Briefings in Bioinformatics, 2022, pp. 1–12

doi: DOI HERE

Advance Access Publication Date: Day Month Year

Paper

PAPER

Explorer: Efficient DNA Coding by De Bruijn Graph towards Arbitrary Local and Global Biochemical Constraints

Chang Dou,[1,†] Yijie Yang,[1,†] Fei Zhu,[1] BingZhi Li[2,3,*] and Yuping Duan[1,*]

[1]Center for Applied Mathematics, Tianjin University, Tianjin, 300072, China, [2]Frontiers Science Center for Synthetic Biology and Key Laboratory of Systems Bioengineering (Ministry of Education), Tianjin University, Tianjin, 300072, China and [3]School of Chemical Engineering and Technology, Tianjin University, Tianjin, 300072, China

[†]First Author and Second Author contribute equally to this work.[*]Corresponding author. yuping.duan@tju.edu.cn,bzli@tju.edu.cn

FOR PUBLISHER ONLY Received on Date Month Year; revised on Date Month Year; accepted on Date Month Year

Abstract

With the exponential growth of digital data, there is a pressing need for innovative storage media and techniques. DNA molecules, due to their stability, storage capacity, and density, offer a promising solution for information storage. However, DNA storage also faces numerous challenges, such as complex biochemical constraints and encoding efficiency. This paper presents **Explorer**, a high-efficiency DNA coding algorithm based on the De Bruijn graph, which leverages its capability to characterize local sequences. **Explorer** enables coding under various biochemical constraints, such as homopolymers, GC content, and undesired motifs. This paper also introduces **Codeformer**, a fast decoding algorithm based on the transformer architecture, to further enhance decoding efficiency. Numerical experiments indicate that, compared to other advanced algorithms, **Explorer** not only achieves stable encoding and decoding under various biochemical constraints but also increases the encoding efficiency and bit rate by more than 10%. Additionally, **Codeformer** demonstrates the ability to efficiently decode large quantities of DNA sequences. Under different parameter settings, its decoding efficiency exceeds that of traditional algorithms by more than two-fold. When **Codeformer** is combined with RS code, its decoding accuracy exceeds 99%, making it a good choice for high-speed decoding applications. These advancements are expected to contribute to the development of DNA-based data storage systems and the broader exploration of DNA as a novel information storage medium.

Key words: DNA storage, biochemical constraint, transformer, De Bruijn graph

Introduction

Due to the exponential growth of digital information, it is necessary to develop new solutions to meet the growing storage needs [1, 2, 3, 4]. Traditional storage devices are facing limitations in capacity and longevity, proving them insufficient for the increasing volume of the generated data. DNA is well-known as one of the most stable biomolecules, which can preserve information for more than centuries with low energy costs. In addition to that, the storage capacity of DNA molecules is about 4.2×10^{21} bits per gram, which is 420 billion times that of traditional storage media [5]. Thus, DNA medium becomes a new data storage technology, which can achieve digital data storage by encoding and decoding the synthesized DNA. DNA storage involves five key components: encoding, synthesis, storage, sequencing, and decoding (see Fig. 1A), where the coding algorithm serves as the intermediary for the reciprocal conversion between digital information and DNA molecules. Thus, the DNA coding algorithm is the most basic and critical step for DNA storage.

Due to the limitation of biochemical technology during the DNA storage process, the DNA sequence must conform to specific biochemical constraints to minimize the error rate in the biochemical process [6]. A key objective in designing encoding algorithms is to ensure that the encoded DNA strings adhere to the biochemical constraints imposed by DNA storage synthesis and sequencing technologies. Two primary biochemical constraints widely employed are GC content balance and homopolymer length. Deviating from the optimal GC content range, whether higher or lower, may result in reduced sequencing coverage, emphasizing the need to constrain GC content within a specific range [7]. The lengths of homopolymer need to be constrained to a specific value to minimize error rates during the synthesis and sequencing [3]. Several existing DNA encoding algorithms have considered the aforementioned two biochemical constraints during the encoding process [5, 8, 9, 10, 11].

While an increasing number of encoding algorithms have been developed to address the aforementioned biochemical constraints, basic constraints alone are no longer adequate

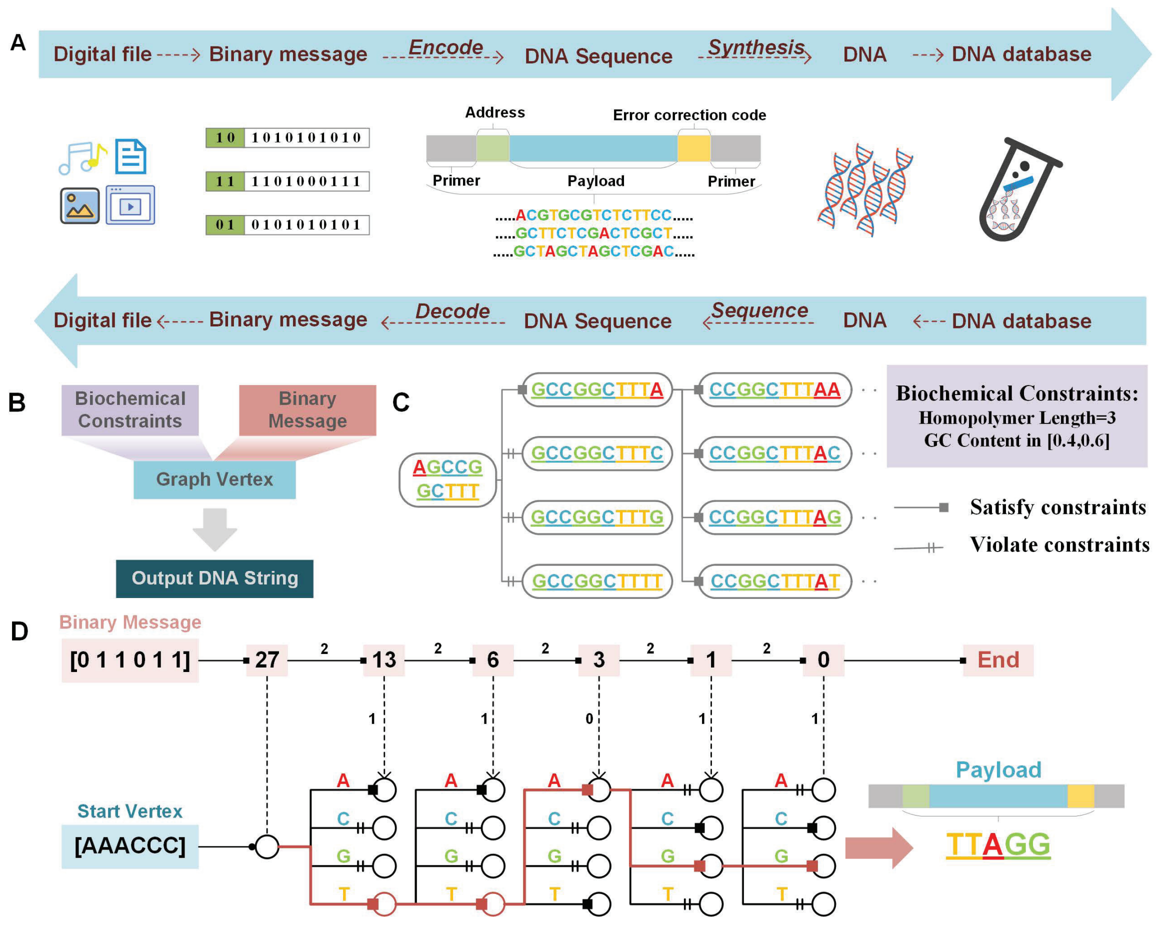

Fig. 1. The Complete Process of DNA Storage pipeline and the **Explorer** Encoding-Decoding Procedure. (A) The Complete Process of DNA storage pipeline. (B) Global Demonstration of the **Explorer**. The algorithm employs binary information as input and takes into account specific biochemical constraints to encode DNA sequences that satisfy said constraints, utilizing the De Bruijn graph and its vertices. (C) In the **Explorer** encoding process, the available adjacent vertices are selected from the vertices in the De Bruijn graph according to the set biochemical constraints. (D) Application Example of the **Explorer** Encoding Algorithm.

to fulfill the demands of the DNA storage domain. Complex processes like DNA strand displacement, hybridization, and transcription bestow DNA molecules with enhanced maneuverability and functionality [12, 13, 14]. Additionally, the utilization of molecular biology techniques for cell line passage, manipulation, and information storage within living cells presents a promising avenue for DNA storage advancement [15, 16, 17, 18, 19]. Nevertheless, the application of these emerging technologies introduces additional potential biochemical constraints, encompassing homopolymer length, GC content, and undesired motifs, among others. Hence, the development of coding algorithms capable of accommodating more biochemical constraints is of growing significance. Both DNA **Fountain** and **Yin-Yang** Code employ screening programs that consider biochemical constraints to manage arbitrary constraints on the generated sequences. However, when confronted with strict biochemical constraints, these methods can give rise to issues related to inefficient coding or high time complexity [20, 21]. The **SPIDER-WEB** is a graph-based architecture for encoding that took local biochemical constraints into account [22]. Nonetheless, **SPIDER-WEB** remains incapable of encoding arbitrary local biochemical constraints due to the exponential growth of the number of vertices in the De Bruijn graph with increasing observation length. The **Hedges** employs modular operations to incorporate diverse local biochemical constraints, enabling constrained encoding. Nevertheless, when subjected to specific stringent constraints, **Hedges** may encounter encoding limitations [10].

