基于质量分数和重新解码的DNA存储迭代软解码算法

Jaeho Jeong, *IEEE*学生会员, Hosung Park,*IEEE*会员, Hee-Youl Kwak, *IEEE*会员,Jong-Seon No,*IEEE*会士, Hahyeon Jeon,Jeong Wook Lee, and Jae-Won Kim, *IEEE*会员

arXiv:2304.03502v1 [cs.IT] 7 Apr 2023

摘要—**自从脱氧核糖核酸（DNA）被视为下一代数据存储介质以来，人们已开展大量研究工作，利用纠错码（纠错码）纠正合成、存储和测序过程中产生的错误。以往从含有错误的测序DNA池中恢复数据的工作，采用了基于多数决策规则的硬判决译码算法。为了提高纠错码的纠错能力和DNA存储系统的鲁棒性，我们提出了一种新的迭代软解码算法，其中软信息从FASTQ文件和信道统计信息中获得。具体而言，我们提出了一种利用质量分数（Q值）计算对数似然比（对数似然比）的新公式，以及一种可能适用于DNA测序领域纠错和错误检测的再解码方法。基于Erlich提出的广泛采用的喷泉码码结构编码方案等，我们使用三组不同的测序数据来展示性能评估的一致性。与当前最先进的解码方法相比，所提出的软解码算法使读段数量减少率提高了2.3%∼7.0%，并且结果表明，该算法能够处理带有插入错误和删除错误的错误测序寡核苷酸读段。**

索引词—**DNA存储、FASTQ文件、喷泉码、迭代解码、Luby变换码、寡核苷酸、质量分数、软解码**

I. 引言

在S大数据时代，海量数据正迅速产生，因此需要一种新型高密度存储介质，使其比现有介质更具竞争力

稿件于 2022 年 xxxxxx 接收；于 20xx 年 xxxx xx 修订；于 20xx 年 xxxx xx 接受。发布日期：20xx 年 xxxx xx；当前版本日期：20xx 年 xxxx xx。本研究部分由三星电子三星研究资助与孵化中心项目（编号 SRFC- IT1802-09）资助，部分由韩国国家研究基金会通过科学、ICT 与未来规划部资助的先锋研究中心计划（编号 NRF-2022M3C1A3081366）资助。（通讯作者：*Jae-Won Kim*。）

Jaeho Jeong 和 Jong-Seon No 就职于韩国首尔 08826 的首尔国立大学电气与计算机工程系、新媒体与通信研究所（INMC）（电子邮箱：jaehoj@ccl.snu.ac.kr；jsno@snu.ac.kr）。

Hosung Park 就职于韩国光州 61186 的全南国立大学计算机工程系和 ICT 融合系统工程系（电子邮箱：hpark1@jnu.ac.kr）。

Hee-Youl Kwak 就职于韩国蔚山 44610 的蔚山大学电气工程学院（电子邮箱：hykwak@ulsan.ac.kr）。

Hahyeon Jeon 就职于韩国蔚山 44919 的 Clinomics（电子邮箱：jeonhh0061@gmail.com）。

Jeong WookLee 就职于韩国浦项 37673 的浦项科技大学化学工程系（电子邮箱：jeongwook@postech.ac.kr）。J

ae-Won Kim is with D部门 o f El$_{ectron}$ ic Engineering, Engineering 研究所（ERI），庆尚国立大学，晋州 52828，韩国韩国（电子邮件：jaewon07.kim@gnu.ac.kr）。

数字对象标识符 xxxxxxxxxxxx

例如磁带或硬盘驱动器等存储介质。主要的信息技术公司正在投入巨额资金建设新的数据中心，以应对不断增长的存储需求。随着新的合成和测序技术的发展，脱氧核糖核酸（DNA）已被认为是新型存储介质的有力候选者 [1]。众所周知，与现有存储介质相比，DNA的信息密度最高可超过数亿倍，并且在温度保持适宜的情况下可以存活数千年 [2]。自 [1]，开创性研究以来，使用 DNA作为存储介质的想法引起了广泛关注，如今大量研究正在进行，以利用其长寿命和高信息密度的优势。

作为数据存储器件，我们考虑由四种核苷酸（即碱基）组成的单链DNA。这四种碱基分别是腺嘌呤、胞嘧啶、鸟嘌呤和胸腺嘧啶，分别简单表示为A、C、G和T。我们将这种单链DNA称为寡核苷酸，或简称寡核苷酸。对于DNA存储系统，写入和读取过程分别称为DNA合成和测序。为了通过测序读取存储在寡核苷酸中的信息，需要预先进行能够扩增信息的聚合酶链式反应（PCR）。事实上，每条寡核苷酸都会被大量合成，并且每条寡核苷酸的合成数量并不均匀。此外，在PCR扩增过程中，存储在寡核苷酸中的每条信息可能会被扩增到不同程度。通过测序设备，测序数据通常以计算机文件格式生成。其中，FASTQ文件已成为DNA测序领域的常见格式之一 [3]，因为它提供了质量分数（Q值），即每个碱基被可靠预测和测序的概率。在上述所有步骤中，可能会发生三类错误，即替换错误、插入错误和删除错误。这些错误的分布和行为与传统存储器件中的情况非常不同。传统存储器件主要考虑替换错误，而很少考虑其他类型的错误。为了处理这些错误，有必要为数据采用经过精心设计的纠错码（ECCs）。

为保证DNA存储的数据可靠性，数据需要满足一些生化约束，例如寡核苷酸长度、最大同聚物连续长度和GC含量 [4], [5]。寡核苷酸的长度通常为数百至数千个碱基，据报道，较长的寡核苷酸难以合成，并且在多个步骤中容易出错。在

[6] 和 [7], 讨论了最大同聚物连续长度和GC含量的影响。研究表明，寡核苷酸中较长的同聚物连续长度会增加错误率，而PCR扩增需要适中的GC含量，以避免较高的丢失率。此外， [8]–[11]中的研究指出，为降低DNA存储的错误率，还应考虑一些约束，例如最小自由能（MFE）约束、末端约束、自互补约束、寡核苷酸序列之间的距离以及地址之间的相关性。

此外，针对DNA存储系统中的新型编码方案，已有许多研究成果 [12]–[22]。Organick 等人 [15]设计了大量引物，并验证了单个文件可以被单独恢复。Yazdi等人 [16], 的研究引入了一种迭代比对算法，将所有类型的错误转换为删除错误；而 Erlich 等人 [17]通过采用喷泉码和 Reed-Solomon（RS）码的编码方案，达到了理论极限（即容量）的 86%。此外，Chandak 等人[18] 使用低密度奇偶校验（LDPC）码和基于对数似然比（LLR）的软解码，开发了一种单一的大型码结构。Press 等人 [19] 利用带有哈希函数的卷积码处理插入错误和删除错误。另外，Wang 等人 [20]仅使用一个引物结合位点（PBS）设计了一种寡核苷酸结构，以提高信息密度；Cao 等人 [21] 则采用基于喷泉码的自适应编码，以满足上述提到的若干约束 [6]–[11]。最后，我们先前的工作 [22] 通过结合多个预处理组件和解码技术，在编码方案与 [17]相同的情况下，提高了解码性能。

