用于纠正 t 幅度受限粘性删除的码

Shuche Wang*、Van Khu Vu§和Vincent Y. F. Tan†‡*
* 新加坡国立大学运筹学与分析研究所，新加坡
† 新加坡国立大学数学系，新加坡
‡ 新加坡国立大学电气与计算机工程系，新加坡§ 新加坡国立大学工业与系统工程及管理系，新加坡
电子邮箱：shuche.wang@u.nus.edu、isevvk@nus.edu.sg、vtan@nus.edu.sg

arXiv:2302.02754v1 [cs.IT] 6 Feb 2023

摘要——由于在闪存、赛道存储器和 DNA 数据存储系统中的应用，用于纠正粘性插入/删除和有限幅度错误的码受到了广泛关注。本文首先研究 t幅度受限的 -粘性删除这一错误类型，并提出一种非系统码来纠正此类错误，其冗余为 $2t(1-1/p)\cdot\log(n+1)+O(1)$，其中 p 是大于 $\ell+1$的最小素数。接下来，我们提出一种系统码构造，具有高效的编码和解码算法，其冗余为 $\lceil 2t_{(1-1/p)\cdot\lceil\log p\rceil\log p}\log(n+1)+O(\log\log n)$，其中 p 是大于 $\ell+1$的最小素数。

I. 引言

近年来，闪存、赛道存储器和 DNA 数据存储系统等数据存储技术的编码技术备受关注。与传统的通信与存储系统不同，替换并非这些新兴存储系统中的主要错误类型。文件同步、赛道存储器 [1] 和 DNA 数据存储 [2]，[3]中普遍存在粘滞插入和删除。此外，闪存中也频繁发生幅度受限错误，因为信息存储在相应的单元电平中 [4]。此外，在某些 DNA 数据存储系统中，部分信息由游程长度表示 [5] ，而在 DNA 序列合成过程中可能发生幅度受限错误。

粘性删除/插入纠错问题最早由 Levenshtein 研究，他提出了一种构造方法，用于纠正 r [6] 中的粘性插入/删除。Levenshtein 还给出了纠正 r 粘性插入/删除[6]的码的最大规模的下界和上界。Dolecek 等人 [7] 提出了一种纠正 r粘性插入的码，其规模至多为 $2^{n+r}/n^r$，改进了 [6] 中的下界。除了非高效的码构造外，Mahdavifar 等人 [8] 还提出了一种渐近最优系统黏性插入纠错码。此外，粘性删除/插入和重复删除可通过格雷映射 [9]视为非对称删除/插入。Tallini 等人 [9]–[13] 提出了一系列用于纠正非对称删除/插入的理论和码设计。针对实用存储系统中的粘性删除/插入码，Chee 等人 [1] 提出了用于磁道存储器中粘性插入的码构造方案；Jain等人 [2] 提出了用于纠正 DNA 数据存储系统中重复错误的码，这与粘性插入/删除问题密切相关。

Cassuto等人 [4] 研究了非对称有限幅度错误，并提出了一种用于纠正此类错误的码。这种错误可推广为错误球 $\mathcal{B}(n,t,k_+,k_-)$：对于长度为 n 的序列，至多有 t 个条目增加不超过 k_+，减少不超过 k_-。因此，用于纠正 t 非对称有限幅度错误的码等价于用错误球 $\mathcal{B}(n,t,k_+,k_-)$ 对 Σ^n 进行装填。已有一系列研究探讨了通过 $\mathcal{B}(n,t,k_+,k_-)$ 进行装填/铺砌，相关工作始于 [14] 年的 $t=1$，并推广至 [15], [16] 年的通用常数 $t\geq 2$。

在这项工作中，我们的目标是构造能够纠正t粘性删除且幅度受限为 ℓ的码，其中 t 和ℓ 均为常数。这意味着删除最多发生在 t个游程中，且每个游程中最多允许删除 ℓ 个重复比特。我们的主要结果如下：

- 我们提出了一种非系统码，用于纠正幅度受限为 ℓ的 t 粘性删除，其冗余为$2t(1-1/p)\cdot\log(n+1)+O(1)$，其中 p 是大于 $\ell+1$的最小素数。当 $t=1$时，该码的冗余还可以进一步降低。
- 我们提出了一种系统码，用于纠正幅度受限为 ℓ的 t 粘性删除，并具有高效的编码和解码算法，其冗余为$\lceil 2t_{(1-1/p)\cdot\lceil\log p\rceil\log p}\log(n+1)+O(\log\log n)$，其中 p 是大于 $\ell+1$的最小素数。

如果忽略 $\lceil\log p\rceil$ 和 $\log p$ 之间的差异，我们会发现，我们构造的系统码的冗余与非系统码相比，至多相差 $O(\log\log n)$。

本文组织如下。第二节给出符号和预备知识。第三节介绍一种用于纠正带有 ℓ幅度受限的 t粘性删除的非系统码，并给出该码大小的下界。第四节介绍高效的编码和解码算法，并分析冗余和时间复杂度。最后，第五节总结本文。

II. 符号与预备知识

现在介绍本文中使用的符号。令 Σ_q 表示一个大小为 q 的有限字母表， Σ_q^n 表示由 Σ_q构成的所有长度为 n 的序列的集合。不失一般性，我们假设 $\Sigma_q = \{0, 1, \ldots, q-1\}$。为简化记号，我们将集合 $\{1, 2, \ldots, m\}$ 记为 $[m]$。对于两个整数 $i < j$，令 $[i, j]$ 表示集合 $\{i, i+1, i+2, \ldots, j\}$。

我们用粗体字母表示序列，例如 $\boldsymbol{x}$；用普通字母表示序列中的元素，例如 $\boldsymbol{x} \in \Sigma_q^n$对应 $\boldsymbol{x} = x_1 \cdots x_n$。对于函数，如果其输出是一个序列，我们也用粗体字母表示，例如 $\boldsymbol{\phi}(\boldsymbol{x})$。 $\boldsymbol{\phi}(\boldsymbol{x})$中的第 i个元素记作 $\phi(\boldsymbol{x})_i$。 $\boldsymbol{x}_{[i,j]}$ 表示从索引i 开始到索引 j结束（含端点）的子串。游程是由相同符号组成的极大子串， $n_r(\boldsymbol{x})$表示序列 $\boldsymbol{x}$的游程数。序列$\boldsymbol{x}$ 的重量 $\mathsf{wt}(\boldsymbol{x})$表示其中非零符号的数量。

定义1. 定义函数 $\boldsymbol{\psi}$： $\Sigma_2^n \to \Sigma_2^n$，使得

$$\psi(\boldsymbol{x})_i = \begin{cases} x_i \oplus x_{i+1}, & i = 1, 2, \ldots, n-1 \\ x_n, & i = n \end{cases}$$

其中， $a \oplus b$ 表示 $(a+b) \bmod 2$。

对于二进制序列 $\boldsymbol{x} \in \Sigma_2^n$，我们可以将其唯一地写成 $\boldsymbol{x} = 0^{r_1}10^{r_2}10^{r_3}\ldots10^{r_{w+1}}$，其中 $w = \mathsf{wt}(\boldsymbol{x})$。为方便下文讨论，我们在 $\boldsymbol{x}$ 末尾追加一位 1，并将其记为 $\boldsymbol{x}1$。由于序列$\boldsymbol{x}1$ 总是以 1 结尾，因此 $\boldsymbol{x}1$ 总可以写成 $\boldsymbol{x}1 = 0^{r_1}10^{r_2}10^{r_3}\ldots0^{r_w}1$，其中 $w = \mathsf{wt}(\boldsymbol{x}1)$。

定义 2. 定义函数 $\boldsymbol{\gamma} : \Sigma_2^n \to \Sigma^w$ and $\boldsymbol{\gamma}(\boldsymbol{x}1) \overset{\text{定义}}{=} (r_1, r_2, r_3, \ldots, r_w) \in \Sigma^w$, 其中 $\boldsymbol{x}1 = 0^{r_1}10^{r_2}10^{r_3}\ldots0^{r_w}1$and $w = \mathsf{wt}(\boldsymbol{x}1)$.