In this paper, we propose the **Explorer**, an efficient and low-energy DNA encoding algorithm based on the De Bruijn graph that can encode sequential data under arbitrary local and global biochemical constraints. We express the subsequences of length n within the encoded DNA sequence as a vertex in De Bruijn graph. By ensuring that each vertex complies with specific biochemical constraints, we can guarantee that the obtained DNA sequences adhere to the local biochemical constraints. Numerical experiments demonstrate that **Explorer** exhibits greater stability under various local constraint lengths, significantly reducing the memory occupancy and improving the encoding efficiency. Motivated by the recent advancements in applying deep neural networks to DNA storage [23, 24], we introduce the **Codeformer**, an efficient end-to-end deep learning decoding method that enables rapid decoding of **Explorer** encoded sequences. Specifically, a translation neural network model is proposed to establish the mapping between DNA sequences and binary sequences, which significantly improves the decoding efficiency and expands the potential applications of the **Explorer** algorithm in large-scale decoding scenarios.

Methodology

Biochemical constraints

Since specific DNA patterns may lead to an increase of errors in DNA synthesis, PCR amplification, and DNA sequencing, biochemical constraints are introduced to guarantee DNA sequences suitable for the biochemical process [25]. More specifically, higher or lower GC **content** usually results in lower coverage during sequencing [7]. Long **homopolymer** in DNA molecules may cause more difficulties in DNA synthesis and higher error rates in DNA sequencing [6, 26]. The **undesired motifs** have important roles in specific biochemical manipulations or storage environments, such as some enzyme cutting sites [27], primer fragments [26], etc. The presence of these motifs in the payload area may lead to false amplification of DNA sequences [25]. Compared to global constraints, local constraints allow for sub-sequences of a certain length within the sequence to meet specific biochemical properties. Thus, local constraints are more stringent than global constraints, meaning that if local constraints are satisfied, global constraints must also be satisfied.

GC **content balance** (GC): The global constraint mandates that the proportion of G and C in the sequence falls within the interval $[\varepsilon_1, \varepsilon_2]$, which implies that the number of GC is within the range $[\varepsilon_1 L, \varepsilon_2 L]$ with L being the sequence length. On the other hand, the local constraint necessitates that the portion of G and C should lie within the range $[\varepsilon_1 \ell, \varepsilon_2 \ell]$ with ℓ being the locally constrained observation length. Obviously, we can divide the sequence into multiple sub-sequences of length ℓ by setting ℓ as a factor of L. Within each sub-sequence, the GC content falls within the specified range, ensuring that the content constraint is satisfied across the entire sequence. Mathematically, it holds the following relationship between the global and local GC content of DNA sequences

$$\mathrm{GC}_{\mathrm{Gloal}} = \sum \mathrm{GC}_{\mathrm{Local}} \in \sum [\varepsilon_1 \ell, \varepsilon_2 \ell] = [\varepsilon_1 L, \varepsilon_2 L]. \qquad (1)$$

Homopolymer length (HL): The local constraint mandates that the length of any homopolymer within a sub-sequence of length ℓ must not surpass M. Consequently, the entire sequence must not contain any homopolymer exceeding the length of M.

Undesired motifs (UM): Similar to homopolymer constraint, special fragments that are prohibited at the local level are also absent at the global level.

It demonstrates that implementing biochemical constraints at the local level can give rise to satisfying global biochemical constraints. Thus, we can approximate arbitrary global constraints with the combination of local constraints. Since the length of undesired motifs can range from a few characters to 30 characters, the encoding algorithm should accommodate local biochemical constraints within an observation length range from 0 to 30. In addition, it holds paramount importance that the observation length can be flexibly adjusted within a specific range.

Explorer - the De Bruijn graph-based encoding algorithm

The De Bruijn graph has been demonstrated to be valuable in the field of molecular biology because of the connectedness between adjacent vertices. It was originally used for genome assembly [28, 29], where Eulerian tours were performed to infer the underlying sequences. Since then, the de Bruijn graph-based sequence assembly methods were further studied for mRNA [30], metagenome [31, 32]. Beyond that, the application of De Bruijn graph also includes genomic variants detection [33], read correction [34], read mapping [35], and long-read assembly [36], etc.

In graph theory, the ℓ-dimensional De Bruijn graph of m symbols is a directed graph that represents the overlap between symbol sequences. For the De Bruijn graph of m symbols, i.e., $S = \{s_1, s_2, \ldots, s_m\}$, we use D_m^ℓ to denote the De Bruijn graph with each vertex being a sequence of length-ℓ within the given m symbols. It then contains m^ℓ vertices, consisting of all possible length-ℓ sequences of the given symbols, defined as

$$D_m^\ell = \{\, (\mathrm{s}_1, \cdots, \mathrm{s}_1,\ \mathrm{s}_1), (\mathrm{s}_1, \cdots, \mathrm{s}_1,\ \mathrm{s}_2), \cdots, (\mathrm{s}_m, \cdots, \mathrm{s}_m, \mathrm{s}_m) \,\}. \qquad (2)$$

Given two vertices A and B, if $A[2{:}\ell]$ is identical to $B[1{:}\ell\text{-}1]$, it implies the existence of an edge from A to B, where the edge is denoted by $B[\ell]$. Furthermore, we designate B as the forward adjacent vertex of A, and conversely, we refer to A as the backward adjacent vertex of B. Thus, both A and B are defined as adjacent vertices of each other.

In our work, we use the De Bruijn graph D_4^ℓ, where ℓ is the observed length of the local biochemical constraints and 4 is used for quaternary encoding. We propose the **Explorer**, which is a coding algorithm under arbitrary biochemical constraints based on the De Bruijn graph. It begins by selecting a vertex that satisfies local constraints. We iteratively search for adjacent vertices meeting the biochemical constraints. By interconnecting the edges in a specific order, we can construct a DNA sequence satisfied the biochemical constraints globally. Note that each subsequence within this constructed sequence corresponds to vertices within the graph, ensuring the fulfillment of local biochemical constraints.

Let $\mathcal{X}$ denote a text sequence information in the form $[x_1 x_2 x_3 x_4 \cdots x_n]$ and $\mathcal{C}$ denote the biochemical constraint. We can obtain the DNA sequence by $\mathcal{Y} = F(\mathcal{X}, \mathcal{C})$ as shown in Fig. 1B. Note that $\mathcal{C}$ can be a combination of several biochemical constraints from **{GC,HL,UM}**. More specifically, we first convert the input binary message $\mathcal{X}$ into decimal numbers $\mathcal{R}$ as follows

$$\mathcal{R} = \sum x_i \cdot 2^{n-i}. \qquad (3)$$

Then we select the initial vertex $\mathcal{V}$ according to the biochemical constraints in the order of A $\rightarrow$ C $\rightarrow$ G $\rightarrow$ T. According to the biochemical constraints, we determine whether the forward adjacent vertex of the current vertex meets the biochemical constraints. Fig. 1C provides an example to show how to determine if the adjacent vertex satisfies biochemical constraints. As seen, the number of forward adjacent vertices that meet the constraint is used as the divisor p. The current decimal number serves as the dividend, and division is performed to obtain the quotient q and the remainder r. There exists the following relationship

$$\mathcal{R} = p * q + r, \quad r \in \{0, 1, 2, 3\}, \qquad (4)$$

where the current output bases is determined based on the remainder value r. We arrange the available bases to $\{0, 1, \ldots, \mathrm{p}-1\}$ in order of A < C < G < T and select the base corresponding to r. If the obtained quotient is not zero, the quotient q is set as a new dividend, and the process is repeated