本文提出了一种基于 RS 码重新解码的迭代软解码算法。我们沿用 [17] 与先前工作 [22], [23]相同的编码方案，并通过修改解码算法展示性能提升。具体而言，我们利用 FASTQ 文件提供的软信息（即寡核苷酸读段的 Q 值）对接收码字（即测序结果）进行解码。在先前的工作 [17], 中，他们将 RS 码用作错误检测码，以检查寡核苷酸的测序读段是否出错。如果测序寡核苷酸读段出错，就会被丢弃，这意味着 [17] 中的解码方法需要更多的测序寡核苷酸读段。

从所提出算法的解码性能来看，我们表明，从FASTQ文件中获取的软信息可以减少所需的测序寡核苷酸读段数量，从而节省测序过程的读取成本。虽然用于软解码的对数似然比计算在 [18] 中基于硬值（即0和1的数量之差），但我们根据整个寡核苷酸读段的Q值和信道统计信息，确定对数似然比值。由于FASTQ文件仅提供每个位置上一个核苷酸的Q值，因此我们结合编辑距离，利用信道统计信息估计其他核苷酸的概率。根据寡核苷酸读段中每个位置上四种碱基的碱基判读概率，我们可以准确估计对数似然比值，并实现高性能软解码。

与我们的先前工作 [22]相比，在该工作中主要的解码过程保持不变； [17],本文提出了一种新颖的解码过程，概括如下:

- 我们首先提出了一种利用FASTQ文件中的软信息计算对数似然比的新公式，这是DNA存储领域的首次尝试，并可为DNA测序领域的多样化研究提供启发。
- 我们提出了一种新的迭代软解码算法，该算法使用基于RS解码结果的再解码方法。结果表明，这种再解码方法可以提取并丢弃含有插入和删除错误的错误测序寡核苷酸读段。
- 我们使用三组不同的测序数据来展示所提出的解码算法相比最先进算法 [22] 的性能提升。

本文的组织结构如下。在第二节中，我们介绍实验设计，包括合成和测序信息。我们还解释所提出的迭代软解码算法以及各项技术的工作原理。在第三节中，我们展示所提出的方法相较于现有解码方法的优越性，并对实验结果进行分析。在第四节中，我们讨论本文未能涵盖的若干问题，并展望未来的研究工作。最后，在第五节中给出结论。

II. 材料与方法

大多数先前的DNA存储研究在测序过程完成后，依赖硬解码进行解码。然而， [18] 中的研究表明，软解码可以应用于DNA存储系统。尽管我们采用了不同于 [18], 的码结构，但我们在实验中采用了对数似然比计算和软解码。

A. 实验设计

我们首先使用两组不同的测序数据集来展示所提出方法的灵活性和一致性。其中一组测序数据集是我们先前的工作 [22] 中使用的实验数据，我们将其称为 $dataA$。对于另一组数据集，称为 $dataB$，我们从 [22] 中剩余的合成池进行了一次测序实验，环境条件相同。

我们的编码结构如图1所示。正如我们在 [22], 中提到的 [24] ，我们使用Luby变换（LT）码作为寡核苷酸间码，并使用RS码作为寡核苷酸内码。每条寡核苷酸的长度为152个核苷酸（nt），其中每条寡核苷酸分别由16nt的种子、128nt的有效载荷和8nt的RS校验位组成。对于RS码，每个校验符号为4nt，我们在 GF（2^8）的有限域中使用（38, 36）RS码，其最小汉明距离为3，这意味着可进行1个符号纠正或2个符号检测。使用具有鲁棒孤子分布的LT编码 [17], , 我们从由513.6KB图像文件转换而来的16050条信息序列生成了18000条寡核苷酸序列。通过二进制数据与DNA碱基之间的映射，我们将两个连续比特转换为一个碱基，

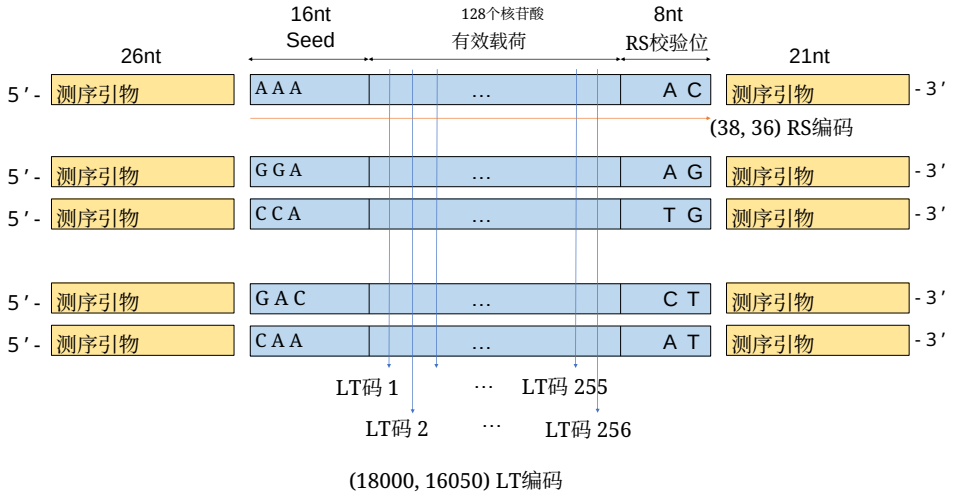

图1。我们实验的编码结构。我们使用（18000，16050）LT码作为寡核苷酸间码，使用（38，36）RS码作为寡核苷酸内码，其中四个核苷酸构成一个RS符号。

即 00 = A、 01 = C、 10 =G 和 11 = T。我们在每条寡核苷酸序列的两端添加两个序列引物：

- 5’端：GTTCAGAGTTCTACAGTCCGACGATC,
- 3侧：TGGAATTCTCGGGTGCCAAGG.

