第 14 卷 第 3 期
2025 年 5 月
集 成 技 术
JOURNAL OF INTEGRATION TECHNOLOGY
Vol. 14 No. 3
May 2025

引文格式：
袁涛, 曲强. 面向 DNA 安全存储的信息增量管理方法 [J]. 集成技术, 2025, 14(3): 38-50.
Yuan T, Qu Q. An incremental information management method for secure DNA storage [J]. Journal of Integration Technology, 2025, 14(3): 38-50.

面向 DNA 安全存储的信息增量管理方法

袁 涛[1,2] 曲 强[2*]

[1](南方科技大学 深圳 518055)
[2](中国科学院深圳先进技术研究院 深圳 518055)

摘 要 在大数据时代背景下，海量数据的存储成为难题。DNA 存储技术作为应对数据存储挑战的前沿解决方案，特别关注信息编辑技术的发展与挑战。初期的 DNA 存储主要服务于“冷”数据，而技术的最新进展已推动其向支持数据更新和管理的进阶应用发展。本文提出一种面向 DNA 安全存储的增量管理方法，设计了支持多方编辑的混合型加密机制和 DNA 增量存储模型。在保证安全的前提下，该模型通过分区存储方案和高效索引编码，在现有技术约束下实现安全高效的信息编辑与管理，可满足现代数据管理对灵活性和经济性的要求，为解决 DNA 数据管理中的核心问题提供了新视角和策略。

关键词 DNA 存储；信息编辑；加密；增量存储
中图分类号 TP301 **文献标志码** A doi: 10.12146/j.issn.2095-3135.20240612001
CSTR: 32239.14.j.issn.2095-3135.20240612001

An Incremental Information Management Method for Secure DNA Storage

YUAN Tao[1,2] QU Qiang[2*]

[1](*Southern University of Science and Technology*, *Shenzhen* 518055, *China*)
[2](*Shenzhen Institutes of Advanced Technology*, *Chinese Academy of Sciences*, *Shenzhen* 518055, *China*)
[*]*Corresponding Author*: *qiang@siat.ac.cn*

Abstract In the era of big data, the storage of massive amounts of data has become a challenging problem. DNA storage technology, as a cutting-edge solution to this challenge, particularly focuses on the development and challenges of information editing technology. Initially, DNA storage primarily served “cold” data, but the latest advancements in the technology have driven its development towards supporting data updates and management for more advanced applications. This paper proposes an incremental management method for secure DNA storage, designing a hybrid encryption mechanism that supports multi-party editing and a DNA

收稿日期：2024-06-12 **修回日期**：2024-10-31
基金项目：国家重点研发计划项目 (2020YFA0909100)
作者简介：袁涛，硕士研究生，研究方向为 DNA 存储； 曲强 (通讯作者)，研究员，研究方向为区块链、DNA 存储，E-mail：qiang@siat.ac.cn。

incremental storage model. While ensuring security, this model achieves secure and efficient information editing and management under existing technological constraints through a partitioned storage scheme and efficient indexing encoding. This approach meets the modern data management requirements for flexibility and cost-effectiveness, providing new perspectives and strategies for addressing core issues in DNA data management.

Keywords　DNA storage; information editing; encryption; incremental storage

Funding　This work is supported by National Key Research and Development Program of China (2020YFA0909100)

1　引　言

在当前这个数据量呈爆炸性增长的时代，信息的海量累积对传统的数据存储与管理技术提出了前所未有的挑战。DNA 分子作为一种前沿的新型信息存储介质，具有卓越的存储密度、超长的使用寿命和极小的维护成本[1-3]等优点，性能显著优于传统物理存储介质，因而备受瞩目。DNA 存储技术利用 DNA 分子中碱基对的有序排列承载数字化信息，利用 DNA 分子作为存储介质的诸多优点，为解决海量数据的存储和应用难题提供了“破局”之法[4]。

传统 DNA 数据存储技术主要面向“冷”数据进行存储，如需长期保存且较少更新的档案文件和关键数据备份等。但随着相关技术的发展，DNA 存储技术也有更多进阶应用，信息编辑构成了其中一个核心环节，即对已转化成 DNA 序列的数据信息进行修正与更新处理，并确保修改后信息能被重新存档。DNA 存储信息编辑研究目前尚处于理论与实践探索的初级阶段，有关高效编辑策略的研究文献较少。当前主导的信息编辑技术框架主要围绕 3 大核心技术展开：

(1) 核酸分子杂交技术[5]，是生物学与化学交叉领域的传统工具；

(2) 成簇规律间隔短回文重复序列/CRISPR 相关蛋白 (clustered regularly interspaced short palindromic repeats/CRISPR-associated protein，CRISPR/Cas) 基因组编辑技术[6]，具有卓越的精确度；

(3) 重叠延伸聚合酶链式反应 (polymerase chain reaction，PCR) 技术[7]，用于平衡成本与效益。

前两种技术凭借其高效的编辑能力脱颖而出，但伴随而来的是较高的经济成本；相反，重叠延伸 PCR 技术虽然成本低廉，但在操作效率上有所欠缺。

2015 年，Tabatabaei Yazdi 等[8]同时使用 CRISPR/Cas 基因组编辑技术和重叠延伸 PCR 技术进行信息编辑，实现了 DNA 存储介质中文件内容的直接且精确的修订。2019 年，Appuswamy 等[9]通过改进重叠延伸 PCR 技术，在 DNA 分子级别上成功执行了数据库记录的“链接”操作，拓展了 DNA 存储技术在数据库管理中的应用范围。2020 年，Lee 等[10]使用重叠延伸 PCR 技术对合成后的 DNA 链进行修改，纠正其合成时产生的碱基错误。同年，Lin 等[11]通过核酸分子杂交技术对存储后多个文件中的特定文件进行锁定、解锁，使 DNA 存储支持单个文件的重命名和删除操作。2022 年，Liu 等[12]运用 CRISPR/Cas 技术，在活细胞内不仅实现了完整信息的写入，