定义 3. 定义函数 $\boldsymbol{\phi} : \Sigma_2^n \to \Sigma^{n_r}$ 使得$\boldsymbol{\phi}(\boldsymbol{x}1) = \boldsymbol{\gamma}(\boldsymbol{\psi}(\boldsymbol{x}1))$, 其中 $n_r = n_r(\boldsymbol{x}1)$.

由于映射函数 $\boldsymbol{\psi}$ 和 $\boldsymbol{\gamma}$ 都是一对一映射函数，因此映射函数 $\boldsymbol{\phi}$ 也是一一映射。

示例 1. 假设 $\boldsymbol{x} = 0111010100$. 那么， $\boldsymbol{x}1 = 01110101001$with $n_r(\boldsymbol{x}1) = 8$ 且 $\boldsymbol{\psi}(\boldsymbol{x}1) = 10011111011$. 此外， $\boldsymbol{\phi}(\boldsymbol{x}1) = \boldsymbol{\gamma}(\boldsymbol{\psi}(\boldsymbol{x}1)) = 02000010$ ，长度为 8。

定义 4. 给定一个序列 $\boldsymbol{y} \in \Sigma_q^n$，定义 ∂: $\Sigma_q^n \to \Sigma_2^n$使得

$$\partial(\boldsymbol{y})_i = \begin{cases} 0, & \text{when } y_i = 0; \\ 1, & \text{otherwise.} \end{cases}$$

此外，对于序列 $\boldsymbol{y} \in \Sigma_q^n$，记 $(\boldsymbol{y} \bmod a) = (y_1 \bmod a, y_2 \bmod a, \ldots, y_n \bmod a)$，其中 $a < q$。

定义 5. 粘性删除是指删除序列某个游程中的 k 个重复比特，但不能删除整个游程。

命题 1. 对于二进制序列 $\boldsymbol{x} \in \Sigma_2^n$，删除其某个游程中的 k个重复比特，等价于 $\boldsymbol{\phi}(\boldsymbol{x})$中的相应项减少 k，其中该游程为 $\boldsymbol{x}$。

证明。根据映射函数 $\boldsymbol{\phi}$ 的定义，每个符号 $\boldsymbol{\phi}(\boldsymbol{x})_i$ 的值等于序列中第 i 个游程的长度减 1。删除游程 $\boldsymbol{x}$ 中的 k 个重复比特，意味着该游程的长度减少 k。 □

因此， t 粘性删除模式 $\boldsymbol{d}$ 是一个序列 $(d_1, d_2, \ldots, d_{n_r})$，满足 $\mathsf{wt}(\partial(\boldsymbol{d})) = t$。假设传输序列 $\boldsymbol{x} \in \Sigma_2^n$受到 $\boldsymbol{d}$的干扰，则接收序列 $\boldsymbol{x}'$应为 $\boldsymbol{\phi}(\boldsymbol{x}'1) = \boldsymbol{\phi}(\boldsymbol{x}1) - \boldsymbol{d}$。

给定序列 $\boldsymbol{y} \in \Sigma_q^n$，非对称 ℓ幅度受限t 错误是指 $\boldsymbol{y}$ 中至多 t 个项最多减少/增加 ℓ，受干扰后的序列可写为 $\boldsymbol{y}' = \boldsymbol{y} - \boldsymbol{e}$，其中 $\mathsf{wt}(\partial(\boldsymbol{e})) = t$ 且 $e_i \le \ell, \forall [n]$。因此，本文将给出幅度受限为 ℓ的 t 粘性删除的定义。

定义 6. t 幅度受限为 ℓ 的粘性删除表示：给定序列 $\boldsymbol{x} \in \Sigma_2^n$， $\boldsymbol{\phi}(\boldsymbol{x}1) = (u_1, \ldots, u_{n_r})$ 中至多有 t 个条目减少不超过 ℓ。受损序列为 $\boldsymbol{x}'$，且 $\boldsymbol{\phi}(\boldsymbol{x}'1)$ 可写为 $\boldsymbol{\phi}(\boldsymbol{x}'1) = (v_1, \ldots, v_{n_r})$，其中 $n_r = n_r(\boldsymbol{x}'1) = n_r(\boldsymbol{x}1)$，

1) 幅度受限删除： $u_i - v_i \le \ell$ 和 $v_i \le u_i, \forall i \in [n_r]$;
2) 粘性删除：满足 $v_i \ne u_i, \forall i \in [n_r]$ 的索引 i 数量至多为 t。

示例 2. 假设我们有 $\boldsymbol{x} = 0100111001 \in \Sigma_2^{10}$，因此 $\boldsymbol{\phi}(\boldsymbol{x}1) = 001211$。如果检索序列 $\boldsymbol{x}' = 010101 \in \Sigma_2^5$ 且对应的 $\boldsymbol{\phi}(\boldsymbol{x}'1) = 000001$，通过比较 $\boldsymbol{\phi}(\boldsymbol{x}1)$ 和 $\boldsymbol{\phi}(\boldsymbol{x}'1)$，我们可以看出删除的幅度为 $\ell \le 2$，粘性删除的数量为 $t = 3$。

记 Φ 为由函数 $\boldsymbol{\phi}$ 映射 Σ_2^n 得到的集合，并记 Σ_2^n 为包含所有长度为 n 的二进制序列的集合。接下来，我们将展示 Φ 的基数。证明见附录 A。

引理 1。 Φ 的基数为：

$$|\Phi| = \sum_{n_r=1}^{n+1} \binom{n}{n_r - 1} = 2^n.$$

为简便起见，令 $D_{t,\ell}(\boldsymbol{x}) \subseteq \Sigma_2^{n-t\ell}$ 表示在 $\boldsymbol{x}$ 上发生 t 次、幅度受 ℓ 限的粘性删除时所有可能序列的集合。码 $\mathcal{C} \subseteq \Sigma_2^n$ 的大小记为$|\mathcal{C}|$，其冗余定义为 $\log(2^n/|\mathcal{C}|)$，其中本文中所有未注明底数的对数均以 2 为底。若对于两个不同的 $\boldsymbol{x}$、 $\boldsymbol{y} \in \mathcal{C}_{t,\ell}$、 $D_{t,\ell}(\boldsymbol{x}) \cap D_{t,\ell}(\boldsymbol{y}) = \emptyset$，则称码 $\mathcal{C}_{t,\ell} \subseteq \Sigma_2^n$ 为 t 次、幅度受 ℓ 限的粘性删除纠错码。

