mathrm{b}}((\gamma_n^{\mathrm{g}})^{-1}(x))$。

证明：根据定理 7.4，转换为二进制可以通过计算输入 x 的每个前缀的奇偶性来完成。利用并行前缀电路 [20]，可以高效地计算输入各个前缀上的结合运算。奇偶性可以通过对所有位进行异或运算来计算。由于异或满足结合律，因此可以通过并行前缀异或来完成转换，其大小为 $\mathcal{O}(n)$，深度为 $\mathcal{O}(\log n)$。□

将一元部分转换为二进制编码取决于 BRGC 部分。一元部分的类型根据 BRGC 部分的奇偶性来选择。我们可以注意到，在 BRGC 到二进制转换过程中已经计算了奇偶性，因此无需单独计算。二进制串的最低有效位表示 BRGC 部分的奇偶性。

观察7.6。设 $x \in [2^n]$，x的*BRGC*码的奇偶性由 $x = \gamma_n^{\mathrm{b}}\ x\ [n]$. 的二进制码的最后一位表示；par($\gamma_n^{\mathrm{g}}$()) ( )

利用奇偶性，我们可以根据一元编码的正确类型将一元编码转换为二进制编码。二进制编码的冗余较少，k 位的一元字可以表示为一个含有 $\lceil \log(k+1) \rceil$位的二进制字。

引理7.7 (一元 $\to$ 二进制). 存在一个大小为 $\mathcal{O}(k)$ 且深度为$\mathcal{O}(\log\ k)$ 的电路，对于 $\ell = \lceil \log(k+1) \rceil$、$x \in \gamma_k^{\mathrm{u}}([k+1])$以及 $\pi \in \mathbb{B}$,

计算到二进制的转换。也就是说，它计算输出 o，其中

$$o = \begin{cases} \gamma_\ell^{\mathrm{b}}((\gamma_k^{\mathrm{u}})^{-1}(x)) & \text{, if } \pi = 0 \\ \gamma_\ell^{\mathrm{b}}((\overline{\gamma}_k^{\mathrm{u}})^{-1}(x)) & \text{, if } \pi = 1\,. \end{cases}$$

证明： 根据定理 5.2，一元编码的一种类型是另一种对应类型的逆，$\gamma_k^{\mathrm{u}}(i) = \overline{\gamma_k^{\mathrm{u}}}(i)$。因此，我们可以通过使用$(\gamma_k^{\mathrm{u}})^{-1}$来计算解码函数$(\gamma_k^{\mathrm{u}})^{-1}$，如下所示：$(\overline{\gamma_k^{\mathrm{u}}})^{-1}(x) = (\gamma_k^{\mathrm{u}})^{-1}(x)$。因此，这等价于计算输出

$$o = \gamma_\ell^{\mathrm{b}}((\gamma_k^{\mathrm{u}})^{-1}(\pi \mathbin{\mathsf{xor}} x))\,,$$

其中，我们计算 π 与 x的每一位的异或。

转换 $\gamma_\ell^{\mathrm{b}}((\gamma_k^{\mathrm{u}})^{-1}(\cdot))$ 可以通过递归定义的电路来实现。不失一般性，我们假设 $k+1$ 是2的幂。[9] 令 $m = \lceil k/2 \rceil$ 为 x中间那一位的索引。根据 $x[m]$，电路递归地处理 m左侧或右侧的所有位。因此，令 $m_l = \lfloor k/2 \rfloor$ 和$m_r = \lceil k/2 \rceil + 1$ 分别为中间索引左侧和右侧的索引。我们定义电路 δ_k，对于 k位输入 x，计算一个 ℓ位输出。当 $k = 1$ 时，电路复制输入 $\delta_1 = x$。当 $k > 1$ 时，电路定义如下：

$$\begin{aligned} \delta_k[1] &\coloneqq x[m]\,, \\ \delta_k[2,\ell] &\coloneqq \mathsf{MUX}(\delta_{\ell-1}(x[1,m_l]), \delta_{\ell-1}(x[m_r,k]), x[m])\,. \end{aligned}$$