还完成了高精度的编辑操作，标志着多模态信息在活细胞环境中的编辑成为可能。紧接着 2023 年，Sadremomtaz 等[13]借助 CRISPR/Cas 基因组编辑技术，对预先保存在体外的长链 DNA 片段实施了碱基的精准替换，并创新性地引入了一种模块化信息存储策略，使存储于 DNA 序列中的信息顺序得以重新配置，极大地增强了 DNA 作为存储介质的灵活性与可编辑性。

然而，这些信息编辑方法在适应数据变动的灵活性、版本控制的严格性和成本效率等方面仍存在明显局限，未充分满足现代数据管理对即时性和经济性的双重要求。尤其是在多用户协同作业的背景下，不同用户对同一数据集的不同编辑需求可能导致版本冲突与数据安全性问题加剧，因此有必要建立更严谨的管理和安全保障机制。总而言之，DNA 存储技术在信息编辑方面迫切需求一个效率高、成本效益高且能够支持复杂多方编辑场景的解决方案，这是当前研究的焦点与挑战所在。

增量存储是一种当数据集非常大或者更新频繁时可以显著提高效率并减少资源消耗的存储方案，在各领域研究中广泛应用。Hinkel 等[14]关注模型分析的隐式增量方法，研究开发了一个可扩展的隐式增量计算系统，并验证了其在模型查询中的适用性，在相关测试中查询增量化后的速度提高了几个数量级。Wang 等[15]探讨了一种增量动态模型下的复杂连续行动迭代困境，通过提出增量更新方法，避免了原有玩家状态的重复刷新，提高了计算效率。Lei 等[16]将增量存储用于移动系统更新，提出了一种一致且高效的增量更新 (consistent and efficient incremental update，CEIU) 机制。CEIU 通过重用旧镜像块索引而非复制块本身，降低了内存消耗和文件访问次数，并确保在系统崩溃或电源中断后，数据的一致性和完整性得到维护，同时优化了空间使用率和更新时间。这些技术体现了增量存储的优势和可用性，因此将增量存储用于 DNA 存储领域，针对当前 DNA 合成和测序技术限制下的信息编辑需求，分别构建不同的存储数据分区方案和索引编码方案，提出高效的索引链接和信息编辑方法，以有效解决当前 DNA 信息编辑存在的难题。

针对当前 DNA 存储领域信息编辑研究匮乏、多方信息编辑支持不足、效率低和成本高等问题，本文提出面向 DNA 安全存储的信息增量管理方法，通过一种面向多方编辑的混合型加密方法，构建 DNA 增量存储模型，可安全、灵活地编辑和管理 DNA 存储信息。实验结果显示，DNA 增量存储模型提供了一种便捷、高效、更低成本的信息编辑方法，其仅存储待进行的编辑操作，不实际修改 DNA 分子，因而成本更低、效率更高。同时，这一方法将存储信息更新这一步延后，在读取时才真正进行更新，使复杂的数据处理操作与 DNA 存储技术解耦，有效兼容已有 DNA 存储方法。此外，基于混合型加密方法，使不同用户保存的数据隔离，从而使不同用户的数据操作隔离，实现可靠的多方信息编辑。混合型加密方法满足多用户环境下的安全需求。

2 相关工作

数据加密算法可分为对称加密和非对称加密算法两类，二者各有优劣。

袁涛等[17]提出一种基于混沌系统和喷泉码的加密编码方法——DNA 混沌喷泉编码 (DNA chaos-fountain encoding，DCFE)，利用超混沌系统加密原理在 DNA 喷泉码编码过程中进行数据加密，保证信息安全，同时实现高信息密度和具备良好纠错能力的 DNA 编码，并可满足多种生物约束条件，用于不同数据类型和规模，其整体架构如图 1 所示。

DCFE 方法的加密编码流程主要如下：

(1) 通过密钥进行超混沌伪随机序列生成，

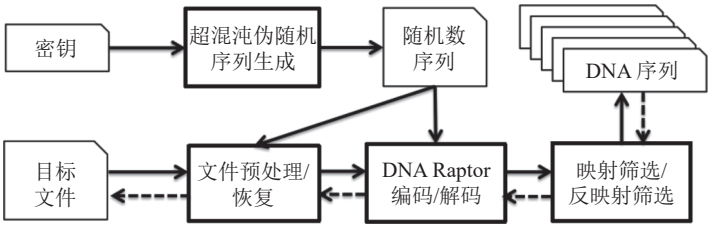


图 1　DNA 混沌喷泉编码方法

Fig. 1　The DNA chaos-fountain encoding method

得到用于加密的随机数；

(2) 将目标文件进行预处理并初次加密；

(3) 进行 DNA Raptor 编码，并再次加密；

(4) 对编码结果进行映射筛选，完成数据到 DNA 序列的转化。

其解密解码流程与加密编码流程一一对应，使用相同的密钥生成相同的随机数进行解密，因而是一种对称加密方法。

RSA (Rivest-Shamir-Adleman) 算法[18]通过大整数的因子分解困难性保证加密对象的安全性，是一种非常经典和广泛应用的非对称加密算法。在 RSA 算法中，密钥生成是关键步骤。首先，选择两个大质数 p 和 q，计算它们的乘积 n；其次，计算欧拉函数 $\phi(n)=(p-1)\cdot(q-1)$；再次，选择一个公共指数 e，要求 $1<e<\phi(n)$，且 e 与 $\phi(n)$ 互质；最后，计算私有指数 d，满足 $d\equiv e^{-1}(\bmod \phi(n))$。这样就得到了公钥 (n,e) 和私钥 (n,d)。

加密过程中，选择明文 P，计算密文 $C\equiv M^e(\bmod n)$；解密过程中，使用私钥计算 $M\equiv C^d(\bmod n)$。RSA 算法的安全性基于在已知 n 的情况下，找到其质因数 p 和 q 的困难性。

3　混合型加密编码方法

在多方编辑场景，多个文件在不同用户访问下的安全性成为问题，单个文件使用对称加密的效率较高，但密钥容易泄露；非对称加密进行文件加解密效率低，但可以保障密钥安全。

针对对称加密算法密钥管理困难和非对称加密算法加解密效率低下的问题，本文采用对称加密保护信息，非对称加密机制进行身份管理，进而实现一种混合型加密编码方法，完成 DNA 存储中多方用户场景下的信息安全存储与身份管理。