引理 2.（参见 *[6]*）能够纠正 r 粘性插入/删除[1] 的码 $\mathcal{C}_{sr}$ 的最大大小$\mathcal{M}_{sr}$ 的上界为：

$$\mathcal{M}_{sr} = \begin{cases} 2^{n+r} \cdot r!/n^r, & \text{when } r \text{ is odd} \\ 2^{n+3r/2} \cdot ((r/2)!)^2/n^r. & \text{when } r \text{ is even.} \end{cases}$$

[1] r 表示粘性删除的最大数量，而在我们的定义中，t 表示粘性删除最多发生在 t 个游程中。

推论 1. 根据引理 2，能够纠正 r 粘性插入/删除的码的冗余下界为 $r\log n + O(1)$，其中 r为常数。

根据我们对 t 粘性删除且幅度受 ℓ限制的定义可知，删除总数最多为 $t\ell$。因此，引理 2 中的码 $\mathcal{C}_{sr}$ 只需令 $r = t\ell$，便可轻松纠正 t 次粘性删除，且幅度受 ℓ限制。当 $r = t\ell$时，根据推论 1，在不限制幅度的情况下，码 $\mathcal{C}_{sr}$的冗余下界为 $t\ell\log n + O(1)$。本文通过引入删除幅度限制，在 $\ell > 2$时可进一步将码的冗余从至少$t\ell\log n + O(1)$降低至至多 $2t(1-1/p)\log(n+1) + O(1)$，其中 p 是大于 $\ell+1$的最小素数。

命题2. 在$\boldsymbol{x}$ 中， t 次粘性删除且幅度受 ℓ限制，等价于 $\boldsymbol{\phi}(\boldsymbol{x}1)$中的 t 个非对称 ℓ有限幅度错误。

证明。通过扩展前述命题 1，可以很容易地证明这一点。□

根据命题 2，我们可以看出，在$\boldsymbol{x} \in \Sigma_2^n$ 中纠正 t 粘性删除（幅度为 ℓ受限）的构造，等价于在 $\boldsymbol{\phi}(\boldsymbol{x}1) \in \Sigma^{n_r}$中纠正$t$ 非对称 ℓ幅度受限错误的码构造，其中 $n_r = n_r(\boldsymbol{x}1)$，且 $\mathrm{wt}(\boldsymbol{\phi}(\boldsymbol{x}1)) = n+1-n_r$。

III. 非系统码构造

本节将给出一种非系统码构造，用于纠正 t 粘性删除（幅度为ℓ受限）。随后，我们将介绍该码的解码算法以及码大小的下界。

构造 1。码 $\mathcal{C}_{t,\ell}$ 定义为

$$\mathcal{C}_{t,\ell} = \{\boldsymbol{x} \in \Sigma_2^n : \boldsymbol{\phi}(\boldsymbol{x}1) \bmod q \in \mathcal{C}_q, \mathrm{wt}(\boldsymbol{\phi}(\boldsymbol{x}1)) = n+1-n_r\},$$

其中 $n_r = n_r(\boldsymbol{x}1)$，且 $\mathcal{C}_q$是一个定义在 Σ_q上的码，满足 $q = \ell+1$。

引理 3. $\mathcal{C}_{t,\ell}$ 能够纠正 t 个幅度受限为ℓ的粘性删除，当 $\boldsymbol{x} \in \mathcal{C}_{t,\ell}$ 且 $\mathcal{C}_q$ 能够纠正 $\boldsymbol{\phi}(\boldsymbol{x}1)$中的$t$ 个对称错误时。

证明。由命题 1 和 2 可知，对 $\boldsymbol{x}$ 纠正 t 个幅度受限为 ℓ的粘性删除，等价于在 $\boldsymbol{\phi}(\boldsymbol{x}1)$中纠正 t 个幅度受限为 ℓ的非对称错误。此外，[4] 中的定理 4 已证明，若 $\mathcal{C}_q$能够纠正 t 个对称错误，则码 $\mathcal{C}$ 能够纠正 t 个幅度受限为 ℓ的非对称错误，其中 $\mathcal{C} = \{\boldsymbol{x} \in \Sigma_2^n : \boldsymbol{x}$ 模 $q \in \mathcal{C}_q\}$，且 $q = \ell+1$。□

引理 4.（*[17],* 定理 10）设 p 是一个素数，使得距离 $2 \le d \le p^{\lceil m/2\rceil - 1}$ 和 $n = p^m - 1$。那么，存在一个窄义 $[n,k,d]$-BCH 码 $\mathcal{C}_p$，定义在 Σ_p 上，且

$$n - k = \lceil (d-1)(1-1/p)\rceil m.$$

定理 1. 设 p 是满足 $p \ge \ell+1$和 $n_r = n_r(\boldsymbol{x}1)$的最小素数。那么，码 $\mathcal{C}_{t,\ell}$ 满足

$$\mathcal{C}_{t,\ell} = \{\boldsymbol{x} \in \Sigma_2^n : \boldsymbol{\phi}(\boldsymbol{x}1) \bmod p \in \mathcal{C}_p, \mathrm{wt}(\boldsymbol{\phi}(\boldsymbol{x}1)) = n+1-n_r\}.$$

能够纠正 t 个幅度受限为 ℓ的粘性删除。

证明。令$\boldsymbol{x} \in \mathcal{C}_{t,\ell}$ 为一个码字，且$\boldsymbol{x}' \in D_{t,\ell}(\boldsymbol{x})$为经过含有 t 个幅度受限为ℓ的粘性删除的信道后得到的输出。令 $\boldsymbol{z}' = \boldsymbol{\phi}(\boldsymbol{x}'1)$模 p，其中 p 是满足 $p \ge \ell+1$的素数。在 $\boldsymbol{z}'$ 上应用 $\mathcal{C}_p$ 的解码算法，并输出 $\boldsymbol{z}^*$。因此， $\boldsymbol{z}^*$ 也是 $\mathcal{C}_p$ 中的一个线性码，并且可以证明 $\boldsymbol{z}^* = \boldsymbol{\phi}(\boldsymbol{x}1)$模 p。记 $\boldsymbol{\epsilon} = (\boldsymbol{z}^* - \boldsymbol{z}')$模 p，可得

$$(\boldsymbol{\phi}(\boldsymbol{x}1) - \boldsymbol{\phi}(\boldsymbol{x}'1)) \bmod p = (\boldsymbol{z}^* - \boldsymbol{z}') \bmod p = \boldsymbol{\epsilon}.$$

因此，输出为 $\boldsymbol{\phi}(\boldsymbol{x}1) = \boldsymbol{\phi}(\boldsymbol{x}'1) + \boldsymbol{\epsilon}$，然后从 $\boldsymbol{\phi}(\boldsymbol{x}1)$ 中恢复 $\boldsymbol{x}$。□