电路 δ_k 实现了一元到二进制转换。如果$x[m] = 1$，根据定理 5.2，我们知道 x 编码的值大于或等于 $\lceil k/2 \rceil$。因此，二进制输出的最高有效位设为1。如果 $x[m] = 0$， x 编码的值小于 $\lceil k/2 \rceil$，且最高有效位设为0。如果 $x[m] = 1$，索引 m 左侧的所有位都是1，且递归调用需要位 m_r 到 k。类似地，如果 $x[m] = 0$，m 右侧的所有位都是0，且递归调用需要0 到 m_l位。一元到二进制转换电路的正确性可通过归纳论证得出。

计算该转换的电路可以通过一层由 k 个异或门组成的电路和一棵多路复用器（MUX）树来实现。由于 MUX 的大小和深度均为常数，因此结论成立。 □

使用大小为 n 和 $\log(k+1)$ 的二进制加法器，我们可以分别对这两个部分进行相加。一元部分产生的进位必须传递到 BRGC 部分的加法中。BRGC 部分产生的进位表示电路发生溢出，我们可以将其传递到输出 ovf。

相加之后，还需要将两个部分分别转换回各自的编码。BRGC 部分的转换很容易，因为二进制码可以通过单层异或门转换为 BRGC。

引理 7.8 (二进制 $\to$ BRGC)。 存在一个大小为 $\mathcal{O}(n)$ 、深度为 $\mathcal{O}(1)$ 的电路，该电路以 $x \in \mathbb{B}^n$ 为输入，计算到 *BRGC* 的转换。也就是说，它计算输出 $o = \gamma_n^{\mathrm{g}}((\gamma_n^{\mathrm{b}})^{-1}(x))$。

证明： 根据定理 7.4，反向转换可以通过单层异或门来实现。该电路的大小为 $\mathcal{O}(n)$，深度为 $\mathcal{O}(1)$。 □

转换为正确的一元类型，同样取决于 BRGC部分的奇偶性。正如定理 7.6所述，奇偶性已由二进制码字的最低有效位指示。

9. 如果 $k+1$ 不是2的幂，我们可以补充足够数量的比特，使其达到下一个2的幂，而不会影响渐近复杂度。

引理 7.9 (二进制 $\to$ 一元)。 存在一个大小为 $\mathcal{O}(k)$ 、深度为$\mathcal{O}(\log k)$ 的电路，对于 $\ell = \lceil \log(k+1) \rceil$、 $x \in \mathbb{B}^\ell$ 和 $\pi \in \mathbb{B}$，可计算二进制到一进制的转换，

$$o = \begin{cases} \gamma_k^{\mathrm{u}}((\gamma_\ell^{\mathrm{b}})^{-1}(x)) & \text{, if } \pi = 0 \\ \overline{\gamma}_k^{\mathrm{u}}((\gamma_\ell^{\mathrm{b}})^{-1}(x)) & \text{, if } \pi = 1\,. \end{cases}$$

证明： 该电路与定理 7.7中的电路类似。我们可以按如下方式使用异或来计算第二种一元编码：

$$o = \pi \mathbin{\mathsf{xor}} \gamma_k^{\mathrm{u}}((\gamma_\ell^{\mathrm{b}})^{-1}(x))\,.$$

不失一般性，我们假设 $k+1$ 是2的幂。[9] 我们定义一个递归电路 δ_ℓ 来计算一元编码。对于 $\ell = 1$，电路定义为 $\delta_\ell(x) = x$。对于 $\ell > 1$，令 $\delta_{\ell-1}(x[2,\ell]) = y$ 为递归子电路的结果。电路按如下方式向 y 添加 $2^{\ell-1}$个0或1：

$$\delta_\ell \coloneqq \mathsf{MUX}(y \circ 0^{2^{\ell-1}}, 1^{2^{\ell-1}} \circ y, x_1)\,,$$

该电路计算从二进制到一元的转换。如果二进制字的第 i位为1，那么电路就在输出左侧添加 2^{i-1}个1。因此，输出中1的总数为 $\sum_{i=1}^{\ell} 2^{i-1} \cdot x_i$。这与二进制编码的值相符。

该电路在每次递归调用时添加一个多路复用器，并在末尾添加一层异或门。因此，该电路的规模渐近线性，深度为 k的对数级。 □

接下来，我们将每个部分转换为二进制，分别对各部分求和，再转换回相应编码，以此实现两个混合码字的加法。$\mathrm{ADD}_{n,k}$ 加法方案实现了混合编码的加法。

引理 7.10. 给定 $x,y \in \gamma_{n,k}^{\mathrm{h}}([M])$ 加法方案 $\mathit{ADD}_{n,k}$ 是正确的；也就是说 $\mathit{ADD}_{n,k}(x,y) = x +_{\gamma^{\mathrm{h}}} y$.