具体地，对称加密部分采用 DCFE 方法，非对称加密部分采用 RSA 算法。同时，二进制数据到 DNA 序列需要编码，DCFE 方法可在加密的同时完成该编码，而 RSA 算法主要针对 DCFE 方法的密钥，无须再次编码。总体上，通过 DCFE 方法和 RSA 算法的结合实现一种适用于 DNA 存储的混合型加密编码方法。

混合型加密方法的一次通信过程如图 2 所示。参与方 A 首先生成一个对称加密密钥 K，使用该密钥 K 加密明文信息 P，得到密文 $C=E_K(P)$。其中 E 为加密函数。同时，参与方 A 使用参与方 B 的公开密钥 PK 加密密钥 K，得到存储密钥信息的密文 $C_K=E_{PK}(K)$。参与方 A 将密文 C 和密钥密文 C_K 一起发送给参与方 B。参与方 B 先使用私钥 SK 解密密钥密文 C_K，得到对称加密算法的解密密钥 $K=D_{SK}(C_K)$。其中 D 为解密函数，使用密钥 K 解密密文 C，得到参与方 A 发送的信息明文 $P=D_K(C)$。

4　DNA 增量存储模型

增量存储指在已有的存储基础上，只保存数

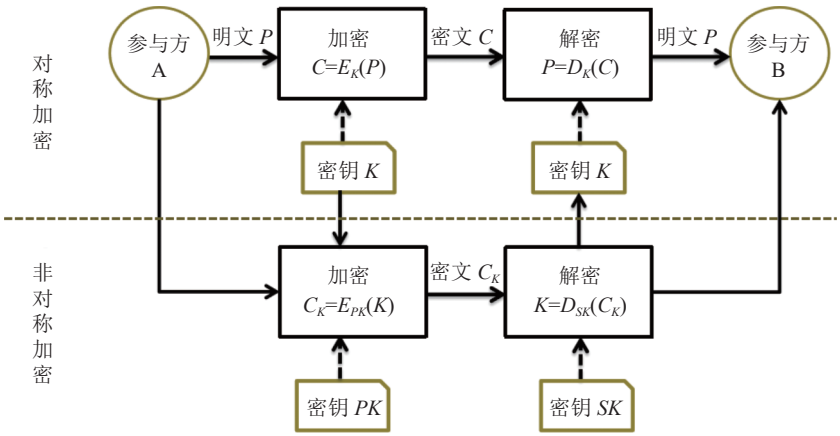


图 2 混合型加密方法

Fig. 2 The hybrid encryption method

据的变化或更新部分，而不将更新后的整个数据集再次存储。将增量存储用于 DNA 存储过程中，不仅可解决 DNA 分子修改信息效率低下且成本高昂的问题，还可避免未变化数据的重复存储。为实现面向 DNA 存储的增量存储，本文构建了 DNA 增量存储模型，即将存储数据的更新操作作为增量信息记录，编码为新的 DNA 序列，在读取时再根据增量信息进行数据更新，避免直接对已有 DNA 链进行修改。其中，文件和对应增量信息通过混合型加密编码方法完成加密和编码，实现由二进制数据到 DNA 序列的安全转化。

DNA 增量存储模型面向多个文件和用户，构造一个按存储数据哈希值寻址的文件索引树结构，同时记录所有增量信息 (即数据的增、删、改等操作)，以实现信息更新，并按照哈希树组织内容版本管理 (不同查询起点代表不同版本)。DNA 增量存储模型通过存储用户身份信息 (通过 RSA 加密的数据表示身份)，实现用户身份管理。DNA 增量存储模型基于存储的身份信息实现信息读取的权限控制 (只有使用正确的公钥解密，才能访问数据)，进而实现信息的安全共享，同时通过追溯哈希树的哈希索引链接文件信息和增量信息，随后通过文件索引重组文件，通过增量信息还原 DNA 存储信息的最终数据状态，以实现完整的信息读取，最终通过增量存储模型达到无须基因编辑的 DNA 存储信息管理目标。

从原理上，本文提出的增量存储框架提供了一种便捷、高效且相对成本更低的信息编辑方法，其仅存储待进行的编辑操作，不实际修改 DNA 分子，因而成本更低、效率更高。同时，这一方法将存储信息更新这一步延后，在读取时才真正进行更新，使复杂的数据处理操作与 DNA 存储技术解耦，有效兼容已有 DNA 存储方法。此外，通过混合加密，使不同用户保存的数据可以进行隔离，从而进一步使不同用户的数据操作隔离，实现可靠的多方信息编辑。

4.1 数据结构

存储数据类型分为原始文件、增量信息和文件索引 3 部分，通过文件索引查找原始文件和增量信息，结合原始文件和增量信息得到更新后的数据。3 种存储数据的具体介绍如下：

(1) 原始文件为待存储数据的最初副本。在当前 DNA 存储中，一条 DNA 链的长度一般在 100～200 nt (nucleotide)，对应 25～50 bytes (理论最大值) 的数据，文件通常切分为多个小数据块进行存储。

(2) 增量信息对应指定文件的更新内容，将

文件的信息编辑操作 (即如何更新文件) 转化为特定格式的增量信息进行存储。由于需要记录每次操作的先后顺序，因此采用链表方式进行管理，从链表中的最新记录依次追溯前一条记录。