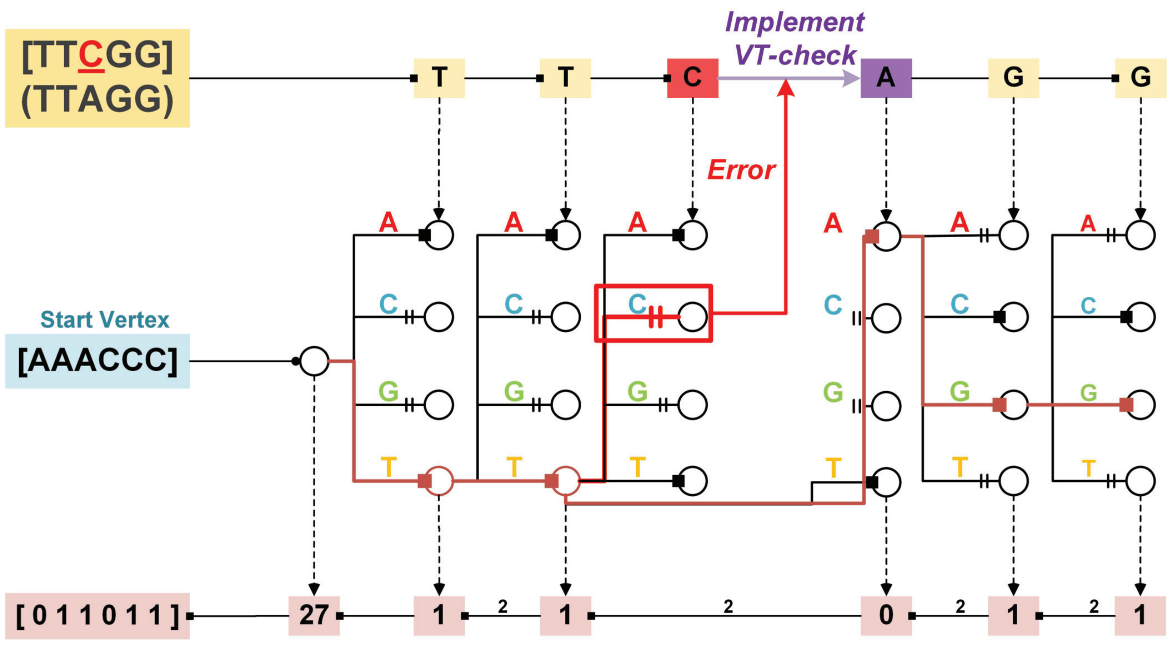

Fig. 2. Decoding and error correction in the case of the **Explorer**.

Algorithm 1 Explorer coding algorithm

```
1: Input: Binary message $\mathcal{X}$, Local biochemical constraints $\mathcal{C}$, De Bruijn Graph $D_4^\ell$
2: Convert $\mathcal{X}$ to the decimal number $\mathcal{R}$; Initialize $\mathcal{Y}$ as an empty string
3: Select the initial vertex $\mathcal{V}$ from $D_4^\ell$ based on constraints $\mathcal{C}$
4: while $\mathcal{R}! = 0$ do
5:     Find the forward adjacent vertex of the current vertex from $D_4^\ell$
6:     Calculate the number $p$ of the available vertex that satisfy the constraints $\mathcal{C}$
7:     Divide $\mathcal{R}$ by $p$, get the quotient $q$ and remainder $r$
8:     Selected a new available base from $\{A, C, G, T\}$ to add to $\mathcal{Y}$ based on the remainder $r$
9:     Set $\mathcal{R}$ as $q$ and set the current vertex according to the remainder $r$
10: end while
11: Return the DNA sequence $\mathcal{Y}$
```

to continuously add new bases to the current DNA sequence. When the obtained quotient q becomes zero, we output the final DNA sequence.

An example of **Explorer** coding is provided in Fig. 1D. The biochemical constraints are to require the homopolymer length not to exceed 3, the GC content to be in $[0.4, 0.6]$, and the observing length ℓ is set to 6. It shows the conversion of binary message [011011] into DNA sequence [TTAGG]. First, it converts the binary sequence into decimal to 27 and set the initial vertex to [AAACCC]. Then it judges whether the forward adjacent vertices of the current vertex [AAACCC] satisfy the local biochemical constraints. Among them, two vertices [AACCCA, AACCCT] satisfy the biochemical constraints. Then it sets 27 as the dividend and 2 as the divisor to perform division with remainder to get the quotient 13 and the remainder of 1. The next base that can be selected at this vertex is $\{A, T\}$. According to the rule between $\{0, 1\}$ and $\{A, T\}$, T is selected as the next base. Thus, the quotient is used as the dividend in the next step, and the vertex is slid to the selected vertex, and so on. Finally, when the quotient is 0, it outputs the resulting complete DNA sequence [TTAGG]. To sum up, we conclude **Explorer** coding algorithm as **Algorithm 1**.

The decoding of **Explorer** reverses the coding process as shown in the formula below. Fig. 2 shows the decoding and error correction algorithms. Similarly, both decoding and error correction algorithms can handle arbitrary local biochemical constraints. The specific formulation of the decoding process can be expressed by $\mathcal{X} = F^{-1}(\mathcal{Y}, \mathcal{C})$.

Codeformer – the transformer-based decoding algorithm

Although the coding efficiency of DNA storage has been improved by the **Explorer** method, it remains inconspicuous. To enhance coding efficiency, we explore the application of deep learning methods for expediting coding processes. Considering the nature of both input and output as sequential data, the entire procedure is conceptualized as a seq2seq process. Accordingly, a neural machine translation model, incorporating an attention mechanism, is employed to acquire the mapping from DNA sequences to binary information sequences. While deep learning approaches have previously found applications

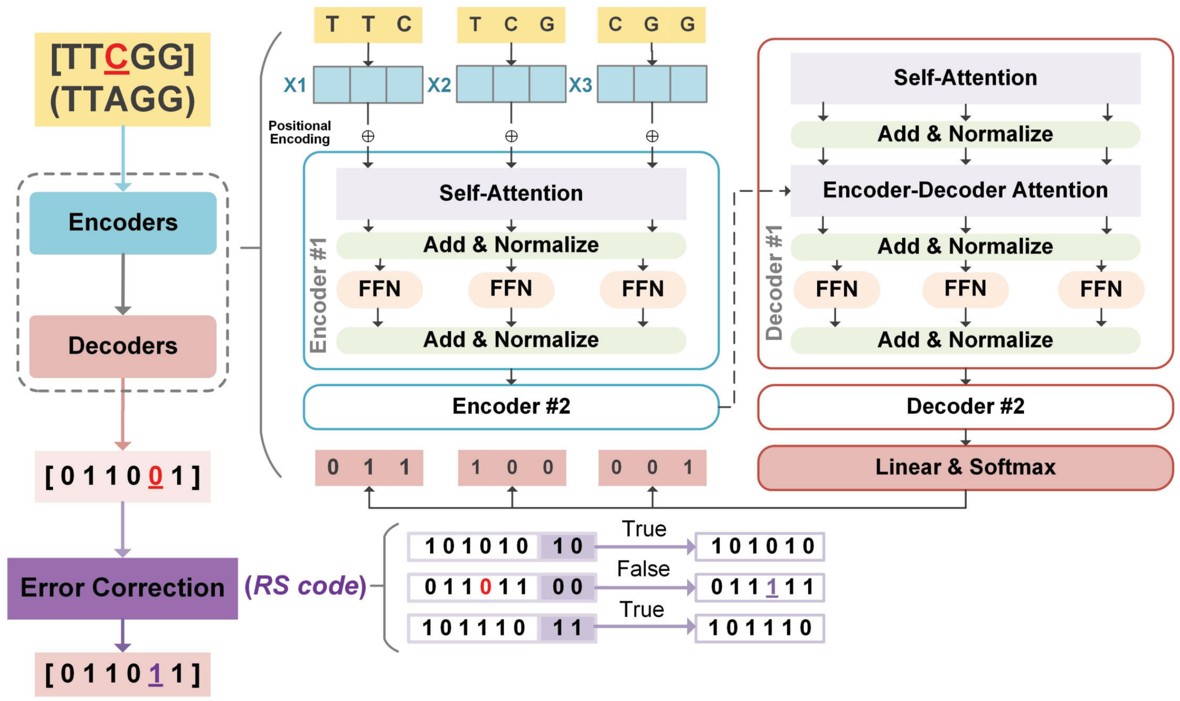

Fig. 3. Decoding and error correction in the case of the **Codeformer**.

in DNA storage, this marks the inaugural application of such methods in the domain of encoding and decoding. Upon completion of training, the model can directly map DNA sequences to binary message sequences, realizing an end-to-end process, which is formulated by

$$\mathcal{X} = T(\mathcal{Y}; \theta). \quad (5)$$

To meet our scalability needs, we implement a highly scalable neural network translation model called the **Codeformer** to learn DNA coding mapping; see Fig. 3. We use the binary sequences and their corresponding DNA sequences encoded by the **Explorer** as the target and source languages, respectively, as the input of the decoder and encoder during the training process. To generate word sequences, the k-mers word segmentation method is employed. Supposing $k = 3$, we can segment the DNA sequence as follows

$$[x_1x_2x_3x_4 \cdots x_n] \rightarrow [x_1x_2x_3, x_2x_3x_4, \cdots, x_{n-2}x_{n-1}x_n].$$

Fig. 3 elucidates the process in which the encoder transforms each tokenized word in a DNA sequence into a corresponding word vector. The vector is subsequently transformed into a hidden layer representation using a self-attention layer followed by a feed-forward neural network (FFN) layer. The decoder then forecasts each potential succeeding word in the binary sequence by amalgamating the vector representation of the previously generated word with the hidden layer vector produced by the encoder. The hidden layer representation is subjected to the softmax layer, generating probabilities that guide the selection of the target word. The attention layer modulates the impact of source latent vectors on the distribution, assigning weights to each source word based on its contribution to target prediction.