证明：我们分别证明该结论对二进制反射格雷码部分和一元部分成立。设 $i = (\gamma_{n,k}^{\mathrm{h}})^{-1}(x)$ 和 $j = (\gamma_{n,k}^{\mathrm{h}})^{-1}(y)$，接下来证明 $\mathrm{ADD}_{n,k}(x,y) = \gamma_{n,k}^{\mathrm{h}}(i+j)$。首先，我们证明 $\mathrm{ADD}_{n,k}(x,y)_u = (x +_{\gamma^{\mathrm{h}}} y)_u$。根据 x_g 和 y_g 的奇偶性，共有四种情况。对于每种情况，我们都必须根据一元编码的类型选择相应的转换。根据定理 7.5 和定理 7.6，我们可以得到 x_g 和 x_y 的正确奇偶位。因此，转换后（见定理 7.7），我们得到 $i \bmod (k+1)$ 和 $j \bmod (k+1)$ 的二进制表示。加法的正确性由二进制加法器的正确性推出，因此我们得到 $(i+j) \bmod (k+1)$ 的二进制表示。根据定理 7.6 和定理 7.9，我们可以得出，转换为一元编码时会选择正确的类型。因此，$\mathrm{ADD}_{n,k}(x,y)_u = \gamma_{n,k}^{\mathrm{h}}(i+j)_u$，这就证明了该结论对一元部分成立。

其次，我们证明 $\mathrm{ADD}_{n,k}(x,y)_g = (x +_{\gamma^{\mathrm{h}}} y)_g$。根据定理 7.5，将 BRGC 部分转换为二进制，可得到 $\lfloor i/(k+1) \rfloor$ 和 $\lfloor j/(k+1) \rfloor$的二进制表示。将一元部分相加后，我们得到一个进位，它指示 $i \bmod (k+1) + j \bmod (k+1) > k+1$ 是否成立。因此，将两个输入与进位进行二进制加法，得到 $\lfloor (i+j)/(k+1) \rfloor$的二进制表示。最后，根据定理 7.8，我们得到 $\mathrm{ADD}_{n,k}(x,y)_g = \gamma_n^{\mathrm{g}}(\lfloor (i+j)/(k+1) \rfloor) = \gamma_{n,k}^{\mathrm{h}}(i+j)_n$。

□

给定每个转换电路的正确实现，定理 7.10 意味着存在一个用于两个混合码相加的电路。

推论 7.11。 存在一个规模为 $\mathcal{O}(n+k)$ 、深度为 $\mathcal{O}(\log n + \log k)$ 的电路，可实现混合编码的加法。

备注。 该方案远非 mc。第 1 节和第 3 节中的例子表明，二进制编码甚至不是1-保持的。由于 ADD$_{n,k}$ 加法方案严重依赖二进制编码，因此它也存在二进制编码的缺点。

7.4 扩展码字的加法

我们可以在上述加法方案的基础上，应用用于 mc 电路的通用构造来相加扩展码字。

使用这种构造时，我们需要考虑到电路必须处理非码字输入，因为扩展码字的解析可能会得到非码字。因此，我们使用一个实现$(\widetilde{\gamma}^{\mathrm{h}})^{-1}$ 的电路（来自定理6.6的证明），以获得可通过 ADD$_{n,\ k}$ 方案相加的码字。扩展$(\widetilde{\gamma}^{\mathrm{h}})^{-1}$ 会根据 BRGC 部分的奇偶性，对一元部分应用映射$^{\mathrm{u}}_{k}$ 。因此，我们需要一个在二进制转换之前对 x_u和 y_u 应用映射$^{\mathrm{u}}_{k}$ 的电路。

该电路可以通过计算定理 5.5 的所有四种情况，并使用输入 $x_{\lceil k/2\rceil}$ 和 π 通过多路复用器选择正确的输出而构建。

引理7.12。 给定输入 $x \in \mathbb{B}^k$ 和 $\pi \in \mathbb{B}$，存在一个大小为 $\mathcal{O}(k)$ 、深度为 $\mathcal{O}(\log\ k)$ 的电路，用于计算 映射$^{\mathrm{u}}_{k}(\pi, x)$。