在 [22], 中，合成了两个寡核苷酸池，以展示同聚物连续长度和GC含量约束的影响。本节首先考虑非受限池（$dataA$ 和 $dataB$），它使我们能够在解码过程中获得整个寡核苷酸池的种子信息。非受限池是指编码时不考虑任何同聚物连续长度或GC含量约束。考虑非受限池的编码结构 [22], ，很容易发现，我们只需在编码过程中直接创建18000种不同的LT码种子结构。这意味着我们使用鲁棒孤子分布 [17]中的前18000个种子；在解码过程中按照相同的步骤计算前18000个种子，就能得到完全相同的种子结构。由于本文旨在提升解码性能，我们尝试限制其他条件，例如种子信息的准确性以及若干预处理步骤。因此，非受限池更适合用于比较不同的解码算法。此外，先前研究 [22],中非受限池的解码性能与受限池相当，这意味着，解码算法能够合理地纠正因未满足同聚物连续长度和GC含量约束而产生的错误。

对于使用 $dataB$ 的新实验，我们采用了与[22]中非受限池相同的测序流程。GC含量比例为 $[0.3289, 0.6842]$ ，最大同聚物连续长度为 13。我们使用 RP1 和 RPI1 引物进行 PCR 扩增，并使用 Illumina Miseq Reagent v3 试剂盒进行 600 个循环的测序，正向和反向各读取 151 nt。此次新测序实验的 Q30 为 96%，输出包含 2200 万条正向读段和 2200 万条反向读段，共计 4400 万条测序寡核苷酸读段，另有 36% 的 PhiX 加标。

B. 所提出的解码算法

所提出的解码过程如图2所示，下面的小节将逐一说明各个步骤。在先前的工作 [22], 中，我们使用寡核苷酸聚类，通过比较完整寡核苷酸序列的汉明距离，将相似的测序寡核苷酸读段归入同一个簇。在这项工作中，我们使用种子聚类，仅比较测序寡核苷酸读段的种子部分，将种子结构完全相同的测序寡核苷酸读段归入同一个簇。由于我们可以通过计算鲁棒孤子分布获得寡核苷酸池的种子信息，因此能够判断哪些种子结构有误、哪些种子结构正确（尽管仍可能存在未检测到的错误）。然后，我们只使用具有预先确定种子结构的簇，并丢弃种子部分不同的簇。利用这些种子信息，我们构建用于 LT 软解码的奇偶校验矩阵 H ，如图3所示。为了构建有效的奇偶校验矩阵，解码器必须知道预先确定的种子值，这就是我们考虑非受限池的原因。

对于 LT码，达到失败率 δ 所需的 LT解码符号数 K 可由[24]中的**定理 12** 推导如下：

$$K = k\beta = k(\sum_{i} \rho(i) + \tau(i)) = k + \sum_{i=1}^{k/R-1} \frac{R}{i} + R\ln(R/\delta),$$

其中， k 等于信息数据包的数量，且$R = c \times \ln(\frac{k}{\delta}) \times \sqrt{k}$为常数值 $c > 0$。通过设置能够反映我们实验条件的参数，我们可以得到 $K = 16951$，其中 $\delta = 0.001$ 和 $c = 0.025$。根据计算出的值 K，我们需要确保在解码过程中，参与 LT 解码的簇数量至少应为 K 或更多。在图 3中，信息位数量等于 16050，校验位数量应在范围 $16951 \leq K \leq 18000$内。之后

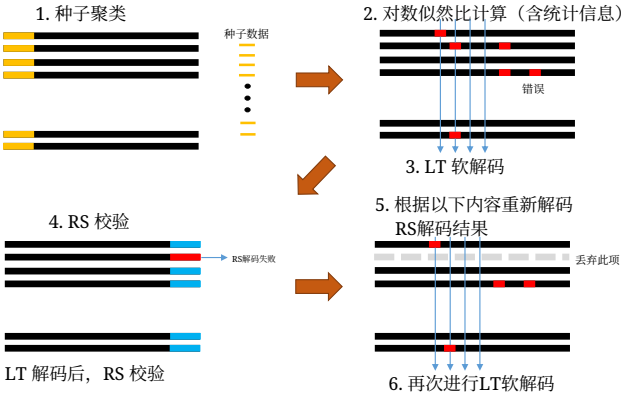

图2。所提出的解码过程说明。

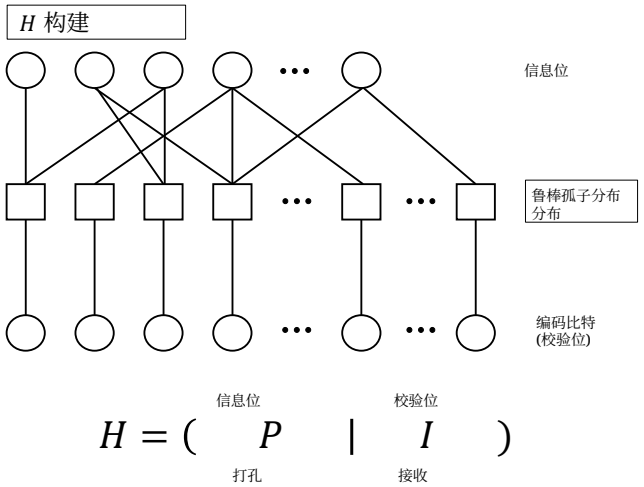

图 3. 奇偶校验矩阵的构造 H。

使用种子信息构建 H 矩阵后，我们使用该 H 矩阵进行信念传播（BP）解码 [25]，迭代解码 500 次。

我们的DNA寡核苷酸池具有LT码和RS码的乘积结构，因此我们可以选择先解码哪一种码。先前的工作 [22]先解码RS码，随后解码LT码，其解码过程与 [17]中的类似。在该工作中，我们使用RS码进行纠错；如果解码后的码字中存在错误，则该簇会对整个LT解码过程产生严重影响。这是因为LT码仅用于防止每条寡核苷酸序列的丢失，而无法为纠错提供任何帮助。然而，在本工作中，我们先解码LT码，随后解码RS码。我们对LT码采用软解码，并利用其进行纠错；同时，使用RS码进行纠错和检错。在这种情况下，如果LT纠错中存在错误，我们可以利用RS码将其检测出来。具体而言，我们将RS码用于两个目的：i）检测种子位置的错误；ii）纠正其他位置的错误。如果RS解码失败，或检测到种子位置存在错误，我们将丢弃该错误信息，并通过构建新的 H 矩阵、移除错误的寡核苷酸读段，再次执行LT软解码。在以下小节中，我们将详细介绍这些解码过程。

1）对数似然比计算：由于置信传播译码的输入是每个位的对数似然比值，因此我们需要从软信息 [26]中提取对数似然比值。在先前使用软解码 [18], 的工作中，他们基于碱基识别结果，使用硬值来计算对数似然比值。通过将碱基识别得到的硬值映射为0和1，他们统计了每个位置上一个簇中0和1的数量，并根据0和1的数量以及启发式确定的交叉概率计算对数似然比值。另一方面，我们尝试提取测序FASTQ文件中提供的软值，即Q值。

为获得准确的对数似然比值，我们使用寡核苷酸池中全部测序数据的错误分布统计信息。我们使用 PEAR 算法 [27] 收集所有长度正确（ 152 nt ）的拼接寡核苷酸读段，并将其与原始编码数据进行比较，以确定每条测序寡核苷酸读段中各类错误的发生情况。然后，对于每个位置，我们计算每种碱基转变组合的条件发生率，即 A→C、A→G、A→T、C→A、C→G、C→T、G→A、G→C、G→T、T→A、T→C 和 T→G，其中 X→Y 表示存储的是 X、测序结果却为 Y 的事件（X ≠ Y）。由于这些概率是条件概率（针对发生错误的情况），我们仅针对发生了任意错误的测序寡核苷酸读段计算条件概率。在解码过程中，我们只考虑位置错误，也就是说，将插入错误和删除错误视为突发替换错误来处理。当只收集长度正确的拼接寡核苷酸读段时，插入错误和删除错误的发生率远低于替换错误，因此这种处理方式是合理的。此外，从纠错码的角度来看，由于我们采用了 [17]中的编码方案，插入错误和删除错误无法纠正。因此，我们只使用长度正确（ 152 nt）的拼接寡核苷酸读段进行解码。按位置整理错误分布后，可以发现转变概率存在一些趋势（图4）；这一结果促使我们在计算对数似然比时，利用全部测序数据中位置错误的碱基转变条件概率。