Upon the completion of comprehensive training of the **Codeformer** model using the designated training set, effective acquisition of rules for decoding DNA sequences into binary sequences is achieved. To decode DNA sequences, an initial step involves tokenization, leading to the creation of DNA word sequences. These sequences are then directly converted into binary word sequences, and subsequently, into binary sequences. While the deep learning method significantly enhances decoding efficiency, it is also accompanied by an increase in the decoding error rate. As shown in Fig. 3, we introduce RS code [37] to rectify errors that occur during the decoding process.

Results and Discussions

Performance evaluation of **Explorer**

The **Explorer** is a graph-based coding algorithm that enables efficient, stable, and low-energy DNA coding under both local and global biochemical constraints, exhibiting consistently high coding efficiency. In this subsection, we analyze the coding efficiency of our Explorer with respect to different observation length and biochemical constraints. Fig. 4A illustrates the variation of the coding efficiency across seven types of biochemical constraints. As can be observed, when the same biochemical constraints is used, coding efficiency decreases as the observation length increases. Similarly, with increased complexity of biochemical constraints, there is also a notable decrease in coding efficiency. Thus, longer observation length and more complex biochemical constraints prolong the time required to ascertain whether a local sequence meets the biochemical constraints, resulting in a decrease in encoding efficiency.

Fig. 4B illustrates the coding efficiency of our Explorer with respect to different lengths of binary sequence and observation lengths. When we fix the length of the binary sequence, an increase in observation length results in a decrease in coding

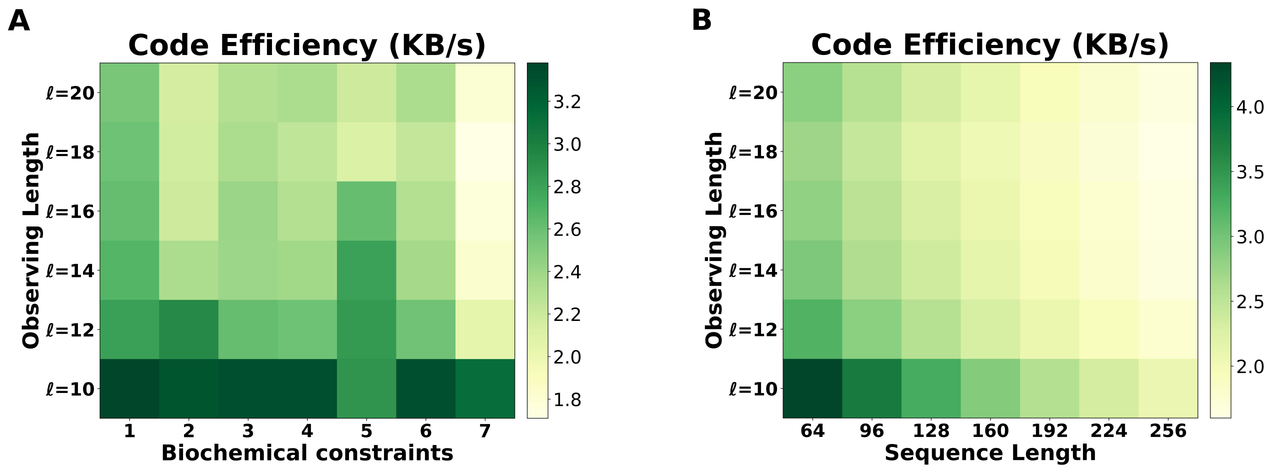

Fig. 4. Variation of coding efficiency affected by different conditions. (A) The variation of coding efficiency under different constraint lengths under seven kinds of constraints. (B) The variation of coding efficiency at different sequence lengths and different constraint lengths.

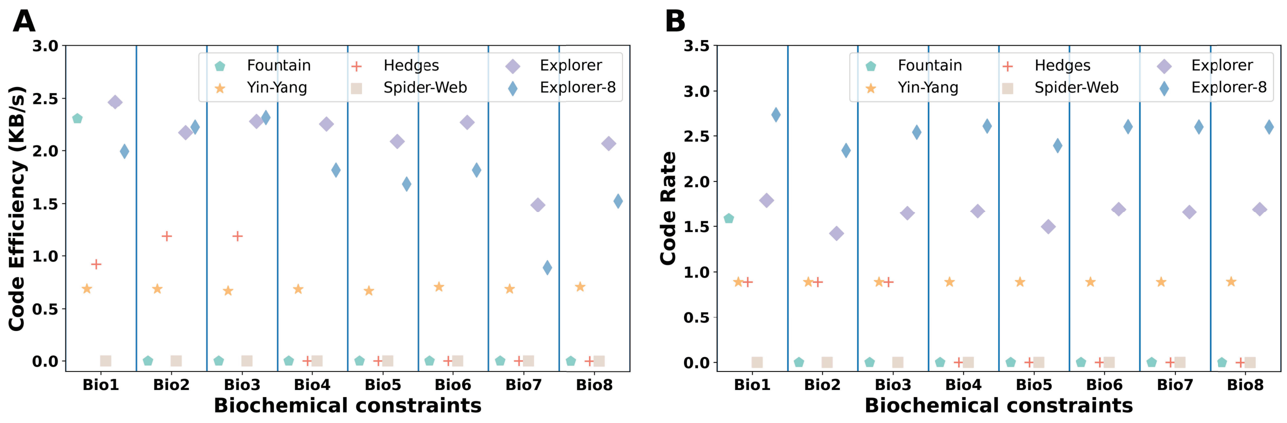

Fig. 5. (A) Comparison of **Fountain**, **Hedges**, **Explorer**, **Yin-Yang**, **SPIDER-WEB**, **Explorer-8** on the code efficiency. (B) Comparison of **Fountain**, **Hedges**, **Explorer**, **Yin-Yang**, **SPIDER-WEB**, **Explorer-8** on the code rate.

efficiency. It is because that verifying whether a local sequence satisfies biochemical constraints becomes increasingly time-consuming. Furthermore, we find that the coding efficiency also decreases as the binary sequence length increases. When the length of the binary sequence increases, the decimal value grows exponentially, which increases the complexity and time of calculations, ultimately resulting in a decrease in encoding efficiency. To sum up, the encoding efficiency of Explorer is impacted by biochemical constraints, the length of sequences and the length of the observation length.

Comparison with SOTA DNA coding methods

Several other coding algorithms allow encoding under specific biochemical constraints, such as **SPIDER-WEB** [22], **Fountain** [20], **Hedges** [10] and **Yin-Yang** [21]. We compared the coding rate and coding efficiency of the **Explorer** with these DNA coding algorithms by encoding the same information with respect to different biochemical constraints. The detailed biochemical constraints are displayed in Table 1, where the observation length is set to 20 for all experiments. In [39], Hoshika *et al.* proposed a more efficient DNA encoding method using artificial bases. By increasing the DNA base, the amount of DNA synthesis per storage unit is reduced. By modifying the molecular structure of natural bases (ATCG), two artificial pairs of bases (SB and PZ) are involved in DNA coding. Combining artificial bases with natural bases, a total of eight different bases are used to synthesize DNA. Thus, we also integrated the concept of octal base encoding into **Explorer** and referred it as **Explorer-8**. The comparison results are presented in Fig. 5A. It demonstrates that **Explorer** can maintain a relatively stable coding efficiency under the eight different biochemical constraints and performs better than the other coding algorithms. Although both **SPIDER-WEB** and **Explorer** are coding algorithms based on graph structures, **SPIDER-WEB** fail to encode data when the observation length reaches 20. Moreover, we observe that **SPIDER-WEB**, **Fountain**, and **Hedges** fail to encode the DNA sequences with certain constraints, resulting in a coding efficiency of zero.