详细的解码步骤如算法 1 所示。

算法 1： $\mathcal{C}_{t,\ell}$ 的解码算法

输入：检索序列 $\boldsymbol{x}' \in D_{t,\ell}(\boldsymbol{x})$
输出：解码序列 $\boldsymbol{x} \in \mathcal{C}_{t,\ell}$。
初始化：令 p 为大于 $\ell+1$。此外，在 $\boldsymbol{x}'$末尾追加 1，得到 $\boldsymbol{\phi}(\boldsymbol{x}'1)$。
步骤 1：对 $\boldsymbol{z}' = \boldsymbol{\phi}(\boldsymbol{x}'1)$ 取模 p。运行解码算法，将 $\mathcal{C}_p$ 应用于 $\boldsymbol{z}'$，得到输出 $\boldsymbol{z}^*$。
步骤 2： $\boldsymbol{\epsilon} = (\boldsymbol{z}^* - \boldsymbol{z}') \bmod p$，以及 $\boldsymbol{\phi}(\boldsymbol{x}1) = \boldsymbol{\phi}(\boldsymbol{x}'1) + \boldsymbol{\epsilon}$。
步骤 3：输出 $\boldsymbol{x}1 = \boldsymbol{\phi}^{-1}(\boldsymbol{\phi}(\boldsymbol{x}1))$，然后 $\boldsymbol{x}$。

例 3。假设 $\boldsymbol{x} = 0100111001$ 和 $\boldsymbol{x}' = D_{3,2}(\boldsymbol{x})$ =010101，其中 $\ell \le 2$，$t = 3$。由于检索序列$\boldsymbol{x}' = 010101$，于是 $\boldsymbol{\phi}(\boldsymbol{x}'1) = 000001$ 和 $\boldsymbol{z}' = \boldsymbol{\phi}(\boldsymbol{x}')$ mod3 $= 000001$，其中 p 是满足 $p \ge l+1 = 3$的最小素数。

对 $\boldsymbol{z}' \in \mathcal{C}_p$运行 $\mathcal{C}_p$ 的解码算法，我们得到输出序列 $\boldsymbol{z}^* = 001211$，因此 $\boldsymbol{\epsilon} = (\boldsymbol{z}^* - \boldsymbol{z})$ mod3 $= 001210$。由此，解码算法$\boldsymbol{\phi}(\boldsymbol{x}1)$ 的输出为 $= \boldsymbol{\phi}(\boldsymbol{x}'1) + \boldsymbol{\epsilon} = 000001 + 001210 = 001211$ 和$\boldsymbol{x} = 0100111001$。

接下来，我们将给出 $\mathcal{C}_{t,\ell}$ 的大小下界。证明见附录 B。

定理 2. 定理 1 中码 $\mathcal{C}_{t,\ell}$ 的大小满足以下界，其中 p 是大于 $\ell+1$ 的最小素数。

$$|\mathcal{C}_{t,\ell}| \ge \frac{2^n}{p(n+1)^{2t(1-1/p)}} \ge \frac{2^n}{(2\ell+2)(n+1)^{t(2\ell+1)/(\ell+1)}}.$$

推论 2. 存在一种码$\mathcal{C}_{t,\ell}$ 能够纠正 t-粘性删除，且幅度受限为 ℓ，冗余不超过 $t(2\ell+1)/(\ell+1)\log(n+1) + O(1)$ 比特。

由上述推论 2 可知，当 $t=1,\ell=1$时，这种码的冗余为$3\,2\log(n+1)+O(1)$。因此，我们考虑特殊情形 $t=1$，并尝试构造一些冗余更低的码。

在 $t=1$ and $\ell=1$ 的情况下，只有一次粘性删除，我们可以如下构造一个码。记 $\mathcal{C}_H$ 为能够纠正单个错误的 $(2^m-1,2^m-m-1,3)$-汉明码。我们在构造 1 中取 $\mathcal{C}_H$ 为 $\mathcal{C}_q$，并令 $n=2^m-1$。因此，码 $\mathcal{C}_{1,\ 1}$ 的大小上界为 $|\mathcal{C}_{1,\ 1}|\geq\frac{2^{n-1}}{n+1}$。

此外，在 $t=1$ and ℓ 为给定整数的情况下，我们现在构造一个能够纠正幅度至多为 ℓ 的粘性删除的码。令 $q=\ell+1$ and p 为满足 $p>n_r.$ 的最小素数。对于任意整数 $a\in[0,p-1]$ 和 $b\in[0,\ell]$，令码 $\mathcal{C}_{q,\ a,\ b}$ 为

$$\mathcal{C}_{q,a,b}=\{\boldsymbol{u}=(u_1,\ldots,u_{n_r})\in\Sigma_q^{n_r}:\sum_{i=1}^{n_r}iu_i\equiv a \bmod p \\ \sum_{i=1}^{n_r}u_i\equiv b \bmod (\ell+1)\}.$$

我们证明 $\mathcal{C}_{q,\ a,\ b}$ 是一个能够纠正单个 ℓ-有限幅度错误的码。证明见附录 C。

引理 5. 上述构造的码 $\mathcal{C}_{q,a,b}$ 可以纠正一次 ℓ 限幅错误。

对于任意可区分的数对 $(a_1,b_1)\neq(a_2,b_2)$，两个码 $\mathcal{C}_{q,a_1,b_1}$ 和 $\mathcal{C}_{q,a_2,b_2}$ 也可区分，即

$$\mathcal{C}_{q,a_1,b_1}\cap\mathcal{C}_{q,a_2,b_2}=\emptyset,\ \cup_{a=0}^{p-1}\cup_{b=0}^{\ell}\mathcal{C}_{q,a,b}=\Sigma_q^{n_r}. \quad (1)$$

对于每个码 $\mathcal{C}_{q,a,b}$，我们定义以下码

$$\mathcal{C}_{t,\ell,a,b}=\{\boldsymbol{x}\in\Sigma_2^n:\boldsymbol{\phi}(\boldsymbol{x}1)\bmod q\in\mathcal{C}_{q,a,b}\}$$

由 (1) 可得

$$\mathcal{C}_{t,\ell,a_1,b_1}\cap\mathcal{C}_{t,\ell,a_2,b_2}=\emptyset, \quad (2)$$

对于任意数对 $(a_1,b_1)\neq(a_2,b_2)$，且

$$\cup_{a=0}^{p-1}\cup_{b=0}^{\ell}\mathcal{C}_{t,l,a,b}=\Sigma_2^n. \quad (3)$$

由 (2) 和 (3)，存在整数 a,b，使得码 $\mathcal{C}_{t,\ell,a,b}$ 的大小至多为 $\frac{2^n}{p\ \ell+1()}$.