现在，我们解释如何从完整的测序数据中推导条件碱基转换概率。令 $Z[n]=\{1,2,\ldots,n\}$ 对于正整数 n 以及 N_{read} 分别表示对完整测序数据运行PEAR算法后得到的拼接寡核苷酸读段数量。对于每个拼接寡核苷酸索引$j \in Z[N_{\text{read}}]$，$g(j)$表示基于编辑距离的最近编码寡核苷酸序列的索引。对于每个位置索引 $i \in Z[152]$ $b^{(1)}$ $y^{i\,j}=b^{(2)}$ $x^{m\,n}$ f_i $b^{(1)} \to b^{(2)}$ $=\Sigma_j I$ $x^{i\,g(j)}=$ m，我们定义 ( ) (, ,), 其中 , 是 nth 个编码寡核苷酸序列中第 th 个位置的碱基，$y^{m,\,n'}$ 是 $m \in Z[152]$,$n \in Z[18000]$时第m个拼接寡核苷酸读段中第 n' 个位置的碱基判读，并且 $n' \in Z[N_{\text{read}}]$, $I(\cdot)$ 表示指示函数，且 $b^{(1)}$，$b^{(2)} \in \{\text{A},\ \text{C},\ \text{G},\ \text{T}\}$ 满足 $b^{(1)} \neq b^{(2)}$。

令 $h(j)$表示与第 j 个拼接寡核苷酸读段相对应的真实编码寡核苷酸索引中的一个索引。然后，我们定义

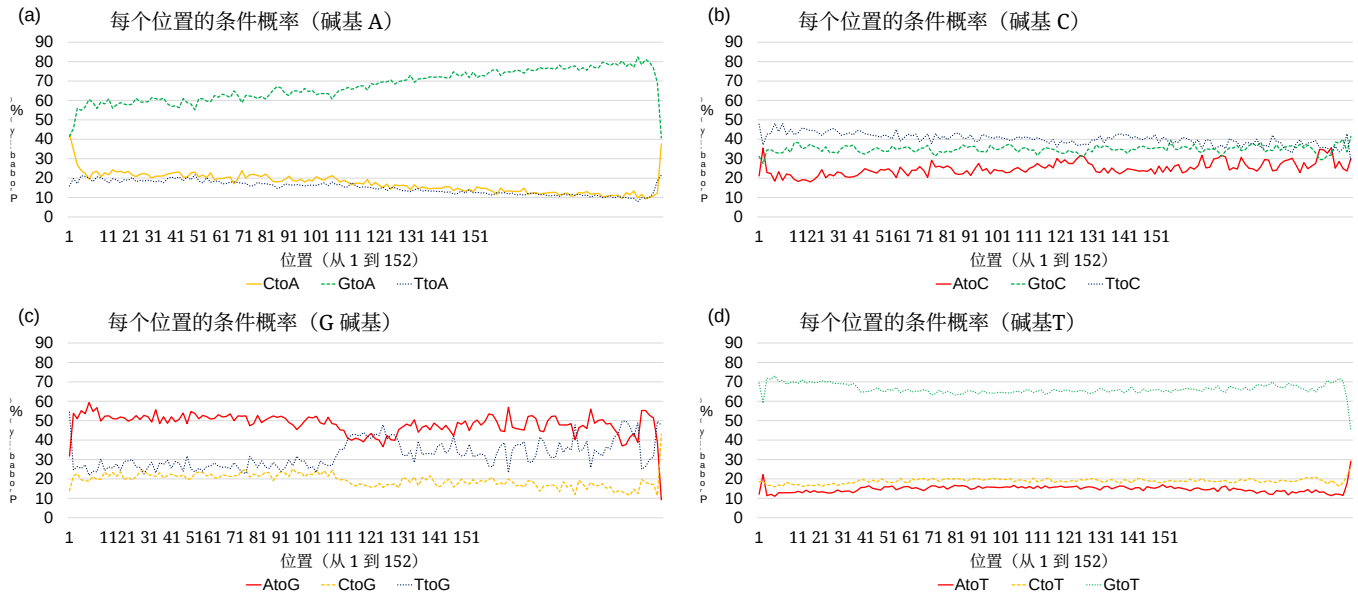

图4。 $dataB$中每个位置发生位置错误时碱基转变的条件概率图。图中的 XtoY 表示存储的是 X，但测序得到的是 Y 的概率（X $\neq$ Y）。概率数据从位置1到152给出。(a) 碱基 A 的条件概率。(b) 碱基 C 的条件概率。(c) G 碱基的条件概率。(d) 碱基T 的条件概率。注意，{C到A、G 到 A、T 到 A}、{A到C、G 到 C、T 到 C}、 {A到G、C到G、T到G}以及 {A到T、C到T、G到T} 的条件概率之和均为100%，因为它们表示每个位置发生错误时的条件碱基转变概率。

第 i个条件碱基转变率如下:

$$P_i(x^{i,h(j)} = b^{(1)} | y^{i,j} = b^{(2)}) = \frac{\frac{f_i(b^{(1)} \to b^{(2)})}{N_i(b^{(1)})}}{\frac{f_i(b^{(1)} \to b^{(2)})}{N_i(b^{(1)})} + \frac{f_i(b^{(3)} \to b^{(2)})}{N_i(b^{(3)})} + \frac{f_i(b^{(4)} \to b^{(2)})}{N_i(b^{(4)})}},$$

其中四种不同的 $b^{(1)}, b^{(2)}, b^{(3)}, b^{(4)} \in$ {A,C, G, T} 和$N_i(b^{(k)}) = \Sigma_j I(x^{i,g(j)} = b^{(k)})$，即，对应于 $k \in Z[4]$的拼接寡核苷酸读段的编码序列中 i位置的 $b^{(k)}$数量。整个测序数据的条件碱基转变概率如图4所示。

现在，我们解释如何使用Q值和信道统计信息计算对数似然比。根据种子信息将拼接寡核苷酸读段分类到簇后，我们计算每个簇的对数似然比值。令 $y^{i,}{}_1{}^j$ 和 $y^{i,}{}_2{}^j$ 表示映射 {A、C、G、T} $\to$ {00, 01, 10, 11}、 $i \in Z[152]$和 $j \in Z[N_{\text{read}}]$时，第 j个拼接寡核苷酸读段的第 i个位置的第1位和第2位。令 $P^{j,}{}_i{}^b(b') = P(x^{i,h(j)} = b' | y^{i,j} = b)$表示$b$， $b' \in$ {A、C、G、T}，我们根据FASTQ文件中的 $b = b'$ 确定该概率。概率推导如下 [28]:

$$P_i^{j,b}(b) = 1 - 10^{-\frac{Q_i^j(b)}{10}},$$

其中， $Q_i^j(b)$ 是 FASTQ文件中第j条拼接寡核苷酸读段的第 i个位置的 Q值（碱基判读为 b）。尽管 Q值无法体现DNA存储中的所有类型错误，但自然可以认为 Q值越高，可靠性越高。