(3) 文件索引将原始文件与其对应的增量信息相关联，通过索引定位最新增量信息记录，并对多个文件进行管理，同时在索引中保存身份信息，以提供不同的访问权限。

3 种存储数据的结构如图 3 所示，具体说明如下：

(1) 哈希值：基于 DNA 存储的特性，索引一旦创建不易修改，因此为唯一标识索引，将基于存储内容计算得到的哈希值 (初始为 64 bit，编码后为 40 nt) 作为索引地址，不同 DNA 序列的查询通过哈希值进行。原始文件和增量信息虽然因一条 DNA 的长度限制而必须进行数据分块处理，但一份文件或增量信息的所有分块对应同一个哈希值，而文件索引不需要分块且其哈希值只基于链接值计算，即一条文件索引对应一个哈希值。所有哈希值独立编码，过程如下：首先，随机生成 1 000 行 64 bit 的随机数 (一旦生成不再改变，所有哈希值共用)，并保存；其次，对每个初始二进制哈希值，不断以卢比变换码 (Luby transform code，LT)[19]编码方式生成一个 64 bit 的编码结果，与该二进制哈希值异或，并在末端添加对应 LT 编码序号 (16 bit)；最后，映射为 DNA 序列 (四进制碱基) 进行约束筛选，直到通过筛选。通过筛选的 40 nt DNA 序列即为编码后的哈希值。此外，当发生哈希冲突时，判定为未通过约束筛选，重新进行映射，即二进制的哈希值不变，但对应的四进制碱基重新生成，以确保全局哈希值唯一。

(2) 密钥值：用于对索引 (除哈希值外) 进行加密和编码，并实现用户身份验证。密钥值随机生成，但其尾部的 16 bit 对应 DCFE 方法序号，使用该序号对索引 (除哈希值外) 进行与哈希值编码方式相同的 LT 编码，得到 DNA 序列。同时，链接值会通过该密钥值进行加密，而密钥值会通过索引固定密钥进行异或加密。索引固定密钥用于对所有密钥值进行加密，该密钥值自动生成，并通过 RSA 算法进行加密后保存；不同用户读取索引时直接使用未加密的索引固定密钥或通过对应 RSA 算法解密得到索引固定密钥，再对密钥值解密，最后用解密后的密钥值解密链接值，进行索引链接。

(3) 链接类别：表示索引指向对象的数据类型，包括文件索引、增量信息、原始文件和文件名 4 类数据 (文件名用于表示哈希值和文件的对

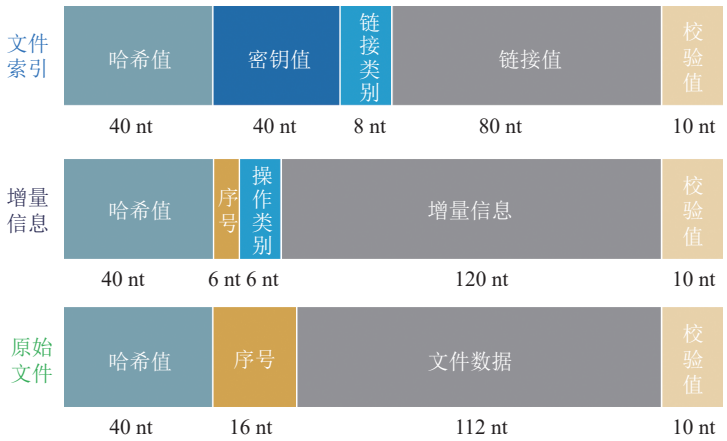


图 3　3 种存储数据结构

Fig. 3　Three types of storage data structures

应关系)。

(4) 链接值：包含 2 个 40 nt 的哈希值，代表该索引链接的两个对象。其中，每个原始文件有一条特殊的对应索引，其链接值为该文件的哈希值和文件名，表明哈希值对应的具体文件，便于通过文件名进行查询。

(5) 校验值：用于对数据进行校验，并进行碱基错误纠正，采用 RS 码。

(6) 序号：是原始文件和增量信息使用 DCFE 方法加密编码时的序号。

(7) 操作类别：用于表示增量信息的类型 (增、删、改)。用户对已存储文件的信息进行编辑操作会转化为固定格式的增量信息 (用户只能对存储数据进行增加、删除和修改，并按固定格式提供对应信息)。

(8) 增量信息：包含增、删、改 3 类，每类信息按照位置 (待编辑的数据从文件的多少字节开始)、大小 (需要改变的数据大小) 和内容 (重新编辑后的数据) 的顺序进行存储，以特定字符间隔，并使用 DCFE 方法进行加密和编码。多条增量信息按先后顺序以链表形式连接，由最新更新内容依次指向前一次更新内容；当读取时，根据增量信息内容以反向顺序 (由早到晚) 执行更新操作，即实际的更新在读取文件时进行。存储数据的每次更新内容都会被存储，以链表形式连接，使文件的每一次变化过程得以记录。

(9) 文件数据：原始文件数据，使用 DCFE 方法进行加密和编码后得到。

4.2 整体模型架构

DNA 增量存储模型主要用于信息编辑环境，结合混合型加密方法保证多方用户下的信息安全。原始文件、增量信息和文件索引等 3 种数据的具体加密过程如下：

(1) 对于原始文件和增量信息，首先，用户需要设置 DCFE 方法加密的密钥 K；其次，使用该方法进行加密编码；再次，生成 RSA 算法公钥 PK 和私钥 SK；最后，通过公钥 PK 加密密钥 K 得到密文 C_K。密文 C_K 和公钥 PK 是多方编辑时的身份验证信息，任何参与用户只能通过该信息进行索引查询，从而访问原始文件和增量信息。

(2) 对于文件索引，随机生成一个 40 nt 的临时密钥，使用其对链接值进行异或加密，再通过私钥 SK 加密临时密钥得到密钥值。文件索引仅通过公钥 PK 即可通过身份验证，使无密文 C_K 而有公钥 PK 的用户仅可查询文件存在与否，获取所有存储文件的名称。RSA 算法用于密钥 K 和索引的密钥值加密。

DNA 增量存储模型的整体加密流程如图 4 所示，其除索引部分外即为混合型加密编码方法。

基于数据最终用途，将 3 种存储数据分别存

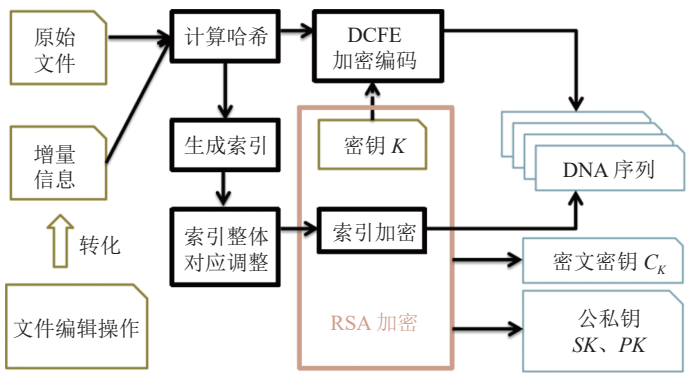


图 4 整体加密流程

Fig. 4 The overall encryption process

在文件区和索引区两部分：文件区包含原始文件、增量信息链和一条文件索引 (记作增量索引，与索引区文件索引做区分)，新增量信息加入链末端，通过一条文件索引记录对应原始文件和增量信息链尾地址，增量索引随每次增量信息更新而更新；索引区包含所有文件区索引，最终指向文件区增量索引。