与引理 3 中的论证类似，我们可以证明，码 $\mathcal{C}_{t,\ \ell,\ a,\ b}$ 能够纠正幅度至多为 $\ell.$ 的一次粘性删除。因此，我们得到如下结果。

定理 3。存在一种码，能够纠正幅度至多为 ℓ 的一次粘性删除，且冗余至多为 $\log n+\log(2\ell+2)$ 比特。

IV. 系统码构造

在上一节中，我们提出了一种非系统码$\mathcal{C}_{t,\ \ell}$用于纠正幅度受限为 ℓ 的 t粘性删除。本节将基于定理1中提出的码 $\mathcal{C}_{t,\ \ell}$给出高效的编码和解码函数。

A. 高效编码

在介绍高效的系统编码算法之前，我们先引入一个在 [18] 中提出的有用引理，该引理用于高效地编码平衡序列。平衡序列指 0 和 1 数量相等的二进制序列，可用于区分冗余的边界。

引理6。（参见 *[18]*）给定输入 $\boldsymbol{x}\in\Sigma_2^k$，令函数 $\boldsymbol{s}':\Sigma_2^k\to\Sigma_2^n$，使得 $\boldsymbol{s}'(\boldsymbol{x})\in\Sigma_2^n$ 为一个平衡序列，其中 $n=k+\log k$。

定义 7。给定输入 $\boldsymbol{x}\in\Sigma_2^k$，定义函数 $\boldsymbol{s}:\Sigma_2^k\to\Sigma_2^{n'}$，使得 $\boldsymbol{s}(\boldsymbol{x})\in\Sigma_2^{n'}$ 的首比特为 1，且 $\boldsymbol{s}(\boldsymbol{x})_{[2,n']}$ 是一个平衡序列，其中包含 $(n'-1)/2$ 0 个和 $(n'-1)/2$ 1 个，且 $n'=k+\log k+1$。

此外，以下引理用于纠正二进制序列中的 r 0-删除。

引理 7。（参见 *[7]*）对于任意 $\boldsymbol{x}\in\Sigma_2^k$，存在标记函数 $f_0:\Sigma_2^k\to\Sigma_2^{n-k}$，使得码（$\boldsymbol{x}$，$f_0(\boldsymbol{x})$）能够纠正 r 0-删除，其中当 r 为常数时，$n-k=r\log n$。

接下来，我们定义从非二进制到二进制的映射函数。

定义 8。给定输入 $\boldsymbol{x}\in\Sigma_2^k$，定义函数$\boldsymbol{b}:\Sigma_p^k\to\Sigma_2^n$，使得 $\boldsymbol{b}(\boldsymbol{u})_{[i\cdot\lceil\log p\rceil+1,(i+1)\cdot\lceil\log p\rceil]}$ 是 u_i 的二进制形式，其中 $n=k\cdot\lceil\log p\rceil$。

给定 t-粘性删除及ℓ-幅度受限信道中的参数 t 和 ℓ，令 p 为大于 $\ell+1$ 的最小素数，并令引理 4 中的 $\mathcal{C}_p$ 为 p 元本原窄义 $[n,k,2t+1]$-BCH 码。

定义 9. 将标记函数定义为 $g:\Sigma_p^k\to\Sigma_p^{n-k}$，使得 $(\boldsymbol{x},g(\boldsymbol{x}))$ 是一个 p 元本原窄义 BCH$[n,k,2t+1]$ 码，其中 $n=k+\lceil 2t(1-1/p)\rceil m$ 且 $n=p^m-1$。

接下来，我们将介绍具体的编码过程。假设输入序列为 $\boldsymbol{c}\in\Sigma_2^k$，且我们有长度为 $r_c=n_r(\boldsymbol{c}1)$ 的 $\boldsymbol{\phi}(\boldsymbol{c}1)$。然后，令 $\boldsymbol{c}'=\boldsymbol{\phi}(\boldsymbol{c}1)$ 对 $p\in\Sigma_p^{r_c}$ 取模，并将 $\boldsymbol{0}^{k+1-r_c}$ 附加到 $\boldsymbol{c}'$ 的末尾。因此，我们记为 $\bar{\boldsymbol{c}}\in\Sigma_p^{k+1}=(\boldsymbol{c}',\boldsymbol{0}^{k+1-r_c})$。

接下来，通过标记函数 g 对 $\bar{\boldsymbol{c}}$ 进行编码，并输出冗余部分 $g(\bar{\boldsymbol{c}})$。然后，我们将冗余部分 $g(\bar{\boldsymbol{c}})$ 映射为二进制序列 $\boldsymbol{b}(g(\bar{\boldsymbol{c}}))$，并将 $\boldsymbol{b}(g(\bar{\boldsymbol{c}}))$ 转换为平衡序列 $h_1(\bar{\boldsymbol{c}})=\boldsymbol{s}(\boldsymbol{b}(g(\bar{\boldsymbol{c}})))$。

此外，我们需要保护冗余部分 $h_1(\bar{\boldsymbol{c}})$。思路是在 $h_1(\bar{\boldsymbol{c}})$ 上应用引理 7 中的码，并输出 $f_0(h_1(\bar{\boldsymbol{c}}))$。另外，将 $f_0(h_1(\bar{\boldsymbol{c}}))$ 转换为平衡序列$\boldsymbol{s}(f_0(h_1(\bar{\boldsymbol{c}})))$，并将其中的每一位重复 $2t\ell+1$ 次。令$h_2(\bar{\boldsymbol{c}})=\mathrm{Rep}_{2t\ell+1}\boldsymbol{s}((f_0(h_1(\bar{\boldsymbol{c}}))))$，其中 $\mathrm{Rep}_k(\boldsymbol{x})$ 是 $\boldsymbol{x}$ 的 k重重复。

最后，我们得到输出 $\mathrm{Enc}(\boldsymbol{c})=(\boldsymbol{c},\boldsymbol{\psi}^{-1}(h(\boldsymbol{c})))$，其中 $h(\boldsymbol{c})=(h_1(\bar{\boldsymbol{c}})\ h_2(\bar{\boldsymbol{c}}))$。详细的编码步骤总结如下算法 2。

算法 2：编码算法

输入： $\boldsymbol{c} \in \Sigma_2^k$
输出：编码序列 $\mathrm{Enc}(\boldsymbol{c}) \in \Sigma_2^N$
初始化：令 p 为大于以下数值的最小素数 $\ell+1$。
步骤 1：在 $\boldsymbol{c}$ 末尾追加 1，得到 $\boldsymbol{\phi}(\boldsymbol{c}1)$，其长度 $r_c = n_r(\boldsymbol{c}1)$。
步骤 2： $\boldsymbol{c}' = \boldsymbol{\phi}(\boldsymbol{c}1)$ 模 $p \in \Sigma_p^{r_c}$。在 $\mathbf{0}^{k+1-r_c}$ 处追加 $\mathbf{0}^{k+1-r_c}$ at 在 $\boldsymbol{c}'$末尾，然后 $\bar{\boldsymbol{c}} = (\boldsymbol{c}', \mathbf{0}^{k+1-r_c})$。
步骤 3：通过 $\mathcal{C}$对 $\bar{\boldsymbol{c}}$ 进行编码$_p$ 并输出 $g(\bar{\boldsymbol{c}})$。映射 $g(\bar{\boldsymbol{c}})$ 转换为平衡二进制序列 $h_1(\bar{\boldsymbol{c}}) = \boldsymbol{s}(\boldsymbol{b}(g(\bar{\boldsymbol{c}})))$.
步骤 4：通过 f_0 保护 $h_1(\bar{\boldsymbol{c}})$，并得到总冗余 $h(\boldsymbol{c}) = (h_1(\bar{\boldsymbol{c}})\ h_2(\bar{\boldsymbol{c}}))$.
步骤 5：输出 $\mathrm{Enc}(\boldsymbol{c}) = (\boldsymbol{c}, \boldsymbol{\psi}^{-1}(h(\boldsymbol{c}))) \in \Sigma_2^N$.