然后，我们可以计算该位置碱基判读 b 对应的另外三种碱基的概率，如下所示:

$$P_i^{j,b}(b^{(k)}) = (1 - P_i^{j,b}(b)) \times P_i(x^{i,h(j)} = b^{(k)} | y^{i,j} = b),$$

其中 $k \in Z[3]$ 和 $b^{(k)} \in$ {A，C，G，T}，使得 $b^{(k)} \neq b$。

例如，假设在 FASTQ文件 中，第 t 个拼接寡核苷酸读段的第 s 个位置测得碱基 A，且其 Q值 为 10。此外，假设我们已知第 s 个位置的错误统计信息，使得条件概率为$P_s(x^{s,}{}_{h(t)} = \text{C} | y^{s,}{}_t = \text{A}) = 50\%$ $P_s(x^{s,}{}_{h(t)} = \text{G} | y^{s,}{}_t = \text{A}) = 25\%$ $P_s(x^{s,}{}_{h(t)} = \text{T} | y^{s,}{}_t = \text{A}) = 25\%$。然后，我们可以得到以下概率:

$$\begin{aligned} P_s^{t,\text{A}}(\text{A}) &= 1 - 10^{-\frac{10}{10}} = 0.9, \\ P_s^{t,\text{A}}(\text{C}) &= (1 - P_s^{t,\text{A}}(\text{A})) \times 0.5 = 0.05, \\ P_s^{t,\text{A}}(\text{G}) &= (1 - P_s^{t,\text{A}}(\text{A})) \times 0.25 = 0.025, \\ P_s^{t,\text{A}}(\text{T}) &= (1 - P_s^{t,\text{A}}(\text{A})) \times 0.25 = 0.025. \end{aligned}$$

最后，考虑映射 {A，C，G，T} $\to$ {00, 01, 10, 11}，我们可以按如下方式计算碱基判读b 的各比特的对数似然比:

$$\text{LLR}(y_1^{i,j}) = \log \frac{P_i^{j,b}(\text{A}) + P_i^{j,b}(\text{C})}{P_i^{j,b}(\text{G}) + P_i^{j,b}(\text{T})},$$

$$\text{LLR}(y_2^{i,j}) = \log \frac{P_i^{j,b}(\text{A}) + P_i^{j,b}(\text{G})}{P_i^{j,b}(\text{C}) + P_i^{j,b}(\text{T})},$$

这些内容源自先前的工作 [29]。

基于预先确定的种子信息，我们可以丢弃种子信息（碱基识别）不正确的拼接寡核苷酸读段，不将其用于对数似然比计算和解码。对于寡核苷酸读段

对于具有相同种子信息的寡核苷酸读段，我们只需将它们的对数似然比相加，用于LT软解码。由于软解码仅用于LT码，而种子位置使用碱基识别信息，因此只需对128个有效载荷位置进行对数似然比计算。对于RS校验位，我们利用Q值，相关内容将在第二节-B3中说明。

2）基于 *RS* 解码结果的再解码：如前所述，我们首先进行 LT 软解码，然后使用 RS 码进行纠错和检错。利用根据种子信息 H 构建的校验矩阵完成 LT 软解码后，我们对解码得到的寡核苷酸进行 RS 解码。如果所有解码得到的寡核苷酸均成功完成 RS 解码，且不存在种子位置错误，我们便从二次解码后的寡核苷酸中恢复数据，并判定整体解码成功或失败。然而，大多数情况下无法同时满足上述条件。为此，我们提出一种基于 RS 解码结果进行重新解码的方法。对于 RS 解码失败或种子位置存在任何检错结果的 LT 解码寡核苷酸，我们会丢弃相应的寡核苷酸簇，然后再次进行 LT 软解码。这意味着我们不会将这些经 RS 检查后判定有误的簇用于解码过程，而是针对其余寡核苷酸读段，使用相同的对数似然比计算，构建一个新的 H 矩阵。此外，由于我们仅使用具有预先确定种子值的寡核苷酸读段，因此自然应将种子位置存在错误的 RS 纠正后寡核苷酸视为错误纠正的寡核苷酸。随后，我们实际发现，仍有一些 RS 纠正后的寡核苷酸簇在种子部分发生了纠正；在实验中，这些簇都被准确剔除。

事实上，这种重新解码操作有助于防止错误信息参与解码过程，这表明最好不要使用错误信息。排除这些未通过 RS 检查的簇后重新解码，仍可能存在其他不满足上述条件的寡核苷酸簇。因此，我们会迭代执行再解码方法数次。

3) 迭代解码：现在，我们介绍算法 1中所示的所提出的迭代软判决解码方法。对于输入，在PEAR算法之后 [27], 我们使用长度正确（152nt）的拼接寡核苷酸读段，并且仅使用与编码过程中预先确定的种子值具有完全相同种子信息的寡核苷酸读段。作为预处理步骤，我们计算有效载荷部分中每个位置的第 1 位和第2 位的对数似然比。另一方面，对于 RS 部分，我们需要使用硬值，因此我们根据 $\Pi_j P_i^j$(A) $\Pi_j P_i^j$(C) $\Pi_j P_i^j$(G)和 $\Pi_j P_i^j$(T) 的最大值确定核苷酸的硬值，其中 $P_i^j(b)$是 $P^{j}{}_{,i}{}^{b'}(b)$，表示与目标寡核苷酸簇对应的第 j个拼接寡核苷酸读段的第 i 个位置的碱基判定值 b'，以及$b, b' \in$ {A,C,G,T}。此外，在算法 1 的步骤 4 中，我们使用$n_{\mathrm{re}} = 3$ 作为重新解码次数。在大多数情况下，增加重新解码次数 n_{re} 可以提高解码性能。然而，同时考虑解码性能的提升和计算复杂度，我们通过启发式方法选择一个适当的次数 $n_{\mathrm{re}} = 3$ 。这使得最坏情况下的迭代解码最大次数为 3；但是，如果我们利用增加的计算复杂度，则可以调整此参数。

算法 1所提出的迭代软解码算法。**输入：** L_m 经 PEAR 算法处理后随机抽取的拼接寡核苷酸读段、预先确定的种子值、最大再解码次数 n_{re}，以及预处理步骤得到的这些拼接寡核苷酸读段的对数似然比和 RS 校验位。**输出：** 16050×256 信息位。**初始化：** $i = 0$。步骤 1）对 256 个二进制 LT 码执行软解码（=128nt）。步骤 2）对每条解码后的寡核苷酸执行 RS 解码。步骤 3）如果所有 RS 解码均成功，且种子位置没有错误，则利用解码后的校验位找到打孔的信息位，并跳过后续步骤。步骤 4）如果 $i = n_{\mathrm{re}}$，则判定解码失败，并跳过下一步。步骤 5）如果 $i < n_{\mathrm{re}}$ ，且存在 RS 解码失败或种子位置错误的寡核苷酸，则移除这些寡核苷酸并重建 H 矩阵。然后，使用其他比特的初始对数似然比以及 $i \leftarrow i + 1$，转到步骤 1）。