原始文件和增量信息总是相互对应，二者的存储与读取操作同步进行，所以构建文件区进行整合。文件区结构如图 5 所示。当文件有更新时，将更新操作作为增量信息进行存储，生成对应的增量信息节点，使其指向原有的增量节点链表头，生成新的增量索引 (文件区根索引) 节点，使其分别指向文件和新增量信息节点，将该增量索引节点的哈希值作为最新的标识文件区的哈希值。

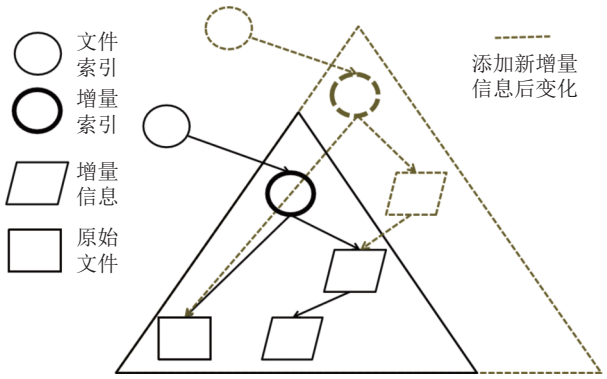


图 5　文件区结构

Fig. 5　The file area structure

4.3　索引方案

DNA 增量存储模型的重点在于索引的构建。本文提出 3 种索引方案，对应不同的存储状况，具体如下：

方案 A 采用二叉树方式构建索引，叶子节点指向文件区对应的唯一增量索引，根节点作为根索引，如图 6 所示；

方案 B 直接使用根索引指向文件区对应的唯一增量索引，根节点同时也是叶子节点，如图 7 所示；

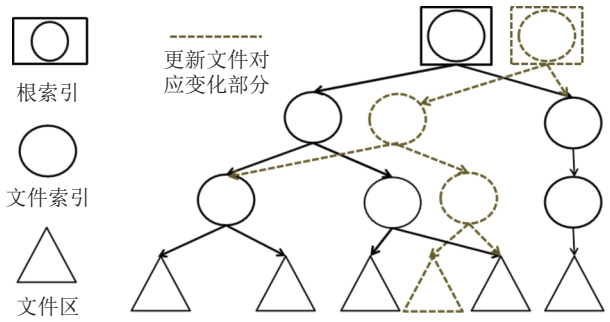


图 6　索引方案 A

Fig. 6　The index scheme A

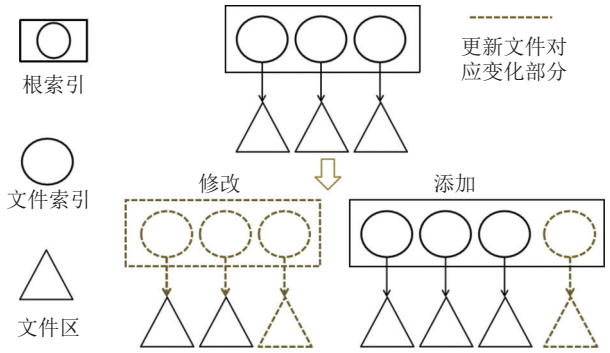


图 7　索引方案 B

Fig. 7　The index scheme B

方案 C 采用方案 B 结合增量信息记录的方式构建索引，即对原始文件构建初始索引，根据增量信息添加新索引，不使用文件区，其初始状态与索引 B 相同，但更新后总是有且仅有一个根索引指向初始索引节点和增量节点，如图 8 所示。

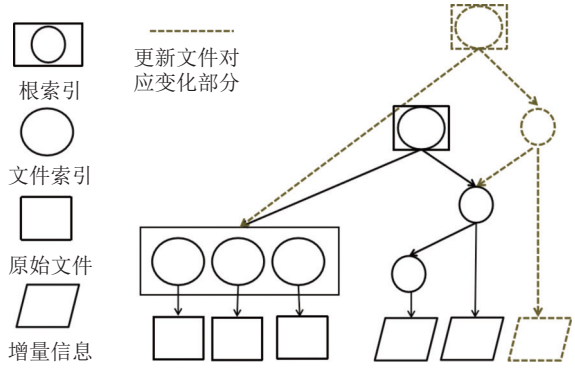


图 8　索引方案 C

Fig. 8　The index scheme C

如图 6 所示，查询时，索引方案 A 从根索引节点开始进行二叉树遍历，找到符合查询目标的叶子节点，根据叶子节点记录的文件哈希值在文件区中获取文件；更新时，获取更新后文件对应的新文件哈希值，生成对应的新叶子节点，然后依次自底向上更新其对应的父节点，将更新后的根节点哈希值作为最近的查询起点。

如图 7 所示，查询时，索引方案 B 直接遍历所有的根索引节点，找到符合查询目标的节点，根据其记录的文件哈希值在文件区中获取文件；更新时，获取更新后文件对应的新文件哈希值，重新生成所有的根索引节点，所有新根索引哈希值相同且作为最近的查询起点。

如图 8 所示，查询时，索引方案 C 如果还未更新，则与索引方案 B 查询方式相同，如果已经更新，则根据唯一根索引找到增量节点和初始索引节点，分别进行遍历，找到符合查询目标的节点，获取增量节点中记录的增量信息，根据初始索引记录的文件哈希值在文件区中获取文件；更新时，将更新操作作为增量信息进行存储，生成对应的增量节点，使其指向原有的增量节点链表头，然后生成新的根索引节点，使其分别指向初始索引节点和新增量节点，将该根索引节点哈希值作为最近的查询起点。