引理 8. 给定一个序列 $\boldsymbol{c} \in \Sigma_2^k$，算法 2 输出一个编码序列，能够纠正具有 ℓ-幅度受限 $\mathrm{Enc}(\boldsymbol{c}) \in \Sigma_2^N$ 的 t 粘性删除。

因此，通过此编码过程，码 $h(\boldsymbol{c}) = (h_1(\bar{\boldsymbol{c}})\ h_2(\bar{\boldsymbol{c}}))$ 的总冗余可表示如下。证明见附录 D。

定理 4。码 $\mathrm{Enc}(\boldsymbol{c}) \in \Sigma_2^N$在给定输入 $\boldsymbol{c} \in \Sigma_2^k$ 时的总冗余为

$$N-k = \frac{\lceil 2t(1-1/p)\rceil \cdot \lceil \log p\rceil}{\log p}\log(N+1) + O(\log\log N).$$

其中， p 是满足 $p \geq \ell+1$的最小素数。

B. 解码算法

不失一般性地，假设编码序列 $\mathrm{Enc}(\boldsymbol{c}) \in \Sigma_2^N$ 通过具有 ℓ-幅度受限信道的 t 次粘性删除进行传输，并且我们得到检索序列 $\boldsymbol{d} \in \Sigma_2^{N-t\ell}$。在本小节中，我们将介绍根据给定的 $\boldsymbol{d} \in \Sigma_2^{N-t\ell}$获取 $\mathrm{Dec}(\boldsymbol{d}) \in \Sigma_2^k$ 的解码算法。具体的解码过程如下。

首先，我们得到 $\boldsymbol{\psi}(\boldsymbol{d})$，它是 $\boldsymbol{d}$的导数，但我们需要确定冗余部分从何处开始。由于在 $\mathrm{Enc}(\boldsymbol{c})$ 中发生 t次具有 ℓ-幅度限制的粘性删除，等价于在 $\boldsymbol{\psi}(\mathrm{Enc}(\boldsymbol{c}))$ 中至多删除 $t\ell$ 0 个 $\boldsymbol{\psi}$，因此 $\boldsymbol{\psi}(\boldsymbol{d})$ 中 1 的数量与 $\boldsymbol{\psi}(\mathrm{Enc}(\boldsymbol{c}))$ 中 1 的数量相同。于是，我们可以从 $\boldsymbol{\psi}(\boldsymbol{d})$ 的末尾开始统计 1 的数量，以找到冗余的起始位置，因为冗余部分是平衡序列。

因此，我们从 $\boldsymbol{\psi}(\boldsymbol{d})$ 的末尾找到第 $(n_2+2t\ell+1)/2$ 个 1 和第 $(n_1/2+n_2/2+t\ell+1)$ 个 1，并分别将它们的条目记为 i_{r2} 和 i_{r1}。对于子序列$\boldsymbol{\psi}(\boldsymbol{d})_{[i_{r2}, N-t\ell]}$，由于 $\mathrm{Enc}(\boldsymbol{c})_{[N-n_2+1, N]}$ 中至多有 $t\ell$ 0s 被删除，$(2t\ell+1)$ 重复码可以帮助恢复 $\boldsymbol{s}(f_0(h_1(\bar{\boldsymbol{c}})))$。此外，我们可以得到校验位$f_0(h_1(\bar{\boldsymbol{c}}))$。

接下来，对于子序列 $\boldsymbol{\psi}(\boldsymbol{d})_{[i_{r1}, i_{r2}-1]}$，$\mathrm{Enc}(\boldsymbol{c})_{[N-n_1-n_2+1, N-n_2]}$ 中也至多有 $t\ell$ 0s 被删除。因此，引理 7 中介绍的码可以保护 $\mathrm{Enc}(\boldsymbol{c})_{[N-n_1-n_2+1, N-n_2]}$ 的冗余部分。然后，借助校验位 $f_0(h_1(\bar{\boldsymbol{c}}))$，可以恢复 $h_1(\bar{\boldsymbol{c}})$。此外，我们可以从 $h_1(\bar{\boldsymbol{c}}) = \boldsymbol{s}(\boldsymbol{b}(g(\bar{\boldsymbol{c}})))$ 中得到 $g(\bar{\boldsymbol{c}})$。

最后，记 $\boldsymbol{z} = (\boldsymbol{\phi}(\boldsymbol{d}_{[1, i_{r1}-1]}, 1)\ \mathbf{0}^{k+1-r_c}) \in \Sigma^{k+1}$ and $\boldsymbol{z}' = \boldsymbol{z} \bmod p$，其中 r_c 是 $\boldsymbol{\phi}(\boldsymbol{d}_{[1, i_{r1}-1]}, 1)$ and $k = N - n_1 - n_2$的长度。接下来的解码步骤与算法 1 相同，其中 $\boldsymbol{z}'$ 是算法 1 步骤 1 的输入。唯一的区别是，在执行 $\boldsymbol{\phi}^{-1}$的最后一步之前，我们需要先移除末尾的 $\mathbf{0}^{k+1-r_c}$。因此，解码 $\boldsymbol{d} \in \Sigma_2^{N-t\ell}$的主要步骤总结于算法3中。

算法3：解码算法 输入： $\boldsymbol{d} \in \Sigma_2^{N-t\ell}$输出：解码序列 $\mathrm{Dec}(\boldsymbol{d}) \in \Sigma_2^k$初始化：令 p 为大于$\ell+1$的最小素数。
步骤 1：获取 $\boldsymbol{\psi}(\boldsymbol{d})$。从 $\boldsymbol{\psi}(\boldsymbol{d})$的末尾开始，找到第 $(n_2+2t\ell+1)/2$ 个 1 和第 $(n_1/2+n_2/2+t\ell+1)$ 个 1，并分别将其值记为 i_{r2} 和 i_{r1}。步骤 2：根据 $\boldsymbol{\psi}(\boldsymbol{d})_{[i_{r2}, N-t\ell]}$ 恢复 $\boldsymbol{s}(f_0(h_1(\bar{\boldsymbol{c}})))$，然后得到 $f_0(h_1(\bar{\boldsymbol{c}}))$。
步骤 3：通过 $f_0(h_1(\bar{\boldsymbol{c}}))$恢复 $h_1(\bar{\boldsymbol{c}})$，然后得到$h_1(\bar{\boldsymbol{c}})$。
步骤 4：记$\boldsymbol{z}' = (\boldsymbol{\phi}(\boldsymbol{d}_{[1, i_{r1}-1]}, 1)\ \mathbf{0}^{k+1-r_c}) \bmod p$。将 $\boldsymbol{z}'$ 作为算法 1 步骤 1 的输入，并执行算法 1 的其余步骤。步骤 5：输出 $\mathrm{Dec}(\boldsymbol{d})$。