In terms of code rate, as demonstrated in Fig. 5B, **Explorer** exhibits its superiority by achieving a higher code rate than other algorithms. It confirms the efficiency of **Explorer** in enhancing data storage density, particularly **Explorer-8**, which boosts the code rate by utilizing eight different bases for encoding, thereby surpassing **Explorer** and other

Table 1. Local biochemical constraints

Index	Homopolymer	GC content	Motifs
01	3	N	N
02	N	[0.1, 0.3]	N
03	N	[0.5, 0.7]	N
04	3	[0.4, 0.6]	N
05	4	[0.45, 0.55]	N
06	4	[0.4, 0.6]	N
07	2	[0.4, 0.6]	Motif-1
08	4	[0.4, 0.6]	Motif-2

* Motif-1: AGCT, GACGC, CAGCAG, GATATC, GGTACC, CTGCAG, GAGCTC, GTCGAC, AGTACT, ACTAGT, GCATGC, AGGCCT, TCTAGA [38]

* Motif-2: CTTGGTCAGACGAGTGCATG, GAGTTACGCGGGGATACATG (Primer) [26]

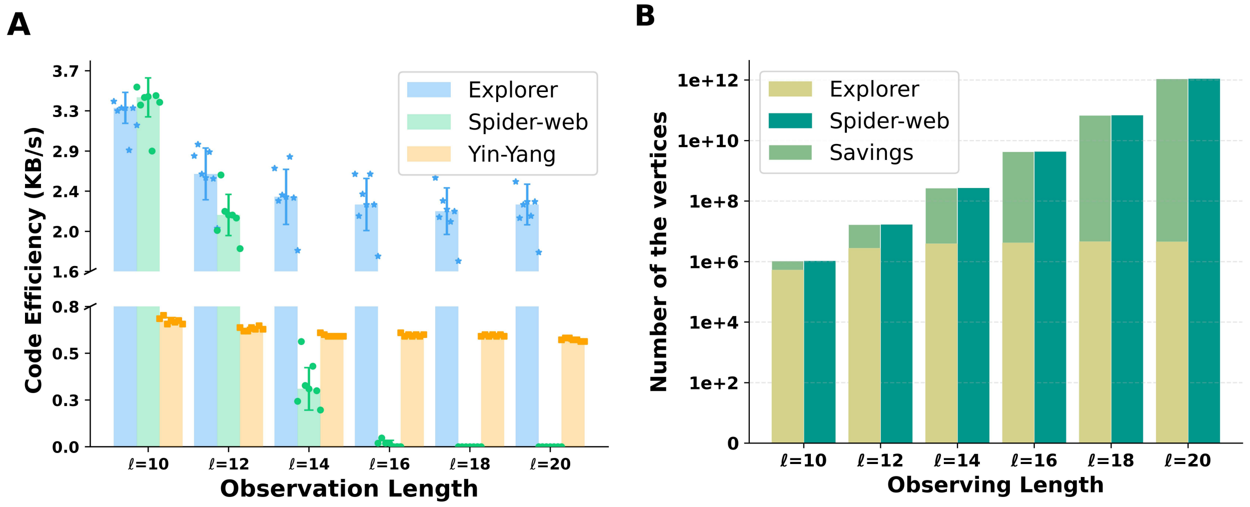

Fig. 6. Comparison between **Explorer**, **SPIDER-WEB** and **Yin-Yang**. (A) The Coding efficiency comparison between **Explorer**, **SPIDER-WEB**, and **Yin-Yang**. The scatter point represents the results of individual experiments, while the histogram heights are constructed from the means of seven experiments. (B) The number of the parameter comparison between **Explorer** and **SPIDER-WEB**.

coding algorithms. The enhanced base diversity delivers a higher information density for DNA storage, yielding significant advantages in storage capacity. The experimental results underscore the efficiency and stability of **Explorer** in dealing with various biochemical constraints, as well as its pronounced advantages in code rate over other existing coding methods.

Furthermore, we compare the time complexity and space complexity across the observation length ranged from 10 to 20. As shown in Fig. 5A, **Yin-Yang** demonstrates excellent coding efficiency and maintains stable performance with respect to different biochemical constraints. And **SPIDER-WEB** is the only coding algorithm based on graph structure. Therefore, we assess the time complexity among the three coding methods: **Explorer**, **SPIDER-WEB**, and **Yin-Yang** for different observation lengths. As can be seen in the Fig. 6A, the coding efficiency of **Yin-Yang** remains stable across different observation lengths without a significant increase as the observation length extending. On the other hand, although **SPIDER-WEB** exhibits the best coding efficiency when the observation length is short, e.g., $\ell = 10$, its coding efficiency significantly diminishes as the observation length increases. In contrast, **Explorer** maintains stable and high coding efficiency across all observed lengths. Notably, when observation length is greater than or equal to 12, its performance markedly surpasses that of the other two methods. Thus, Explorer is more suitable for dealing with globally biochemical constraints.

For the space complexity comparison, we employed the number of vertices in the graph as the comparative metric. Therefore, we compared the Explorer with the other graph-based coding method **SPIDER-WEB**. As shown in Fig. 6B, when **SPIDER-WEB** handles longer observation lengths, the number of parameters increases exponentially, necessitating significant time and space resources for graph construction. The exponential growth of space complexity eventually surpasses the processing capabilities of typical computers, resulting in **SPIDER-WEB** incapable of encoding tasks for observation length larger than 18. In stark contrast, the number of vertices of our **Explorer** maintains stable for all observation lengths, such that it is minimally affected by increasing the observation length.

In addition, we conduct the experiments to evaluate both the time complexity and space complexity of **Explorer** with different sizes of input data. As depicted in Fig. 7, both the memory space utilized and the time expended by the **Explorer** increase linearly with the size of the input data. Such linear relationship indicates that the model possesses commendable

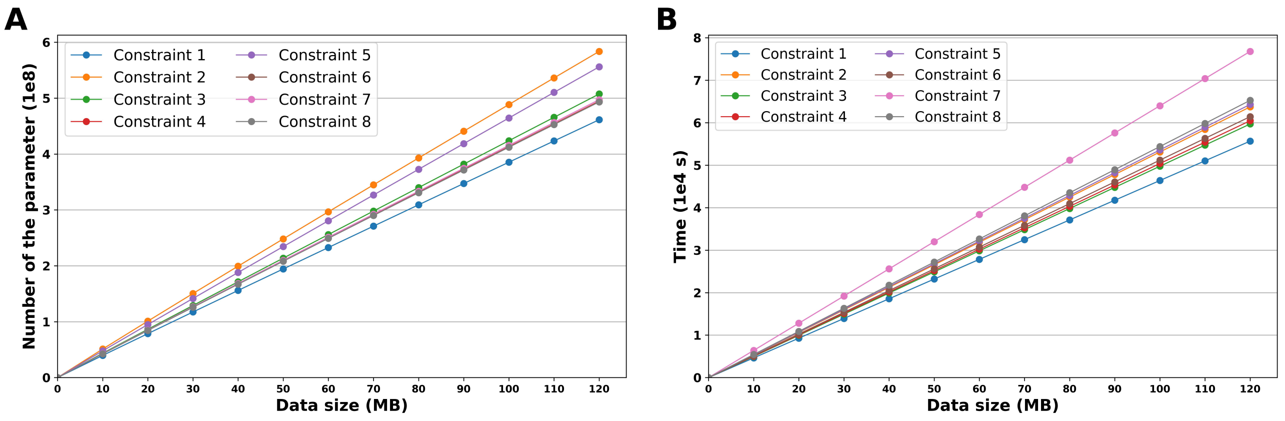

Fig. 7. The time and space complexity with respect to different sizes of data. (A) The time complexity of **Explorer** with different input data sizes. (B) The space complexity of **Explorer** with different input data sizes.

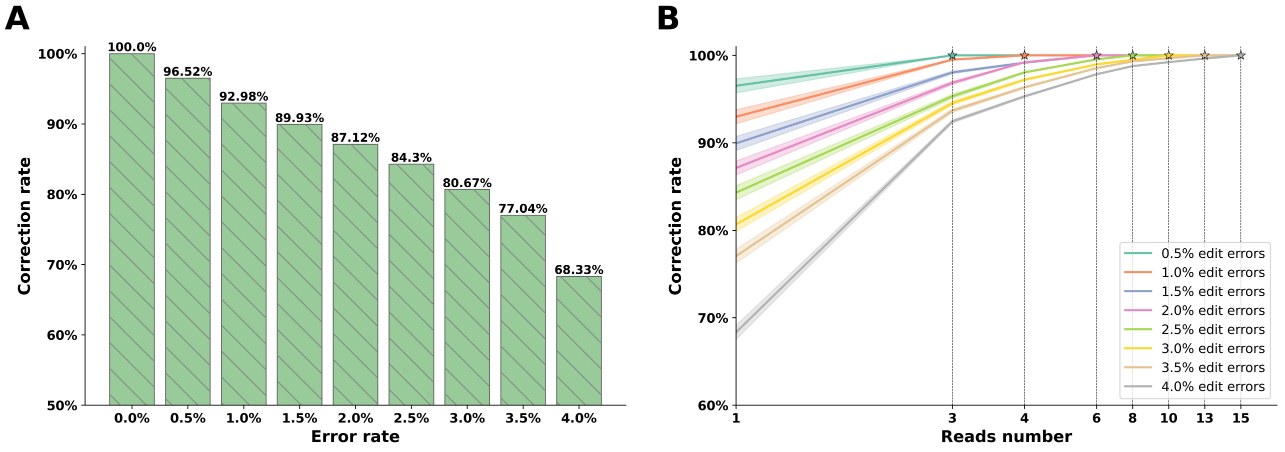

Fig. 8. The evaluation of decoding accuracy. (A) The correction rate of **Explorer** under different error rates. (B) The minimum reads number required to support that the sequence with the highest frequency is the correct sequence.

scalability. The predictable nature of the linear growth is crucial for resource planning and system design, as it allows for the anticipation of resource requirements as data volumes expands.