DNA 增量存储模型读取数据时，首先找到根索引，根据根索引找到对应子索引和增量信息，不断追溯哈希树的哈希索引链接文件信息和增量信息；其次，重组原始文件；最后，通过增量信息还原数据更新操作，获取存储数据更新后的最终状态，达到无须基因编辑的 DNA 存储信息管理目标。其中，根据索引查询时，需根据密钥值验证身份信息，获取真实密钥值，解密链接值；根据链接值找到对应二叉树子节点，对索引、增量信息、原始文件分别进行解码，进行相应操作；根据增量信息还原数据更新操作时，按增量信息链先后顺序将增量信息压入信息栈，得到原始文件后，如果信息栈不为空，则出栈；根据增量信息还原文件更新操作。由于旧的索引不会被删除，因此选择不同的查询起点 (根据根索引哈希值开始查询，不同根索引代表不同版本)，可获取不同版本的存储信息，实现多版本内容管理。

5 实验结果分析

本文构建的 DNA 增量存储模型的主要功能是引入信息增量管理机制，实现 DNA 存储中高效而低成本的信息编辑。不同信息编辑方法的对比如表 1 所示。此外，DNA 增量存储模型的重点在于索引，针对其使用的不同索引方案效果，

表 1 不同信息编辑方法对比

Table 1 The comparison of different information editing methods

方法	适用数据类型	数据加密	信息编辑技术	内容版本控制	相对成本	可编辑数据量
Tabatabaei Yazdi 等[8]	通用文件	否	基因组编辑技术、重叠延伸 PCR 技术	无	高	少量 DNA 分子
Appuswamy 等[9]	数据库表	否	重叠延伸 PCR 技术	无	低	长 DNA 片段
Lee 等[10]	文本文件	否	重叠延伸 PCR 技术	无	低	长 DNA 片段
Lin 等[11]	通用文件	否	核酸分子杂交技术	无	高	文件整体
Liu 等[12]	通用文件	否	基因组编辑技术	无	高	长 DNA 片段
Sadremomtaz 等[13]	通用文件	否	基因组编辑技术	无	高	长 DNA 片段
本文	通用文件	是	DNA 增量存储模型	记录每次修改信息	低	任意

本文从查询开销和更新开销方面进行评估，讨论其在不同场景下的适用范围。同时，本文对其在多方编辑环境下的信息安全进行理论分析，证明其安全性。

由表 1 可知，DNA 增量存储模型通过存储增量信息的方式完成信息编辑，适用于任意类型文件，并且记录每次修改的信息，以实现内容版本控制 (可回溯文件历史版本)，这是其他信息编辑方法所不具备的功能。同时，存储的信息通过混合型加密方法进行加密，其安全性得到保障。DNA 增量存储模型的成本在于额外存储的增量信息和索引，比复杂的基因组编辑技术和核酸分子杂交技术的成本更低，且效率更高，并能实现任意数据量的信息编辑。

在当前 DNA 存储中，信息编辑研究集中于生化技术的应用，从存储的物理层面考虑，忽略了存储的逻辑层面。本文将增量存储这一逻辑存储技术引入 DNA 存储领域，从新的角度进行 DNA 存储中的信息编辑，拓展了其使用数据类型和数据量，实现了内容版本控制，使 DNA 存储更符合当前数据存储的要求。

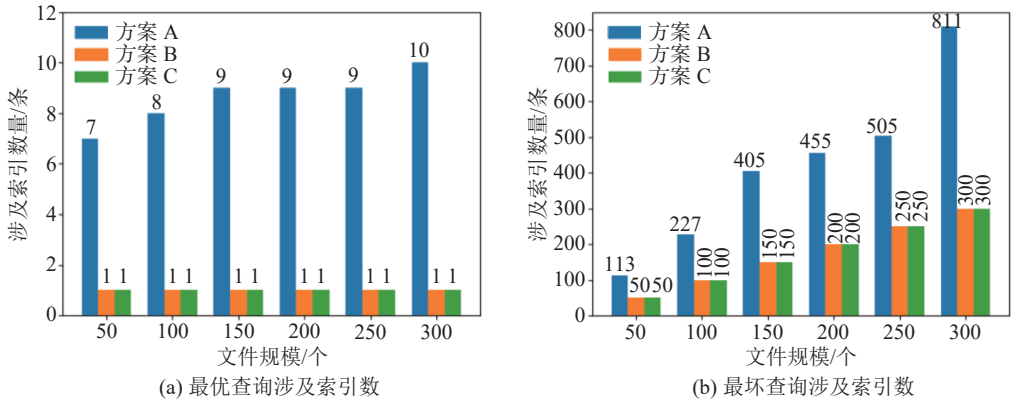
5.1　DNA 增量存储模型索引效果分析

本文评估了 3 种索引方案的查询开销和更新开销。

在未进行任何增量操作 (存储一次增量信息)，仅存储原始文件数据的条件下，不同规模的索引 (每个文件对应一条索引) 进行一次查询，最优情况下涉及的索引数量 (理论找到查询目标需要访问的最少索引数量) 和最坏情况下涉及的索引数量 (理论找到查询目标需要访问的最多索引数量) 如图 9 所示。可以看出，方案 A 涉及的索引数最多，这是因为其二叉树结构带来了路径开销，而方案 B 和 C 没有这部分路径开销。在最优情况下，方案 B 和 C 查询涉及的索引数始终不变，维持在最少的 1 条。在最坏情况下，所有查询涉及的索引数与规模近似正相关，而方案 A 的查询开销是方案 B 和 C 的 2 倍。

在一半原始文件进行增量操作后 (一半文件有至少一条对应的增量信息)，不同规模的索引进行一次查询，最优情况下和最坏情况下涉及的索引数量如图 10 所示。其中，方案 C 的查询开销受到显著影响，尤其是最优情况下，一次查询涉及的索引数量远大于方案 A 和 B。

进行一次增量操作，除了需要存储对应的增量信息外，还需要对索引进行更新。本文在不同规模的索引条件下进行一次增量操作，分析在不同规模的已有索引数量条件下，结果如图 11 所示。可以看出，每次增量操作下方案 B 的更新开销最大，因为需要更新所有索引；而方案 A 的开销相对较小，且随规模增大开销缓慢增加；方案