证明： 映射 映射$^{\mathrm{u}}_{k}$ (π, x) 在定理5.5中定义。四种情况均可由并行前缀电路（PPC） [20]计算。然而，这些情况需要采用不同策略：使用与或或作为并行前缀运算，并从左向右或从右向左读取输入。各策略列于下表。

π	$(x_u)_{\lceil k/2\rceil}$	运算	方向
0	0	PPC-与	从左到右
0	1	PPC-或	从右到左
1	0	PPC 和 并行前缀电路或	从右到左
1	1		从左到右

我们可以并行计算全部四种情形。随后由一个多路复用器根据选择比特 π 和 $(x_u)_{\lceil k/2\rceil}$ 选出结果。完整电路由四个并行的 PPC 电路和一个 k 位 (4:1)-多路复用器组成。因此，该电路关于 k 具有渐近线性规模和对数深度。 □

我们可以利用已知的 mc 电路构造，将本节给出的电路变换为执行可恢复加法的加法器。

推论 7.13 （出自 [5]）。 对于 $n,\ k \in \mathbb{N}$ 存在一个规模为 $(n+k)$ $^{\mathcal{O}(k)}$、深度为 $\mathcal{O}\left(k\log\ n+\log\ k\right)$ 的电路，它实现 $\lceil k/2\rceil$-可恢复加法。

证明。 我们将 Ikenmeyer 等人的构造应用于 ADD$_{n,\ k}$ 与 映射$^{\mathrm{u}}_{k}$ 的组合。由 [5], , 这实现了 $\gamma^{\mathrm{h}}_{n,\ k}$- 加法的亚稳态闭包，前提是不稳定的比特不超过 k 个。由于 $\gamma^{\mathrm{h}}_{n,\ k}$ 编码是 k-保持且$\lceil k/2\rceil$-可恢复的，因此这是 $\lceil k/2\rceil$-可恢复加法的正确实现。

□

表2给出了 $n=5$ and $k=3$ 的一次电路执行示例。该示例展示了不精确度最高为2的编码相加。我们使用 VHDL 实现了该电路，并使用x 进行仿真，以模拟亚稳态信号的最坏情况行为。

同样，依靠屏蔽寄存器，我们可以得到一个效率更高的电路。

推论 7.14 (关于 [1])。 对于 $n,\ k \in \mathbb{N}$ 存在一个规模为$\mathcal{O}(k(n+k))$、深度为 $\mathcal{O}(k\log\ n+\log k)$ 的电路，实现 $\lceil k/2\rceil$-可恢复加法。

证明： 与定理 7.13 的证明类似，但使用来自 [1] 的构造。 □

8 其他相关工作

不精确编码

格雷码应用广泛，因为它具有1-可恢复性，例如可用于从小的测量误差中恢复。回顾来看，Friedrichs 等人 [1] 指出，格雷码具有1-可恢复性。然而，据我们所知，我们是首个定义 k-可恢复性这一性质的。在 [3], 中，研究表明，通过实际电路可以获得格雷码时间测量。令人惊讶的是，在转换为1-可恢复格雷码之前，测量电路会将与本文所述编码相同的 k-可恢复编码作为中间结果输出。这些编码在这一背景下自然出现，其原因截然不同，这表明它们具有普遍意义。

亚稳态容忍电路

除了关于亚稳态包含（又称无冒险）电路的经典研究外，近期人们对这一领域也表现出浓厚兴趣。Ikenmeyer 等人 [5] 研究了无冒险电路的复杂度，并证明了若干结果，其中包括：无冒险电路复杂度自然地推广了单调电路复杂度，而且布尔电路复杂度与无冒险电路复杂度之间可能存在很大差距。他们研究中的一项核心设计工具是通用变换：给定一个布尔电路以及可能处于亚稳态的输入位数量的上界k ，该变换会生成一个对所有此类输入均为 mc 的电路。该构造带来的规模开销为 $n^{O(k)}$；例如，当 k 为常数时，规模增长相对于输入规模是多项式级的。对固定且有界数量的不稳定位进行推测，也是 Tarawneh 等人 [21],所提出的亚稳态容忍设计风格的核心；他们将这类推测计算融入同步状态机。可以将他们的工作视为系统层面上的一个实例，其基本思想与启发 Ikenmeyer 等人提出该变换的思想相同。