III. 结果

为展示所提出的解码算法相比最先进算法的优越性，我们针对两组测序数据采用了两种解码算法。一种是先前工作中的硬判决解码方法 [22] ，另一种是所提出的软判决解码方法。为进行公平比较，我们使用$dataA$和 $dataB$进行这些实验，并作出如第二节-A中所述的已知种子信息的假设。

A. 实验结果

我们首先随机采样少量测序寡核苷酸读段，并在每个采样点增加2000条测序寡核苷酸读段，以寻找完美恢复点。需要注意的是，由于寡核苷酸长度（152nt）和种子位置信息均为确定值，算法1中的 L_m小于或等于表I中的随机采样数。对于每个随机采样数，我们进行50次试验，并统计其中解码成功的次数。结果汇总于表I，并如图5所示。

结果表明，所提出的方法在更低的随机采样数下即可达到完美恢复点，这意味着具有更好的解码性能。与 [22], 中的硬判决解码方法相比，我们可以减少 2000∼6000条测序寡核苷酸读段，即读段数量减少2.3%∼7.0%。事实上，在我们的实验中，我们确认较少的测序寡核苷酸读段意味着读取成本的降低。我们的实验采用Illumina测序设备，并且每次测序过程始终需要使用一个Illumina流动槽。然而，解码成功所需的寡核苷酸读段数量更少，意味着我们可以使用流动槽中更小的部分来读取数据。

此外，尽管我们在此过程中没有引入任何其他预处理方法，但我们的解码算法仍然更优。

表I 所提出的软解码算法与先前的硬判决译码算法各进行50次试验的实验结果

	$dataA$		$dataB$	
随机采样数	所提出的soft解码	先前的hard解码[22]	所提出的soft解码	先前的hard解码[22]
72k	16	11		
74k	41	22		
76k	45	38	16	10
78k	48	42	32	22
80k	**50**	45	43	32
82k	**50**	49	48	42
84k	**50**	49	49	48
86k	**50**	**50**	**50**	49
88k			**50**	**50**

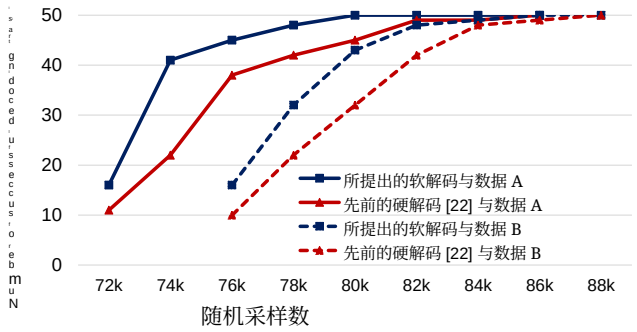

图5. 在 $dataA$ 和 $dataB$ 中，所提出的软解码与先前的硬解码算法的解码结果，每种算法均进行了50次试验。

聚类步骤中未引入任何其他预处理方法，我们的解码算法仍然更优。如果像 [22] 软判决解码方法中那样添加并修改其他预处理方法，例如基于汉明距离的聚类，或基于汉明距离剔除测序错误的寡核苷酸读段，我们或许能取得比表I所示当前结果更好的解码性能。请注意，在实验中我们共有 18000 种寡核苷酸序列，并且 2000∼6000测序寡核苷酸读段是测序环境中的深度片段。此外，所提出的软解码算法在两种测序数据上均优于最先进的算法，这表明我们的算法具有一致性。

B. 分析

在相同随机采样点上， $dataA$的解码性能优于 $dataB$ 的主要原因是：如表II所示，经过比对算法后， $dataA$ 和 $dataB$ 中完美拼接的寡核苷酸读段比例不同。我们使用PEAR算法 [27] 拼接正向读段和反向读段； $dataA$ 中83.83%的测序寡核苷酸读段被合并为正确长度（152nt），而 $dataB$ 中这一比例为78.64%。由于我们在运行PEAR算法之前对测序寡核苷酸读段进行采样，因此运行PEAR算法后， $dataA$ 中的拼接寡核苷酸读段会多于 $dataB$中的。

关于错误统计，我们收集了所有长度正确的拼接寡核苷酸读段，并将其与表II中的原始数据进行比较。两组测序数据的错误情况存在差异，这也会影响解码性能。此外，这表明，对于

表II $dataA$ 和 $dataB$ 中长度正确（152nt）的拼接寡核苷酸读段的统计特性比较

统计数据	$dataA$	$dataB$
长度正确（152nt）PEAR 通过率	83.83%	78.64%
平均数量每个碱基的错误位置数	1.100×10^{-3}	9.604×10^{-4}
平均数量每个碱基的插入与删除错误数	1.237×10^{-5}	1.744×10^{-5}
平均数量每个碱基的替换错误数	9.858×10^{-4}	8.352×10^{-4}

表III 所提出的软判决解码方法中各解码技术的效果 $dataB$ （对数似然比计算与重新解码）

LLR计算	Chandak的方法 [18]		所提出的方法	
随机采样数字	没有重新解码	with 重新解码	没有重新解码	with 重新解码
76k	0	2	0	**16**
78k	0	21	4	**32**
80k	9	38	10	**43**
82k	19	46	19	**48**
84k	23	48	21	**49**
86k	22	48	24	**50**

同一个合成池和相同的测序方法，每次测序的错误统计也可能略有不同。顺便一提，两组测序数据的错误率在Illumina测序环境中都处于可接受的水平。

为了展示所提出的解码算法中各项解码技术的作用，我们在表III中开展了更多实验 $dataB$ 。对于所提出的对数似然比计算方法，我们将其与基于碱基判读系统的Chandak软解码方法 [18] 进行比较。对于再解码方法，我们分别采用结合再解码系统和不结合再解码系统的方式，实施这两种软解码方法。表III显示，所提出的软解码方法表现最佳，且每项技术对我们的贡献都不可或缺。具体而言，我们发现，没有再解码系统时，DNA存储系统的信道非常不稳定，甚至在某些点上，Chandak的方法会导致性能逆转。关于对数似然比计算，值得注意的是，我们的方法在大多数情况下都更优。因此，我们可以认为，利用软信息（从测序数据中提取的Q值和信道统计信息）的软解码方法对DNA存储系统非常有用。此外，我们注意到，与对数似然比计算的新公式相比，再解码方法在这里贡献更大。没有再解码系统时，我们的性能甚至不如表I中的先前工作 [22]。