(a) 最优查询涉及索引数　　(b) 最坏查询涉及索引数

图 9　不同规模原始文件查询开销

Fig. 9　The cost of querying raw files of different scales

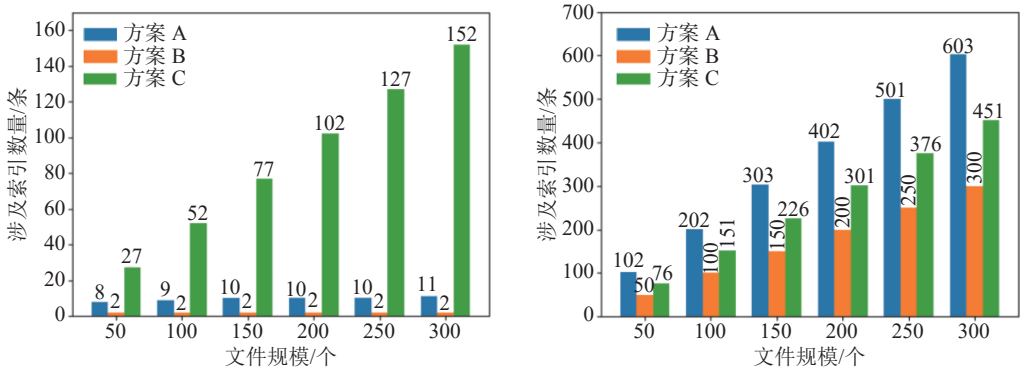


图 10　50% 原始文件增量操作后的查询开销

Fig. 10　The cost of querying after 50% incremental operation of raw files

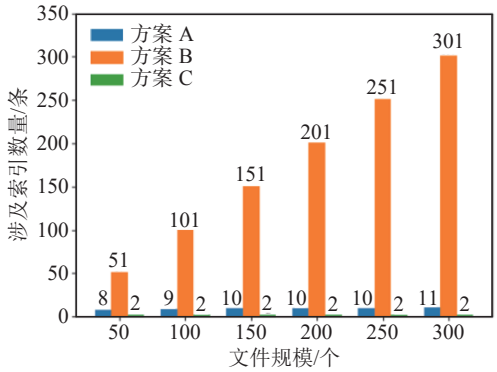


图 11　一次增量操作的索引更新开销

Fig. 11　The index update cost for one incremental operation

C 的更新开销恒定，且始终维持在最低水平。

相同规模下，索引方案 C 的查询开销在增量操作后产生变化，因此，本文选择在 100 条索引的固定规模下，衡量不同方案的增量操作次数与查询开销的关系，结果如图 12 所示。可以看出，方案 A 和 B 的查询开销与增量操作次数无关，而方案 C 的查询开销随增量操作次数的增加而线性增加。方案 C 的初始查询开销虽然较小，但达到一定的更新次数后，查询开销将变得极大。

综合考虑 3 种索引方案，方案 A 的查询开销较大，但更新开销小；方案 B 的查询开销极小，但更新开销极大；方案 C 的初始查询开销极小，更新开销也小，但查询开销会随更新次数的增多而线性增大。

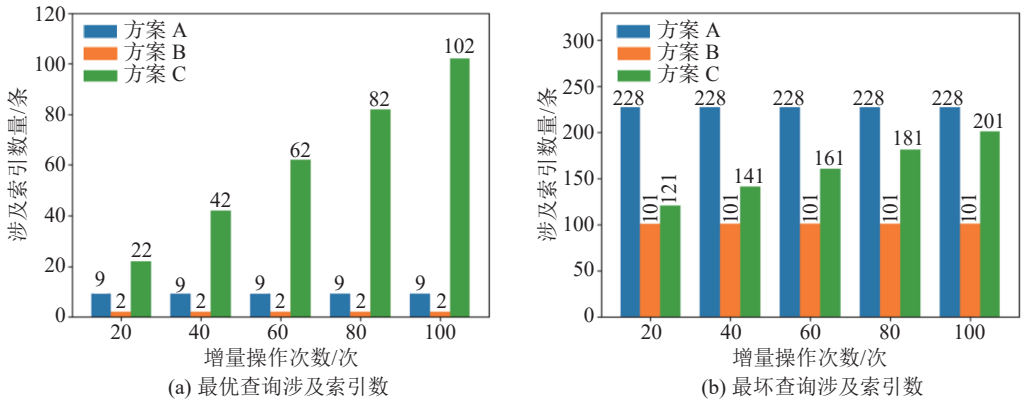


图 12　不同方案的增量操作次数与查询开销的关系

Fig. 12　The relationship between incremental operation times and query cost in different schemes

当存储的是冷数据，更新次数少，测序成本高而合成成本低时，推荐方案 B；当存储的是冷数据，更新次数少，测序成本低而合成成本高时，推荐方案 C；当已更新多次，并已存储大量增量信息时，推荐方案 A，此时可重新构建全部索引。

5.2　多方编辑安全性分析

多方编辑安全性的重点在于在多方协同环境下对同一数据进行编辑时的安全保障，不仅需要考虑数据的安全性，还需要考虑加密密钥的安全性。针对单用户存储数据的安全攻击主要从明文、密文和密钥 3 方面进行，具体攻击方式如下：

(1) 暴力攻击，暴力搜索所有可能的密钥；

(2) 统计攻击，根据密文的统计特征分析明文和密钥；

(3) 差分攻击，通过对比不同明文对的差异和相应密文对的差异，推断加密算法的内部结构或密钥。

此外，多用户通信时存在直接暴露密钥的安全问题，可能遭遇中间人攻击，即攻击者位于通信双方之间，拦截和篡改信息以欺骗或替换密钥。

本文首先利用混合型加密保证多方编辑环境下的信息安全；其次，利用对称加密方式对原始数据进行加密，保证数据安全性；最后，利用非对称加密方式对对称加密密钥进行加密，保证密钥安全性。混合型加密方式的安全性实际基于所使用的对称加密算法的安全性和非对称加密算法的安全性。