C. 时间复杂度

对于编码算法，给定常数 t, ℓ，每个码字通过以下步骤生成：

- 首先，给定输入二进制消息串 $\boldsymbol{c}$ 并输出$\bar{\boldsymbol{c}}$。时间复杂度为 $O(n)$。
- 其次，通过 $\mathcal{C}_p$对 $\bar{\boldsymbol{c}}$ 进行编码。 p元窄义 $[n, k, 2t+1]$-BCH 码 $\mathcal{C}_p$的时间复杂度为$O(tn\log\ n)$。
- 第三，将 $\bar{\boldsymbol{c}}$ 的标记函数映射为平衡二进制序列，时间复杂度为 $O(\log n)$。
- 第四，使用引理 7 中的码保护冗余部分 $h_1(\bar{\boldsymbol{c}})$，并将其转换为平衡序列和重复码，时间复杂度为 $O(\log n)$。

因此，编码器的时间复杂度由 p 元窄义 BCH 码主导，其时间复杂度为$O(tn\log\ n)$。

对于解码算法，我们可以轻易证明，时间复杂度主要取决于对 p 元窄义 BCH 码的解码以及对引理 7 中码的解码。与编码过程不同，引理 7 中码的解码采用蛮力法，因此时间复杂度为

$O((n_2'')^{t\ell}) = O((\log\ n)^{t\ell})$。因此，解码的总时间复杂度为 $O(tn + (\log n)^{t\ell})$。

V. 结论

本文提出了用于纠正幅度受限为 ℓ 的 t 粘性删除的码。我们首先针对这类错误提出了一种非系统码，并分析了其大小。随后，我们构造了系统码，并提出了高效的编码和解码算法。然而，仍有一些有趣的问题有待研究，包括将这项工作扩展到更多次删除的情形，而不仅限于常数 t。

参考文献

[1] Y. M. Chee、H. M. Kiah、A. Vardy、V. K. Vu 和 E. Yaakobi，《赛道存储器的编码》，IEEE信息论汇刊，第 64 卷，第 11 期，第 7094–7112 页，2018 年。[2] S. Jain、F. F. Hassanzadeh、M. Schwartz 和 J. Bruck，《用于在生物体 DNA 中存储数据的重复纠错码》，IEEE信息论汇刊，第 63 卷，5010，2017 8，4996，无页码。[3] M. Kova˘cevi´c 和 V. Y. F. Tan，《DNA存储系统中纠正定长重复错误的渐近最优码》，IEEE 通信快报，第 22 卷，第 11 期，第 2194–2197 页，2018 年。[4] Y. Cassuto、M. Schwartz、V. Bohossian 和 J. Bruck，《应用于多级闪存的非对称有限幅度错误码》，IEEE信息论汇刊，第 56 卷，第 4 期，第 1582–1595 页，2010 年。[5] H. H. Lee、R. Kalhor、N. Goela、J. Bolot 和 G. M. Church，《用于数字信息存储的无终止子、模板非依赖性酶促 DNA 合成》，《自然通讯》，第 10 卷，第 1 期，第 1–12 页，2019 年。[6] V. Levenshtein，《能够纠正伪插入和 1 删除的二进制码》，《信息传输问题》，第 1 卷，第 1 期，第 8–17 页，1965 年。[7] L. Dolecek 和 V. Anantharam，《重复错误校正集：显式构造与前缀方法》，《离散数学 SIAM 杂志》，第 23 卷，第 4 期，第 2120–2146 页，2010 年。[8] H. Mahdavifar 和 A. Vardy，《具有高效编码与解码的渐近最优粘性插入纠错码》，载于 2017 年 IEEE 国际信息论研讨会（ISIT）。IEEE，2017 年，第 2683–2687 页。[9] L. G. Tallini、N. Elarief 和 B. Bose，《高效重复错误纠错码》，载于 2010 年 IEEE 国际信息论研讨会。IEEE，2010 年，第 1012–1016 页。[10] L. G. Tallini 和 B. Bose，《一类新的错误控制码与对称函数》，载于 2008 年 IEEE 国际信息论研讨会。IEEE，2008 年，第 980–984 页。[11] ——，《关于 L1距离错误控制码》，载于 2011 年 IEEE 国际信息论研讨会论文集。IEEE，2011 年，第 1061–1065 页。[12] ——，《关于 L1度量非对称/单向错误控制码、受限重量码和 σ码》，载于 2013 年 IEEE 国际信息论研讨会。IEEE，2013 年，第 694–698 页。[13] L. G. Tallini、N. Alqwaifly 和 B. Bose，《L1度量中符号“0”的删除与插入，以及非对称/单向错误控制码》，IEEE信息论汇刊，第 69 卷，第 1 期，第 86–106 页，2022 年。[14] M. Schwartz，《拟十字格点铺砌及其在闪存中的应用》，IEEE信息论汇刊，第 58 卷，第 4 期，第 2397–2405 页，2011 年。[15] H. Wei、X. Wang 和 M. Schwartz，《非对称限幅球的格点堆积与覆盖》，IEEE信息论汇刊，第 67 卷，第 8 期，第 5104–5115 页，2021 年。[16] H. Wei 和 M. Schwartz，《非对称限幅球的铺砌》，《欧洲组合学杂志》，第 100 卷，文章编号 103450，2022 年。[17] S. A. Aly、A. Klappenecker 和 P. K. Sarvepalli，《关于量子和经典 BCH 码》，IEEE信息论汇刊，第 53 卷，第 3 期，第 1183–1188 页，2007 年。[18] D. Knuth，《高效平衡码》，IEEE信息论汇刊，第 32 卷，第 1 期，第 51–53 页，1986 年。

附录 A 引理 1 的证明

引理 1. Φ 的基数为:

$$|\Phi| = \sum_{n_r=1}^{n+1} \binom{n}{n_r - 1} = 2^n.$$

证明。对于二进制序列 $\boldsymbol{x} \in \Sigma_2^n$，相应的序列 $\boldsymbol{\phi}(\boldsymbol{x}1)$ 的长度为 $n_r = n_r(\boldsymbol{x}1)$，且 $\mathrm{wt}(\boldsymbol{\phi}(\boldsymbol{x}1)) = n + 1 - n_r$。此外，$\Phi$ 的基数可视为将 $n + 1 - n_r$ 个不可区分对象放入 n_r 个可区分盒子中的排列方式数。因此，我们可以得到 Φ 的基数，如引理 1 所示。

另一方面，由于映射函数 $\boldsymbol{\phi}$ 是一一映射，Φ 的基数应与 $|\Sigma_2^n| = 2^n$ 相同。 □

附录 B 定理 2 的证明

定理 2. 定理 1 中码 $\mathcal{C}_{t,\ \ell}$ 的大小满足以下界，其中 p 是大于$\ell + 1$的最小素数。

$$\begin{aligned}|\mathcal{C}_{t,\ell}| &\geq \frac{2^n}{p(n+1)^{2t(1-1/p)}} \\ &\geq \frac{2^n}{(2\ell+2)(n+1)^{t(2\ell+1)/(\ell+1)}}.\end{aligned}$$