近期还有一系列关于无冒险电路和公式的规模与结构的研究。Jukna [22] 分析了无冒险这一要求如何增大电路规模，并给出了在这一约束下计算代价显著增加的显式函数。Ikenmeyer、Komarath 和 Saurabh [23]构建了无冒险的 Karchmer-Wigderson 框架，并得到了无冒险公式的精细界；例如，他们表明，即使是多路复用器这样简单的函数，在无冒险设定下也可能需要规模大得多的公式。最近，London、Arazi 和 Shpilka [24]研究了一般布尔函数（包括随机函数）的无冒险公式，并确定了已知下界技术何时是紧的。总的来说，这些研究表明，无冒险计算和亚稳态感知计算是一个活跃且不断发展的研究方向。

9 结论

存在可能的位错误（如亚稳态）时，二进制加法便失去了意义。我们提出了一种新的编码方案及相应的加法过程，即使输入存在有界的总不精确度，也能保持精度。我们的混合码$\gamma^{\mathrm{h}}_{n,\ k}$ 使用 k 个冗余位，并能保持 k，但只能达到 $\lceil k/2\rceil$ 可恢复性：最多可容忍 k 个不精确或处于亚稳态的输入位，而不会增加输出不精确性，但其中最多只有 $\lceil k/2\rceil$ 个

可以确定性地恢复。这种可容忍的不精确度与有保证的可恢复性之间的不对称性，反映了一个固有差距；我们将其留作未来研究的一个有趣方向。

我们给出了匹配下界，证明对于总不精确度的固定上限 k，我们的编码具有最优的渐近码率。此外，利用已知的亚稳态容纳电路黑盒构造，我们得到了亚稳态容纳加法器和纯组合加法器，可针对这些编码实现区间加法。对于常数 k，与标准二进制加法器相比，这只会带来多项式级的电路规模开销，而不会（渐近地）增加其深度。

从应用角度来看，我们的框架表明，具有有界不确定性的数字加法不必退化为延迟随 n线性增长的一元编码。特别是在不稳定输入由亚稳态引起，且可以接受一定程度输出不精确性的场景中（例如某些控制回路），只要总输入不精确度保持在规定上限 k以内，我们的编码和加法器就能计算出正确的区间和。我们推测，将此类编码与基于同步器的设计和/或屏蔽存储元件结合使用，可以改善延迟与平均故障间隔时间权衡，相较于纯粹基于同步器的方法更具优势。然而，要对此进行定量论证，还需要经过优化的电路实现和布局后分析，这将留待未来研究。