本文使用 DCFE 方法作为对称加密算法对存储数据加密和编码，其安全性已进行过验证分析，可以有效抵抗暴力攻击、统计攻击和差分攻击[17]。

本文采用 RSA 算法作为非对称加密算法对 DCFE 方法的密钥进行加密，RSA 密钥长度为 2 048 位。RSA 算法的安全性依赖于大数因子分解的困难性，至今还没有一个已知的多项式时间算法能在可接受的时间内实现大数因子分解。大数因子分解的困难性与密钥长度相关，512 位和 1 024 位长度的密钥由于计算技术的发展已经不再安全，有概率被暴力攻击破解。而 2 048 位密钥目前被认为是一种较为安全的选项，众多证书颁发机构使用 2 048 位密钥的 RSA 算法进行非对称加密。本文选取 2 048 位密钥的 RSA 算法进行密钥加密，通过公私钥分离，抵抗中间人攻击，保证密钥安全性。

6　结　论

在 DNA 存储技术的进阶应用中，信息编辑可构成一个核心环节，涉及对已转化成 DNA 序列的数据信息进行修正与更新处理，并确保其能被重新存档。在当前的 DNA 存储领域，信息编辑方向的研究较少，存在多方编辑支持不足、效率低和成本高等问题。本文提出了面向 DNA 安全存储的信息增量管理方法，首先通过面向多方编辑的混合型加密方法保证多方编辑环境下的信息安全；其次通过构建 DNA 增量存储模型，记录所有增量信息，实现 DNA 存储信息编辑；最后，设计多种增量存储索引方案，解决增量信息在不同场景下存储的适配性问题，达到无须基因编辑的 DNA 存储信息管理目标，降低 DNA 存储数据更新开销。本文通过实验验证了 DNA 增量存储模型的索引效果，并分析了多方编辑的安全性。然而，本文提出的方法尚有不足，后续工作可进行探索改进，为 DNA 存储技术的推广应用充实基础，具体如下：在加密方面，可结合更新的加密算法进行改进，如最新的抗量子攻击算法等；在数据处理方面，可根据不同数据类型结合压缩算法进行压缩，进一步降低存储空间；在编码方面，可尝试使用最新的 DNA 编码方法进行优化；在信息编辑方面，可结合已有基因编

辑技术和增量存储方案，实现更好的信息编辑方法。

参 考 文 献

[1] Bonnet J, Colotte M, Coudy D, et al. Chain and conformation stability of solid-state DNA: implications for room temperature storage [J]. Nucleic Acids Research, 2010, 38(5): 1531-1546.

[2] Church GM, Gao Y, Kosuri S. Next-generation digital information storage in DNA [J]. Science, 2012, 337(6102): 1628.

[3] Goldman N, Bertone P, Chen SY, et al. Towards practical, high-capacity, low-maintenance information storage in synthesized DNA [J]. Nature, 2013, 494(7435): 77-80.

[4] 昝乡镇, 姚翔宇, 许鹏, 等. DNA 存储文件系统研究进展 [J]. 电子与信息学报, 2023, 45(6): 1911-1920.
Zan XZ, Yao XY, Xu P, et al. A survey on file architecture in DNA storage [J]. Journal of Electronics and Information Technology, 2023, 45(6): 1911-1920.

[5] Southern EM. Detection of specific sequences among DNA fragments separated by gel electrophoresis [J]. Journal of Molecular Biology, 1975, 98(3): 503-517.

[6] Makarova KS, Grishin NV, Shabalina SA, et al. A putative RNA-interference-based immune system in prokaryotes: computational analysis of the predicted enzymatic machinery, functional analogies with eukaryotic RNAi, and hypothetical mechanisms of action [J]. Biology Direct, 2006, 1: 1-26.

[7] Bryksin AV, Matsumura I. Overlap extension PCR cloning: a simple and reliable way to create recombinant plasmids [J]. BioTechniques, 2010, 48(6): 463-465.

[8] Tabatabaei Yazdi SMH, Yuan YB, Ma J, et al. A rewritable, random-access DNA-based storage system [J]. Scientific Reports, 2015, 5: 14138.

[9] Appuswamy R, Brigand KL, Barbry P, et al. OligoArchive: using DNA in the DBMS storage hierarchy [C] // Proceedings of the Conference on Innovative Data Systems Research, 2019: 98.

[10] Lee UJ, Hwang S, Kim KE, et al. DNA data storage in Perl [J]. Biotechnology and Bioprocess Engineering, 2020, 25: 607-615.

[11] Lin KN, Volkel K, Tuck JM, et al. Dynamic and scalable DNA-based information storage [J]. Nature Communications, 2020, 11(1): 2981.

[12] Liu YY, Ren YB, Li JJ, et al. *In vivo* processing of digital information molecularly with targeted specificity and robust reliability [J]. Science Advances, 2022, 8(31): eabo7415.

[13] Sadremomtaz A, Glass RF, Guerrero JE, et al. Digital data storage on DNA tape using CRISPR base editors [J]. Nature Communications, 2023, 14(1): 6472.

[14] Hinkel G, Heinrich R, Reussner R. An extensible approach to implicit incremental model analyses [J]. Software and Systems Modeling, 2019, 18: 3151-3187.

[15] Wang Z, Li HJ, Jin XY, et al. Complex continuous action iterated dilemma with incremental dynamic model [J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2024, 54(4): 2309-2319.

[16] Lei R, Chen X, Liu D, et al. CEIU: consistent and efficient incremental update mechanism for mobile systems on flash storage [J]. Journal of Systems Architecture, 2024, 152: 103151.

[17] 袁涛, 曲强, 姜青山. 基于混沌系统和喷泉码的 DNA 加密编码方法 [J]. 集成技术, 2024, 13(3): 4-24.
Yuan T, Qu Q, Jiang QS. An encrypted DNA encoding method based on chaotic system and fountain code [J]. Journal of Integration Technology, 2024, 13(3): 4-24.

[18] Rivest R L, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems [J]. Communications of the ACM, 1978, 21(2): 120-126

[19] Luby M. LT codes [C] // Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002: 271-280.