Decoding accuracy of **Explorer**

In this section, we experimentally validate the decoding and error correction performance of Explorer. To accord with the existing DNA storage technologies, we introduce a set of constraints to examine the decoding performance of our Explorer. More specifically, we set the homopolymer length to be 2 to better accommodate enzymatic DNA synthesis [40] and the GC content to be 50% to enhance the success rate of PCR amplification [41]. Additionally, we prohibit specific sequences that may lead to increased error rates [26, 38].

We use the decoding accuracy as a metric to evaluate the precision of decoding. The criterion for accurate decoding stipulates that each position within the sequence should be correctly decoded, underscoring the imperative for reliability in the data conversion process. Mathematically, we can define the decoding accuracy as

$$\text{Accuracy} = \frac{\#\{\text{Decoding Sequence} == \text{Encoding Sequence}\}}{\#\{\text{Input Sequence }\}}, \tag{6}$$

where # denotes the number of elements. Obviously, a sequence contributes to the decoding Accuarcy only if it is perfectly corrected at all positions. Moreover, in the context of DNA sequence encoding and decoding, high decoding accuracy exemplifies outstanding model performance.

We use 10,000 DNA sequences (DNA of length 200 nt) that satisfied biochemical constraints as decoding DNA samples. For simplicity, we assume that substitutions, insertions, and deletions occur with equal probability and that errors occur at random positions in DNA sequences. The correction rates with respect to different error rates are shown in Fig. 8A. As can be seen, Explorer can correct most errors and achieve a decoding accuracy of 96.5%, when there is a 0.5% error rate (i.e., one editing error in a 200 nt long sequence).

Additionally, considering that the DNA sequencing process typically involves multiple sequence copies, we can assume that the correct sequence is the one with the highest frequency. By increasing the number of copies, the proportion of correct

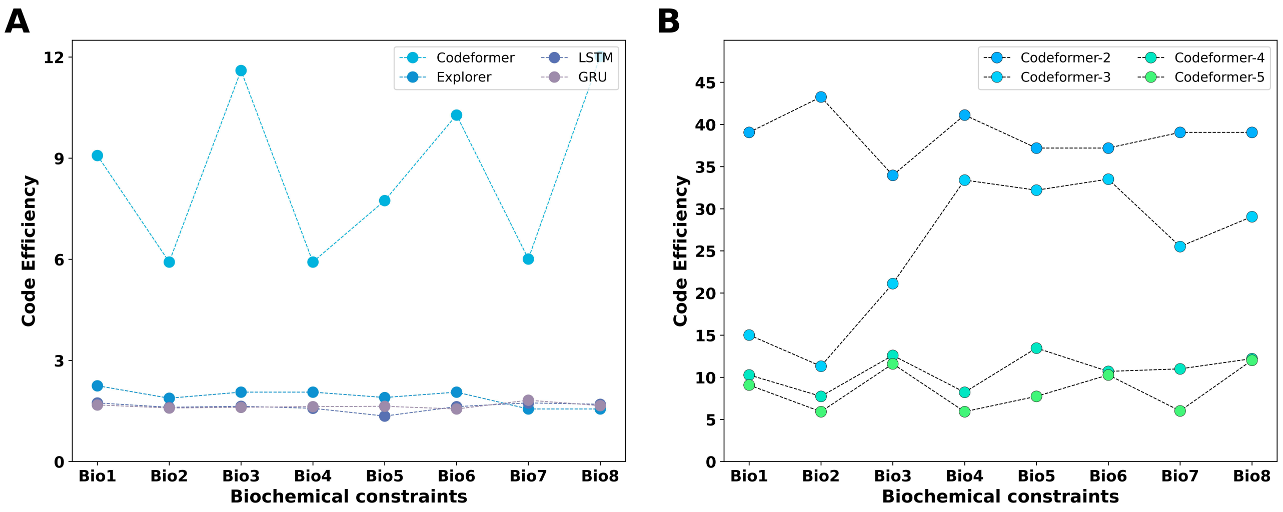

Fig. 9. (A) Comparison of code Efficiency of **Codeformer**, **Explorer**, and Models Based on LSTM and GRU. (B) Code efficiency of the **Codeformer** under different search widths.

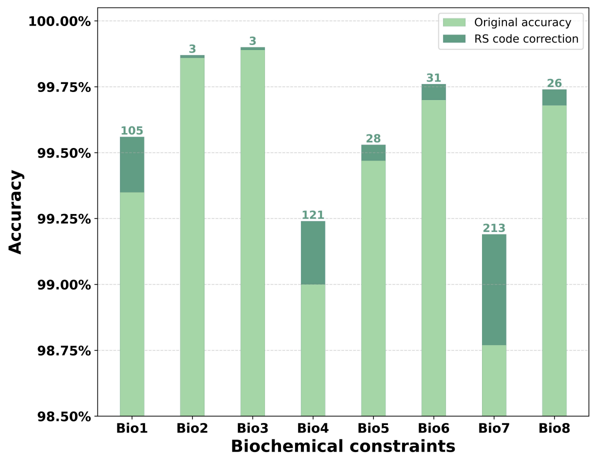

Fig. 10. Decoding accuracy of the **Codeformer** and the number of chains corrected by the RS code.

sequences among all sequences can be increased and used for decoding. To study the minimum reads number required for the above hypothesis, that is, the correct sequence occurring most frequently, we conducted 10,000 random tests with the different error rates mentioned above. As shown in Fig. 8B, when the error rate is 0.5%, the correct decoding sequence can be obtained through the above assumption when the reads number exceeds 3. For the above assumptions to hold, the number of reads per sequence needs to be no less than 3, 4, 6, 10, and 15 for editing error rates of 0.5%, 1.0%, 2.0%, 3.0%, and 4.0%, respectively. Therefore, by utilizing a small number of sequence copies, our Explorer can correctly decode the DNA sequences when edit errors are involved in the data.

Performance evaluation of **Codeformer**

The training and testing datasets for **Codeformer** consist of pairs comprising a binary sequence and a DNA sequence. The DNA sequences are obtained using the Explorer from binary sequences. We build up the training dataset including 300,000 pairs of sequences, and the test dataset consisting of 50,000 pairs. It is important to note that the testing dataset is independent and meticulously curated to ensure there is no overlap or duplication with the training dataset.

At the first place, we conduct the ablation experiments on the **Codeformer** by replacing the transformer module with LSTM and GRU. As shown in Fig. 9A, when the transformer module is replaced with two alternative networks, the code efficiency of the decoding algorithm decreases. It demonstrates that **Codeformer** can operate more efficiently than LSTM and GRU models. The increased efficiency is attributable to the powerful parallel computing ability of Transformer in processing data. The **Codeformer** utilizes a beam search strategy to enhance its selection process, which is a heuristic algorithm used in graphical models to keep multiple probable decoding options at each step. It allows the exploration of more decoding paths, potentially improving accuracy. The beam width, a crucial parameter, determines the number of sequences retained per step. Adjusting this width balances accuracy and efficiency: larger widths enhance accuracy but increase computational demand, while smaller widths speed up decoding at the potential cost of reduced accuracy. As shown in Fig. 9A, when the beam search width is set to 5, the decoding efficiency of **Codeformer** is nearly ten times faster than **Explorer**. Moreover, Fig. 9B illustrates decoding efficiency of **Codeformer** using different beam search widths, where X in **Codeformer**-X denotes the beam search width set for **Codeformer**. The results indicate that as the search width increases, the decoding efficiency of the **Codeformer** gradually decreases. It is because the larger search width requiring the **Codeformer** to evaluate more potential decoding paths. Therefore, we set the beam width to be 5 to balance the efficiency and accuracy.

These results demonstrate that **Codeformer** has high decoding efficiency. Although increasing the search width may reduce the overall effectiveness of **Codeformer**, it can still maintain high efficiency relative to traditional algorithms, even at higher search widths. It effectively addresses the challenge of large-scale parallel decoding.

Although the deep learning based DNA coding can substantially enhance encoding and decoding efficiency, it may also suffer from the increase in decoding error rate. Thus, we employ Reed-Solomon (RS) codes [37] to correct errors of **Codeformer** decoding, aiming to reduce the error rate and maintain efficient encoding. A series of experiments are conducted to evaluate the performance of RS code in correcting the decoding error. Fig. 10 illustrates the decoding accuracy of **Codeformer** and the number of DNA chains corrected by RS code at a search width of 5. Obviously, the RS code successfully increases the decoding accuracy of **Codeformer**. It demonstrates that by combining the **Codeformer** with RS codes, the goal of improving decoding efficiency while reducing encoding and decoding error rates can be successfully achieved.