证明. 记 $\boldsymbol{z} = \boldsymbol{\phi}(\boldsymbol{x}1)$ 模 p。$\boldsymbol{\phi}(\boldsymbol{x}1)$可写为$\boldsymbol{\phi}(\boldsymbol{x}1) \to (\boldsymbol{z}, \boldsymbol{a})$，使得 $\boldsymbol{\phi}(\boldsymbol{x}1) = \boldsymbol{z} + p \cdot \boldsymbol{a}$，其中 $\boldsymbol{a}$ 是与 $\boldsymbol{\phi}(\boldsymbol{x}1)$长度相同的向量，且 $\boldsymbol{z}$。此外，由于$\boldsymbol{z} \in \mathcal{C}_p$ 且 $\mathcal{C}_p$ 是线性码，长度为 n_r 的码 $\mathcal{C}_p$ 可视为将$\Sigma_p^{n_r}$划分为 p^{n_r-k} 个类后得到的集合。

记 $\boldsymbol{\phi}(\boldsymbol{x}1)^{n_r}$ 为长度为 n_r的 $\boldsymbol{\phi}(\boldsymbol{x}1)$。因此，对于任意固定的游程数 n_r，满足 $\boldsymbol{\phi}(\boldsymbol{x}1)^{n_r}$ 模 $p \in \mathcal{C}_p$ 且长度为 n_r 的 $\boldsymbol{\phi}(\boldsymbol{x}1)^{n_r}$ 的基数为:

$$|\boldsymbol{\phi}(\boldsymbol{x}1)^{n_r}| = \frac{\binom{n}{n_r-1}}{p^{n_r-k}}.$$

那么，定理 1 中码的大小 $\mathcal{C}_{t,\ \ell}$ 可表示为:

$$\begin{aligned}|\mathcal{C}_{t,\ell}| = \sum_{n_r=1}^{n+1} |\boldsymbol{\phi}(\boldsymbol{x}1)^{n_r}| &= \sum_{n_r=1}^{n+1} \left[\frac{\binom{n}{n_r-1}}{p^{n_r-k}} \right] \\ &\geq \frac{\sum_{n_r=1}^{n+1} \binom{n}{n_r-1}}{p^{n+1-k}} = \frac{2^n}{p^{n+1-k}}. \quad (4)\end{aligned}$$

由引理 4 和定理 1，令 $d = 2t + 1$ 和 $m = \log_p(n + 1)$。

$$\begin{aligned}p^{n+1-k} &= p^{2t(1-1/p)\cdot\log_p(n+1)+1} = p(n+1)^{2t(1-1/p)} \\ &\leq (2\ell+2)(n+1)^{t(2\ell+1)/(\ell+1)}. \quad (5)\end{aligned}$$

其中，最后一个不等式是基于著名的伯特兰–切比雪夫定理，由 $\ell+1 \leq p < 2(\ell+1)$ 得出的。

因此，由式 (4) 和 (5) 可知，定理 1 中码 $\mathcal{C}_{t,\ \ell}$ 的大小满足以下上界：

$$|\mathcal{C}_{t,\ell}| \geq \frac{2^n}{(2\ell+2)(n+1)^{t(2\ell+1)/(\ell+1)}}.$$ □

APPENDIX C
PROOF OF LEMMA 5

引理 5. 上文构造的码 $\mathcal{C}_{q,a,b}$ 能够纠正一次 ℓ-受限幅度错误。

证明. 设 $\boldsymbol{u} = (u_1, \ldots, u_{n_r})$ 是一个原始码字，而 $\boldsymbol{v} = (v_1, \ldots, v_{n_r})$ 是由 $\boldsymbol{u}$ 经过至多一次 ℓ-受限幅度错误得到的字。也就是说，$\sum_{i=1}^{n_r} u_i - \sum_{i=1}^{n_r} v_i = t \leq \ell$。因此，如果 $t \neq 0$，则存在一个下标 i_0，使得 $u_{i_0} - v_{i_0} = t$. 由此，$\sum_{i=1}^{n_r} v_i + i_0 t = \sum_{i=1}^{n_r} u_i \equiv a$ 模 p.。下面证明满足上述条件的下标 i_0 是唯一的。假设存在两个下标 i_1 和 i_2，使得 $\sum_{i=1}^{n_r} v_i + i_1 t = \sum_{i=1}^{n_r} v_i + i_2 t \equiv a$ 模 p。则，$(i_2 - i_1)t \equiv 0$模 p.。这是不可能的，因为 p 是素数，并且 $(i_2 - i_1)\ t < p$.。因此，我们可以唯一确定下标 i_0，并恢复原始字 $\boldsymbol{u}$.。所以，码 $\mathcal{C}_{q,a,b}$能够纠正一次 ℓ-受限幅度错误。 □

APPENDIX D
PROOF OF THEOREM 4

定理 4。给定输入 $\boldsymbol{c} \in \Sigma_2^k$ 时，码 $\mathrm{Enc}(\boldsymbol{c}) \in \Sigma_2^N$ 的总冗余为

$$N - k = \frac{\lceil 2t(1-1/p) \rceil \cdot \lceil \log p \rceil}{\log p} \log(N+1) + O(\log\log N).$$

其中 p 是满足 $p \geq \ell + 1$ 的最小素数。

证明。令 $m = \log_p(N+1)$，因此 $N = p^m - 1$。冗余部分的长度如下：

- n_1'' 是 $g(\bar{c})$ 的长度：$n_1'' = \lceil 2t(1-1/p) \rceil m$;
- n_1' 是 $b(g(\bar{c}))$ 的长度：$n_1' = n_1'' \cdot \lceil \log p \rceil$;
- n_1 是 $s(b(g(\bar{c})))$ 的长度：$n_1 = n_1' + \log n_1' + 1$;
- n_2''是 $f_0(h_1(\bar{c}))$的长度：$n_2'' = t\ell \log n_1$;
- n_2'是 $s(f_0(h_1(\bar{c})))$的长度：$n_2' = n_2'' + \log n_2'' + 1$;
- n_2 是 $h_2(\bar{c})$的长度：$n_2 = (2t\ell + 1)n_2'$;

根据上述陈述，我们可以看出 $N - k = n_1 + n_2$，其中

$$n_1' = (\lceil 2t(1-1/p) \rceil m) \cdot \lceil \log p \rceil$$

其中 $m = \log_p(N+1)$。因此，我们有

$$n_1' = \frac{\lceil 2t(1-1/p) \rceil \cdot \lceil \log p \rceil}{\log p} \log(N+1)$$

由于 t 和 p 都是常数，因此$\log\ n_1' = O(\log\log\ N)$和 $n_2 = O(\log\log\ N)$。因此，给定输入 $\boldsymbol{c} \in \Sigma_2^k$ 时，码 $\mathrm{Enc}(\boldsymbol{c})$的总冗余 $\in \Sigma_2^N$可表述为定理 4。 □