IV. 讨论与未来工作

在第三节中，由于解码器已知预先确定的种子值，我们考虑了两种非约束数据（$dataA$和 $dataB$）。然而，人们可能会想知道，所提出的解码算法带来的性能提升是否仅对非约束数据有效。因此，我们使用 [22] 中受限池的实验数据，进行了50次试验，仅添加了

表 IV 所提出的软解码算法与先前的硬解码算法在受限池数据上各进行 50次试验的实验结果 [22]

[22]中受限池的实验数据		
随机采样数	所提出的软解码	先前的硬解码 [22]
72k	12	10
74k	28	22
76k	43	37
78k	49	43
80k	50	49
82k	50	50

假设我们知道受限池的种子信息（表IV）。我们可以看到，在受限池中，我们的软解码算法也优于硬解码算- 法。这表明，无论同聚物连续长度和GC含量约束如何，我们的解码算法都更优。

除了上述算法之外，我们还尝试了各种方法来改善解码性能。正如我们在先前的工作中曾使用Q值的乘积来对寡核苷酸簇组进行排序 [22], 我们也尝试将其用于软解码算法。表达式变为：

$$P_j = \prod_{i=1}^{152} P_i^{j,b}(b) = \prod_{i=1}^{152}(1 - 10^{-\frac{Q_i^j(b)}{10}}),$$

此外，我们还根据图6中原始编码数据，计算了该 P_j 值与每个寡核苷酸簇的位置错误数量之间的相关性。这一次，我们发现，一些具有较高Q值且错误较多的寡核苷酸读段（在图6中用红色圆圈标出）在重新解码过程中被丢弃，其中许多读段含有插入和删除错误。这意味着，所提出的迭代软判决解码方法能够有效处理一些插入和删除错误。同时，我们尝试在解码阶段丢弃一些Q值较低的寡核苷酸读段，但效果相对有限（在表I的50次试验中，解码成功次数仅多出0次或1次）；该方法在随机采样点较多时有效，但在随机采样点较少时有时会导致性能变差。然而，由于Q值的乘积与错误数量之间存在相关性，丢弃一些Q值较低的寡核苷酸读段可能是有效的。因此，我们认为，弄清这种相关性与解码性能之间的关系是一项重要的未来工作。

此外，当我们确定对数似然比计算公式时，我们观察到，根据每个寡核苷酸簇的大小分别提供不同的对数似然比可靠性，对于某些随机采样点可以获得更好的解码性能。具体而言，我们为大小为1的簇分配最低的对数似然比值，为大小为 2 的簇分配稍大的对数似然比值，甚至为大小为三及以上的簇提供更多更大的对数似然比值，这最终会略微改善我们的解码性能。我们认为，这是因为较大簇中的信息比小簇中的信息更值得信赖。我们无法找到其他理论线索和实验标准来确定要相乘的变量数量

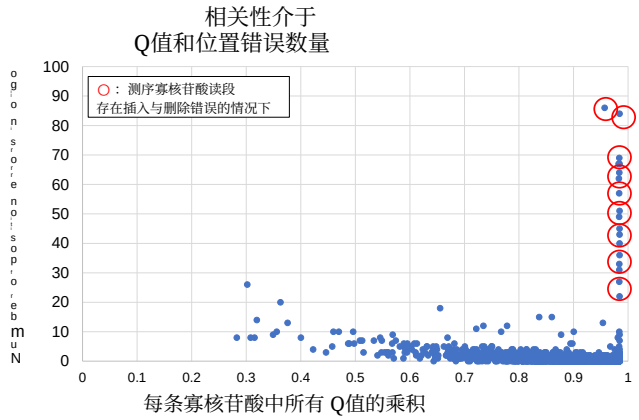

图6. 每个寡核苷酸簇中Q值与位置错误数量之间的相关性。该图取自50次试验中的一次，所示为成功解码点，即 $dataB$的86k。

以减少对数似然比值，将差分分布留待未来工作进行对数似然比计算。

事实上，与硬判决解码方法相比，软解码在计算能力和计算复杂度方面可能需要更高的成本。然而，DNA存储中的合成和测序过程需要更多资源和时间，因此，略微增加计算操作不会对整个流程产生显著影响。通过付出少量计算能力方面的成本，我们可以节省大量读取DNA序列的成本，甚至可能提高整个数据的码率。由于DNA存储的主要目标是冷数据，我们认为这一假设是合理的。

事实上，有几种纠错码在软解码方面的性能优于LT码。在喷泉码中，RaptorQ码 [30]作为LT码的改进型码已被开发出来。此外，低密度奇偶校验码以其出色的软解码纠错性能而闻名[31]。我们在[17] 中采用的LT码和RS码编码结构是为硬判决解码方法设计的，并未考虑软解码方法。因此，我们确信，如果使用经过精心设计、适配所计算软信息的纠错码开展另一项实验，所提出的算法相较于现有解码方法将显著提升性能。此外，使用LT码的优势在于可以控制同聚物连续长度和GC含量方面的一些约束。然而，如 [22], 所示，这些约束也可以通过信号处理算法加以管理。因此，应考虑在DNA存储中采用不使用LT码的其他方法。从纠错码的角度来看，极端情况下可以采用软解码性能最佳、能够处理位置错误且不受其他约束影响的纠错码。

V. 结论

在本文中，我们为DNA存储系统提出了一种新的迭代软解码算法，可以从FASTQ文件和信道统计信息中获取软信息。通过

利用条件碱基转变率和Q值，我们为拼接寡核苷酸读段中每个位置推导出四种碱基的概率，并提出了一种新的对数似然比计算方法，可用于DNA测序领域的纠错与检测。所提出的解码算法采用了结合再解码系统的LT码软解码。就所需的测序寡核苷酸读段数量而言，该算法优于 [22]中的最先进算法和 [18] 中现有的软解码算法，这意味着所提出的解码算法能够降低测序过程的读取成本。