Prospects and limitations

With the rapid advancements in deep learning, various learning-based encoding-decoding models have been proposed for DNA storage [42, 43, 44, 45, 46, 47]. Compared to traditional source and channel coding strategies, deep learning-based methods significantly enhance encoding and decoding efficiency, which is crucial given the massive volume of data involved. It is noteworthy that many learning-based encoding-decoding methods, including our **Codeformer**, are data-driven. These methods leverage neural networks to learn the mapping between binary information and codes generated using different coding techniques, such as finite-state constrained codes [45] and Turbo codes [43]. Moreover, learning-based methods have demonstrated high efficiency in image data encoding and decoding [47, 46, 42].

However, deep learning models are often criticized for their black-box nature, which obfuscates their internal decision-making processes. This lack of transparency can significantly hinder applications in fields where understanding the rationale behind each decision is critical. Specifically, in the DNA storage domain, where the accuracy of every encoded and decoded sequence is paramount, the inability to interpret the logic of coding methods could limit their applications. Efforts have been made to derive model-based methods to enhance the interpretability of the channel Encoding/Decoding in the context of DNA storage. For example, a forward-backward algorithm associated with marker codes [44] is unfolded into a network to enhance interpretability. Given the development of interpretable graph models, including XGNN [48], SubgraphX [49] etc., we aim to develop an interpretable graph neural network by incorporating De Bruijn graph structure for DNA coding in the future.

Conclusion

In this work, we presented the **Explorer**, which is a DNA coding method capable of handling increasingly intricate local biochemical constraints, including toxic DNA sequences, primers, and other lengthy sequences that amplify biochemical errors within the DNA storage system [50, 51]. By utilizing the structure of the De Bruijn graph, and ensuring that the vertices in the graph meet specific biochemical constraints, the entire DNA sequence can be guaranteed to satisfy the corresponding local biochemical constraints. The Explorer can provide sequences that simultaneously adhere to arbitrary local biochemical constraints and meet arbitrary global constraints. It also has advantages compared with other advanced coding algorithms in terms of code rate and code efficiency. What is more, we also proposed the **Codeformer** model, which is a transformer-based learning model. The **Codeformer** can achieve more efficient conversion between binary information and DNA strings, which may help to promote industrialization of DNA storage.

Although the proposed coding methods demonstrate superiority, they still face several unresolved issues. Going forward, enhancements to the **Explorer** can involve designing it as a fixed-length code generation algorithm to ensure uniform DNA sequence length. To address the issue of sequence errors in the **Codeformer**, it is essential to develop effective error correction methods. Additionally, future research should concentrate on the development of models that harmonize efficiency with transparent decision-making processes, thereby enhancing their applicability in the field of DNA storage.

Data and code availability

The datasets and the codes are freely available at https://github.com/DouC17/Explorer

Funding

This work was supported by the National Key Research and Development Program of China under grant No. 2020YFA0712100.

Key Points

- We introduced **Explorer**, an coding algorithm implemented using De Bruijn graph, enabling DNA encoding and decoding under various local or global biochemical constraints.
- **Explorer** outperforms other notable encoding algorithms in terms of both coding efficiency and code rate.
- We introduced **Codeformer**, a deep learning decoding algorithm founded on the transformer model.
- **Codeformer** facilitates more efficient conversion between binary information and DNA strings, achieving decoding speeds up to ten times faster than traditional methods.

References

1. Ben Cao, Xiaokang Zhang, Shuang Cui, and Qiang Zhang. Adaptive coding for dna storage with high storage density and low coverage. *NPJ systems biology and applications*, 8(1):23, 2022.
2. Chengtao Xu, Biao Ma, Zhongli Gao, Xing Dong, Chao Zhao, and Hong Liu. Electrochemical dna synthesis and sequencing on a single electrode with scalability for integrated data storage. *Science Advances*, 7(46):eabk0100, 2021.
3. Bichlien H Nguyen, Christopher N Takahashi, Gagan Gupta, Jake A Smith, Richard Rouse, Paul Berndt, Sergey Yekhanin, David P Ward, Siena D Ang, Patrick Garvan, et al. Scaling dna data storage with nanoscale electrode wells. *Science advances*, 7(48):eabi6714, 2021.

4. Guanjin Qu, Zihui Yan, and Huaming Wu. Clover: tree structure-based efficient dna clustering for dna-based data storage. *Briefings in Bioinformatics*, 23(5):bbac336, 2022.
5. Shufang Zhang, Beibei Huang, Xiangming Song, Tao Zhang, Hanjie Wang, and Yuhong Liu. A high storage density strategy for digital information based on synthetic dna. *3 Biotech*, 9:1–10, 2019.
6. Nick Goldman, Paul Bertone, Siyuan Chen, Christophe Dessimoz, Emily M LeProust, Botond Sipos, and Ewan Birney. Towards practical, high-capacity, low-maintenance information storage in synthesized dna. *nature*, 494(7435):77–80, 2013.
7. Michael G Ross, Carsten Russ, Maura Costello, Andrew Hollinger, Niall J Lennon, Ryan Hegarty, Chad Nusbaum, and David B Jaffe. Characterizing and measuring bias in sequence data. *Genome biology*, 14:1–20, 2013.
8. George M Church, Yuan Gao, and Sriram Kosuri. Next-generation digital information storage in dna. *Science*, 337(6102):1628–1628, 2012.
9. Tuan Thanh Nguyen, Kui Cai, Kees A Schouhamer Immink, and Han Mao Kiah. Capacity-approaching constrained codes with error correction for dna-based data storage. *IEEE Transactions on Information Theory*, 67(8):5602–5613, 2021.
10. William H Press, John A Hawkins, Stephen K Jones Jr, Jeffrey M Schaub, and Ilya J Finkelstein. Hedges error-correcting code for dna storage corrects indels and allows sequence constraints. *Proceedings of the National Academy of Sciences*, 117(31):18489–18496, 2020.
11. Xiayang Li, Moxuan Chen, and Huaming Wu. Multiple errors correction for position-limited dna sequences with gc balance and no homopolymer for dna-based data storage. *Briefings in Bioinformatics*, 24(1):bbac484, 2023.
12. Callista Bee, Yuan-Jyue Chen, Melissa Queen, David Ward, Xiaomeng Liu, Lee Organick, Georg Seelig, Karin Strauss, and Luis Ceze. Molecular-level similarity search brings computing to dna data storage. *Nature communications*, 12(1):4764, 2021.
13. Boya Wang, Cameron Chalk, and David Soloveichik. Simd—— dna: Single instruction, multiple data computation with dna strand displacement cascades. In *DNA Computing and Molecular Programming: 25th International Conference, DNA 25, Seattle, WA, USA, August 5–9, 2019, Proceedings 25*, pages 219–235. Springer, 2019.
14. Kevin N Lin, Kevin Volkel, James M Tuck, and Albert J Keung. Dynamic and scalable dna-based information storage. *Nature communications*, 11(1):2981, 2020.
15. Seth L Shipman, Jeff Nivala, Jeffrey D Macklis, and George M Church. Crispr–cas encoding of a digital movie into the genomes of a population of living bacteria. *Nature*, 547(7663):345–349, 2017.
16. Yangyi Liu, Yubin Ren, Jingjing Li, Fan Wang, Fei Wang, Chao Ma, Dong Chen, Xingyu Jiang, Chunhai Fan, Hongjie Zhang, et al. In vivo processing of digital information molecularly with targeted specificity and robust reliability. *Science Advances*, 8(31):eabo7415, 2022.
17. Lifu Song and An-Ping Zeng. Orthogonal information encoding in living cells with high error-tolerance, safety, and fidelity. *ACS synthetic biology*, 7(3):866–874, 2018.
18. Muhalb M Alsaffar, Mohammad Hasan, Gavin P McStay, and Mohamed Sedky. Digital dna lifecycle security and privacy: an overview. *Briefings in Bioinformatics*, 23(2):bbab607, 2022.
19. Jialu Hu, Yuanke Zhong, and Xuequn Shang. A versatile and scalable single-cell data integration algorithm based on domain-adversarial and variational approximation. *Briefings in Bioinformatics*, 23(1):bbab400, 2022.
20. Yaniv Erlich and Dina Zielinski. Dna fountain enables a robust and efficient storage architecture. *science*, 355(6328):950–954, 2017.
21. Zhi Ping, Shihong Chen, Guangyu Zhou, Xiaoluo Huang, Sha Joe Zhu, Haoling Zhang, Henry H Lee, Zhaojun Lan, Jie Cui, Tai Chen, et al. Towards practical and robust dna-based data archiving using the yin–yang codec system. *Nature Computational Science*, 2(4):234–242, 2022.
22. Haoling Zhang, Zhaojun Lan, Wenwei Zhang, Xun Xu, Zhi Ping, Yiwei Zhang, and Yue Shen. Spider-web generates coding algorithms with superior error tolerance and real-time information retrieval capacity. *arXiv preprint arXiv:2204.02855*, 2022.
23. Jinny X Zhang, Boyan Yordanov, Alexander Gaunt, Michael X Wang, Peng Dai, Yuan-Jyue Chen, Kerou Zhang, John Z Fang, Neil Dalchau, Jiaming Li, et al. A deep learning model for predicting next-generation sequencing depth from dna sequence. *Nature communications*, 12(1):4387, 2021.
24. Alan JX Guo, Cong Liang, and Qing-Hu Hou. Deep squared euclidean approximation to the levenshtein distance for dna storage. In *International Conference on Machine Learning*, pages 8095–8108. PMLR, 2022.
25. Marius Welzel, Peter Michael Schwarz, Hannah F Löchel, Tolganay Kabdullayeva, Sandra Clemens, Anke Becker, Bernd Freisleben, and Dominik Heider. Dna-aeon provides flexible arithmetic coding for constraint adherence and error correction in dna storage. *Nature Communications*, 14(1):628, 2023.
26. Jerrod J Schwartz, Choli Lee, and Jay Shendure. Accurate gene synthesis with tag-directed retrieval of sequence-verified dna molecules. *Nature methods*, 9(9):913–915, 2012.
27. Barry Polisky, Patricia Greene, David E Garfin, Brian J McCarthy, Howard M Goodman, and Herbert W Boyer. Specificity of substrate recognition by the ecori restriction endonuclease. *Proceedings of the National Academy of Sciences*, 72(9):3310–3314, 1975.
28. Ramana M Idury and Michael S Waterman. A new algorithm for dna sequence assembly. *Journal of computational biology*, 2(2):291–306, 1995.
29. Phillip EC Compeau, Pavel A Pevzner, and Glenn Tesler. How to apply de bruijn graphs to genome assembly. *Nature biotechnology*, 29(11):987–991, 2011.
30. Manfred G Grabherr, Brian J Haas, Moran Yassour, Joshua Z Levin, Dawn A Thompson, Ido Amit, Xian Adiconis, Lin Fan, Raktima Raychowdhury, Qiandong Zeng, et al. Full-length transcriptome assembly from rna-seq data without a reference genome. *Nature biotechnology*, 29(7):644–652, 2011.
31. Yu Peng, Henry CM Leung, Siu-Ming Yiu, and Francis YL Chin. Meta-idba: a de novo assembler for metagenomic data. *Bioinformatics*, 27(13):i94–i101, 2011.
32. Dinghua Li, Chi-Man Liu, Ruibang Luo, Kunihiko Sadakane, and Tak-Wah Lam. Megahit: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de bruijn graph. *Bioinformatics*, 31(10):1674–1676, 2015.
33. Zamin Iqbal, Mario Caccamo, Isaac Turner, Paul Flicek, and Gil McVean. De novo assembly and genotyping of