致谢

感谢 Matthias F¨ugger 提供的宝贵讨论。

参考文献

[1] S. Friedrichs、M. F¨ugger 和 C. Lenzen，《亚稳态容纳电路》，*IEEE*计算机汇刊，第67卷，第8期，2018年。[2] J. Bund、C. Lenzen 和 M. Medina，《小型无冒险传导器》，*IEEE*计算机汇刊，2025年。[3] M. F ¨ugger、A. Kinali、C. Lenzen 和 T. Polzer，《亚稳态感知的内存高效时间数字转换器》，载于第*23*届 *IEEE* 异步电路与系统国际研讨会，2017年。[4] J. Bund、C. Lenzen 和 M. Medina，《通过并行前缀计算实现最优亚稳态包含排序》，*IEEE*计算机汇刊，第69卷，第2期，第198–211页，2020年。[在线]。获取地址：https://doi.org/10 1109 2019 2939818 。/TC。C. Ikenmeyer、[5] B. Komarath、C. Lenzen、V. Lysikov、A. Mokhov 和 K. Sreenivasaiah，《无冒险电路的复杂性》，《美国计算机协会会刊》，第66卷，第4期，第25:1–25:20页，2019年。[6] R. Ginosar，《亚稳态与同步器：教程》，*IEEE*计算机设计与测试，第28卷，第5期，第23–35页，2011年。[7] L. R. Marino，《异步输入对时序网络可靠性的影响》，*IEEE* 计算机汇刊，第100卷，第11期，第1082–1090页，1977年。[8] R. Ginosar，《欺骗同步器的十四种方法》，载于第九届异步电路与系统国际研讨会论文集，*2003*年。IEEE，2003年，第89–96页。[9] S. Beer、R. Ginosar、J. Cox、T. Chaney 和 D. M. Zar，《65纳米及更先进工艺面临的亚稳态挑战：仿真与测量》，载于*2013*年欧洲设计、自动化与测试会议暨展览会（*DATE*）。IEEE，2013年，第1297–1302页。[10] D. J. Kinniment，《数字系统中的同步与仲裁》。威利出版社，2008年。[11] D. A. Huffman，《无冒险开关网络的设计与应用》，《美国计算机协会会刊》，第4卷，第1期，第47–62页，1957年。[12] S. Srinivas、I. W. Jones、G. Panic 和 C. Lenzen，《基于锁存器的快速电压跌落响应设计》，2025年。 [在线]。获取地址：https://arxiv.org/abs/2501 18843 。J. Lundelius Welch 和 [13] N. A. Lynch，《一种新的时钟同步容错算法》，信息与计算，第 77 卷，第1期，第1–36页，1988年。[14] S. C. Kleene，《元数学导论》。北荷兰出版社，1952年。[15] S. H. Caldwell，《开关电路与逻辑设计》。John Wiley & Sons Inc，1958年。[16] S. K¨orner，《经验与理论：科学哲学论文》，《国际哲学与科学方法丛书》。伦敦：Routledge & Kegan Paul，1966年。[17] W. Hu、J. Oberg、A. Irturk、M. Tiwari、T. Sherwood、D. Mu 和 R. Kastner，《生成门级信息流跟踪逻辑的复杂性》，*IEEE* 信息取证与安全汇刊，第7卷，第3期，第1067–1080页，2012年。[18] M. Tiwari、H. M. Wassel、B. Mazloom、S. Mysore、F. T. Chong 和 T. Sherwood，《从门电路开始的完整信息流跟踪》，*SIGARCH* 计算机体系结构新闻，第37卷，第1期，第109–120页，2009年。[19] R. W. Hamming，《检错与纠错编码》，贝尔系统技术期刊，第29卷，第2期，第147–160页，1950年。[20] R. E. Ladner 和 M. J. Fischer，《并行前缀计算》，《美国计算机协会会刊》（*JACM*），第27卷，第4期，第831–838页，1980年。[21] G. Tarawneh、M. F¨ugger 和 C. Lenzen，《亚稳态容错计算》，载于第*23*届 *IEEE* 异步电路与系统国际研讨会，*ASYNC 2017*，美国加利福尼亚州圣迭戈，*2017*年*5*月*21*日至*24*日。IEEE 计算机学会，2017年，第25–32页。 [在线]。获取地址：https://doi.org/10 1109 2017 9 。/ASYNC。S. Jukna，[22] 《无冒险电路札记》，*SIAM* 离散数学杂志，第35卷，第2期，第770–787页，2021年。 [在线]。获取地址：https://doi.org/10 1137 20 1355240 。/ M C. Ikenmeyer、B. [23] Komarath 和 N. Saurabh，《无冒险计算的 Karchmer-Wigderson 博弈》，载于第*14*届理论计算机科学创新会议，*ITCS 2023*，美国马萨诸塞州剑桥市麻省理工学院，*2023*年*1*月*10*日至*13*日，LIPIcs丛书，Y. T. Kalai 编，第251卷。Schloss Dagstuhl - Leibniz-Zentrum f¨ur Informatik，2023年，第74:1–74:25页。 [在线]。获取地址：https://doi.org/10 4230 2023 74 。/LIPIcs.ITCS。[24] L. L. Arazi 和 A. Shpilka，《无冒险公式的复杂性》，载于第*52*届自动机、语言与编程国际研讨会，*ICALP 2025*，丹麦奥胡斯，*2025*年*7*月*8*日至*11*日，LIPIcs丛书，K. Censor-Hillel、F. Grandoni、J. Ouaknine 和 G. Puppis 编，第334卷。Schloss Dagstuhl - Leibniz-Zentrum f¨ur Informatik，2025年，第115:1–115:19页。 [在线]。获取地址：https://doi.org/10.4230/LIPIcs.ICALP.2025.115