参考文献

[1] G. M. Church、Y. Gao 和 S. Kosuri，“DNA 中的下一代数字信息存储”，*Science*，第 337 卷，1628 页，2012 年第 6102 期，页码 1628。[2] Y. Dong、F. Sun、Z. Ping、Q. Ouyang 和 L. Qian，“DNA存储：研究现状与未来展望”，*National Science Review*，第 7 卷第 6 期，1092–1107 页，2020 年。[3] P. J. A. Cock、C. J. Fields、N. Goto、M. L. Heuer 和 P. M. Rice，“带质量分数序列的 Sanger FASTQ文件格式及 Solexa/Illumina FASTQ 变体”，*Nucleic Acids Research*，第 38 卷第 6 期，1767–1771 页，2010 年。[4] C. Xu、C. Zhao、B. Ma 和 H. Liu，“合成 DNA 数据存储中的不确定性”，*Nucleic Acids Research*，第 49 卷第 10 期，5451–5469 页，2021 年。[5] A. El-Shaikh、M. Welzel、D. Heider 和 B. Seeger，“DNA存储系统中的大规模随机访问”，*NAR Genomics and Bioinformatics*，第 4 卷第 1 期，2022 年，文章编号 lqab126。[6] M. G. Ross、C. Russ、M. Costello、A. Hollinger、N. J. Lennon、R. Hegarty、C. Nusbaum 和 D. B. Jaffe，“序列数据偏差的表征与测量”，*Genome Biology*，第 14 卷第 5 期，1–20 页，2013 年。[7] J. J. Schwartz、C. Lee 和 J. Shendure，“通过标签定向检索经序列验证的 DNA 分子，实现准确的基因合成”，*Nature Methods*，第 9 卷第 9 期，913–915 页，2012 年。[8] B. Cao、X. Zhang、J. Wu、B. Wang、Q. Zhang 和 X. Wei，“用于 DNA存储的最小自由能编码”，*IEEE Transactions on NanoBio- science*，第 20 卷第 2 期，212–222 页，2021 年。[9] B. Cao、X. Li、X. Zhang、B. Wang、Q. Zhang 和 X. Wei，“通过 DMVO 算法为 DNA存储设计无相关地址约束”，*IEEE/ACMTransactionsonComputationalBiologyandBioinformatics*，第 19 卷第 2 期，866–877 页，2022 年。[10] J. Wu、Y. Zheng、B. Wang 和 Q. Zhang，“通过末端约束增强 DNA存储集合的物理与热力学特性”，*IEEE Transactions on NanoBioscience*，第 21 卷，193 页，2022 年第 2 期，页码 184。[11] Q. Yin、Y. Zheng、B. Wang 和 Q. Zhang，“用于归档 DNA存储的约束编码集合设计”，*IEEE/ACM Transactions on Computational Biology and Bioinformatics*，第 19 卷第 6 期，3384–3394 页，2021 年。[12] N. Goldman、P. Bertone、S. Chen、C. Dessimoz、E. M. LeProust、B. Sipos 和 E. Birney，“迈向合成 DNA 中实用、高容量、低维护成本的信息存储”，*Nature*，第 494 卷第 7435 期，77—80 页，2013 年。[13] R. N. Grass、R. Heckel、M. Puddu、D. Paunescu 和 W. J. Stark，“利用纠错码在二氧化硅中稳健地化学保存 DNA 上的数字信息”，*Angewandte Chemie International Edition*，第 54 卷第 8 期，2552—2555 页，2015 年。[14] J. Bornholt、R. Lopez、D. M. Carmean、L. Ceze、G. Seelig 和 K. Strauss，“基于 DNA 的归档存储系统”，载于*Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*，2016 年，637—649 页。[15] L. Organick、S. D. Ang、Y.-J. Chen、R. Lopez、S. Yekhanin、K. Makarychev、M. Z. Racz、G. Kamath、P. Gopalan、B. Nguyen、C. N. Takahashi、S. Newman、H.-Y. Parker、C. Rashtchian、K. Stewart、G. Gupta、R. Carlson、J. Mulligan、D. Carmean、G. Seelig、L. Ceze 和 K. Strauss，“大规模 DNA 数据存储中的随机访问”，*Nature Biotechnology*，第 36 卷第 3 期，242–248 页，2018 年。

[16] S. M. H. T. Yazdi、R. Gabrys 和 O. Milenkovic，“便携式且无错误的基于 DNA 的数据存储”，*Scientific Reports*，第 7 卷，6 页，2017 年第 1 期，页码 1。[17] Y. Erlich 和 D. Zielinski，“DNA Fountain 实现稳健高效的存储架构”，*Science*，第 355 卷第 6328 期，950–954 页，2017 年。[18] S. Chandak、K. Tatwawadi、B. Lau、J. Mardia、M. Kubit、J. Neu、P. Griffin、M. Wootters、T. Weissman 和 H. Ji，“使用低密度奇偶校验码改进基于 DNA 的数据存储读写成本权衡”，载于*201957th Annual Allerton Conference on Communication, Control, and Computing(Allerton)*，IEEE，2019 年，147–156 页。[19] W. H. Press、J. A. Hawkins、S. K. Jones. Jr、J. M. Schaub 和 I. J. Finkelstein，“用于 DNA存储的 HEDGES 纠错码可纠正插入、缺失和替换错误，并允许序列约束”，载于*Proceedings of the National Academy of Sciences*，第 117 卷第 31 期，18489–18496 页，2020 年。[20] Y. Wang、M. Noor-A-Rahim、J. Zhang、E. Gunawan、Y. L. Guan 和 C. L. Poh，“采用单个引物结合位点的寡核苷酸设计，实现高容量基于 DNA 的数据存储”，*IEEE/ACM Transactions on Computational Biology and Bioinformatics*，第 17 卷第 6 期，2176–2182 页，2019 年。[21] B. Cao、X. Zhang、S. Cui 和 Q. Zhang，“用于高存储密度和低覆盖率 DNA存储的自适应编码”，*NPJ Systems Biology and Applications*，第 8 卷第 1 期，1–12 页，2022 年。[22] J. Jeong、S.-J. Park、J.-W. Kim、J.-S. No、H. H. Jeon、J. W. Lee、A. No、S. Kim 和 H. Park，“基于喷泉码的 DNA存储系统中的协同序列聚类与解码”，*Bioinformatics*，第 37 卷第 19 期，3136–3143 页，2021 年。[23] S. Kang、Y. Gao、J. Jeong、S.-J. Park、J.-W. Kim、J.-S. No、H. Jeon、J. W. Lee、S. Kim、H. Park 和 A. No，“用于 DNA存储信道模拟器的生成对抗网络”，*IEEE Access*，第 11 卷，3793 页，2023 年，页码 3781。[24] M. Luby，“LT码”，载于*Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002*，IEEE Computer Society，271–271 页，2002 年。[25] Y. Han 和 W. E. Ryan，“低误码平台解码器：用于低密度奇偶校验码”，*IEEE Transactions on Communications*，第 57 卷第 6 期，1663–1673 页，2009 年。[26] R. Palanki 和 J. S. Yedidia，“噪声信道上的无速率码”，载于*Proceedings of IEEE International Symposium on Information Theory (ISIT), 2004*，IEEE，37 页，2004 年。[27] J. Zhang、K. Kobert、T. Flouri 和 A. Stamatakis，“PEAR：一种快速准确的 Illumina 双端读段合并器”，*Bioinformatics*，第 30 卷第 5 期，614–620 页，2014 年。[28] Illumina Inc.，“bcl2fastq 转换用户指南”，Illumina Inc.，美国加利福尼亚州圣迭戈。版本 1.8.4，第 23–24 页。[29] X. Lu、J. Jeong、J.-W. Kim、J.-S. No、H. Park、A. No 和 S. Kim，“DNA存储中基于错误率的低密度奇偶校验码对数似然比处理”，*IEEE Access*，第 8 卷，162892–162902 页，2020 年。[30] A. Shokrollahi，“Raptor 码”，*IEEE Transactions on Information Theory*，第 52 卷第 6 期，2551–2567 页，2006 年。[31] D. J. C. MacKay 和 R. M. Neal，“低密度奇偶校验码接近香农极限的性能”，*Electronics Letters*，第 33 卷第 6 期，457–458 页，1997 年。