variants using colored de bruijn graphs. *Nature genetics*, 44(2):226–232, 2012.

34. Antoine Limasset, Jean-François Flot, and Pierre Peterlongo. Toward perfect reads: self-correction of short reads via mapping on de bruijn graphs. *Bioinformatics*, 36(5):1374–1381, 2020.
35. Fatemeh Almodaresi, Mohsen Zakeri, and Rob Patro. Puffaligner: a fast, efficient and accurate aligner based on the pufferfish index. *Bioinformatics*, 37(22):4048–4055, 2021.
36. Jue Ruan and Heng Li. Fast and accurate long-read assembly with wtdbg2. *Nature methods*, 17(2):155–158, 2020.
37. Robert N Grass, Reinhard Heckel, Michela Puddu, Daniela Paunescu, and Wendelin J Stark. Robust chemical preservation of digital information on dna in silica with error-correcting codes. *Angewandte Chemie International Edition*, 54(8):2552–2555, 2015.
38. Richard J Roberts. Restriction and modification enzymes and their recognition sequences. *Gene*, 8(4):329–343, 1980.
39. Shuichi Hoshika, Nicole A Leal, Myong-Jung Kim, Myong-Sang Kim, Nilesh B Karalkar, Hyo-Joong Kim, Alison M Bates, Norman E Watkins Jr, Holly A SantaLucia, Adam J Meyer, et al. Hachimoji dna and rna: A genetic system with eight building blocks. *Science*, 363(6429):884–887, 2019.
40. Roy Shafir, Omer Sabary, Leon Anavy, Eitan Yaakobi, and Zohar Yakhini. Sequence reconstruction under stutter noise in enzymatic dna synthesis. In *2021 IEEE Information Theory Workshop (ITW)*, pages 1–6. IEEE, 2021.
41. Yair Benita, Ronald S Oosting, Martin C Lok, Michael J Wise, and Ian Humphery-Smith. Regionalized gc content of template dna as a predictor of pcr success. *Nucleic acids research*, 31(16):e99–e99, 2003.
42. Wenfeng Wu, Luping Xiang, Qiang Liu, and Kun Yang. Deep joint source-channel coding for dna image storage: A novel approach with enhanced error resilience and biological constraint optimization. *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, 2023.
43. Marius Welzel, Hagen Dreßler, and Dominik Heider. Turbo autoencoders for the dna data storage channel with autoturbo-dna. *Iscience*, 27(5), 2024.
44. Guochen Ma, Xiaopeng Jiao, Jianjun Mu, Hui Han, and Yaming Yang. Deep learning-based detection for marker codes over insertion and deletion channels. *arXiv preprint arXiv:2401.01155*, 2024.
45. Panpan Li, Kui Cai, Guanghui Song, Wentu Song, Zhen Mei, and Xingwei Zhong. Neural network-based decoding of constrained codes for dna data storage. In *2020 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, pages 1–4. IEEE, 2020.
46. Jitesh Pradhan, Arup Kumar Pal, SK Hafizul Islam, and Chiranjeev Bhaya. Dna encoding-based nucleotide pattern and deep features for instance and class-based image retrieval. *IEEE Transactions on NanoBioscience*, 2023.
47. Chao Pan, S Kasra Tabatabaei, SM Hossein Tabatabaei Yazdi, Alvaro G Hernandez, Charles M Schroeder, and Olgica Milenkovic. Rewritable two-dimensional dna-based data storage with machine learning reconstruction. *Nature communications*, 13(1):2984, 2022.
48. Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. Xgnn: Towards model-level explanations of graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 430–438, 2020.
49. Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. In *International conference on machine learning*, pages 12241–12252. PMLR, 2021.
50. Xavier Pic and Marc Antonini. Image storage on synthetic dna using autoencoders. *arXiv preprint arXiv:2203.09981*, 2022.
51. David Mahan Knipe, Peter M Howley, et al. *Fundamental virology*. Number Ed. 4. Lippincott Williams & Wilkins, 2001.

Chang Dou. Chang Dou received the B.S. degree in mathematics from Ocean university of China, Qingdao, China, in 2021. He is currently working toward the M.S. degree at Tianjin University, Tianjin, China. His main interests include DNA storage and deep learning.

Yijie Yang. Yijie Yang received the B.S. degree in Mathematics from Harbin Engineering University (HEU), Harbin, Heilongjiang, China, in 2019. He is currently working toward the Ph.D. degree with the Center for Applied Mathematics, Tianjin University(TJU), Tianjin. His research interests include image segmentation, image reconstruction and DNA storage.

Fei Zhu. Fei Zhu received the B.S. degree in mathematics and applied mathematics and in economics from Xi'an Jiaotong University, Xi'an, China, in 2011. She received the M.S. and the Ph.D. degrees in systems optimization and security from the University of Technology of Troyes (UTT), Troyes, France, in 2013 and 2016, respectively. She is currently an Associate Professor with the Center for Applied Mathematics, Tianjin University, Tianjin, China. Her research interests include nonlinear signal processing, hyperspectral image processing and DNA storage.

BingZhi Li. Bingzhi Li is a professor with the School of Chemical Engineering and Technology, Tianjin University, China. He received the PhD degree from Tianjin University, Tianjin, China, in 2005. His current research interests include Synthetic Biology, Biochemistry and Molecular Biology and Pharmaceutical Engineering.

Yuping Duan. Yuping Duan is a full professor at the School of Mathematical Sciences of Beijing Normal University (BNU). Before joining BNU, she was a professor at Tianjin University in 2015 to 2023, and a research scientist at I2R, A*STAR in 2012 to 2015. She received her Ph.D. from Nanyang Technological University in 2012. Her research interests are image processing and computer vision, variational methods, deep learning methods, and DNA storage.