Improved read/write cost tradeoff in DNA-based data storage using LDPC codes

Shubham Chandak
Stanford University
Allerton 2019
Outline

- Motivation
- DNA storage setup
- Theoretical analysis
- Proposed framework
- Results
- Conclusions
Motivation
The amount of stored data is growing exponentially:

![Worldwide Byte Shipments by Storage Media Type](https://www.seagate.com/our-story/data-age-2025/)

Source: Data Age 2025, sponsored by Seagate with data from IDC Global DataSphere, Nov 2018

200 Petabyte
200 Petabyte

40,000 x 5 TByte HDDs
40 tons

10s of years
200 Petabyte

40,000 x 5 TByte HDDs
40 tons
10s of years

DNA
1 gram
1,000s of years
200 Petabyte

40,000 x 5 TByte HDDs
40 tons
10s of years

DNA
1 gram
1,000s of years
Easy duplication
Hot News for the Summer from CATALOG

CATALOG Encodes Wikipedia Into DNA!

https://catalogdna.com/uncategorized/hot-news-for-the-summer-from-catalog/
DNA storage setup
How to store data in DNA sequences?
How to store data in DNA sequences?

File

Segmentation

File
How to store data in DNA sequences?

File → Segmentation → Outer code → Inner code
How to store data in DNA sequences?

File → Segmentation → Outer code → Inner code

Also add index for recovering order of segments
How to store data in DNA sequences?

File → Segmentation → Outer code → Inner code → Synthesis → Storage

http://www.customarrayinc.com/
How to store data in DNA sequences?

1. **File**
   - Segmentation
   - Outer code
   - Inner code
   - Synthesis

2. **Storage**
   - Sequencing + Basecalling
   - Storage
   - Sequenced reads

- Duplication
- Permutation
- Loss
- Corruption
How to store data in DNA sequences?

1. File → Segmentation → Outer code → Inner code → Synthesis → Storage
2. Decoding: Duplication, Permutation, Loss, Corruption
3. Reconstructed file
4. Sequencing + Basecalling → Sequenced reads
How to store data in DNA sequences?

- Separate codes for erasure and error correction
- Heavy reliance on “consensus”
Previous works

- Multiple previous works focusing on:
  - Error correction coding
  - Random access to subsets of synthesized sequences using PCR primers
  - Scalable and cost effective synthesis techniques
  - Different sequencing platforms
  - Theoretical analysis

Theoretical analysis
Read-write cost tradeoff

- Fundamental quantities from a coding theory perspective:
  - Writing cost (bases synthesized/message bit)
  - Reading cost (bases sequenced/message bit)
  - Note: “Coverage” (= bases sequenced/bases synthesized) doesn’t capture the actual reading cost.
Read-write cost tradeoff

- Fundamental quantities from a coding theory perspective:
  - Writing cost (bases synthesized/message bit)
  - Reading cost (bases sequenced/message bit)
  - Note: “Coverage” (= bases sequenced/bases synthesized) doesn’t capture the actual reading cost.

- Fixed sequence length means asymptotic information capacity = 0!
Read-write cost tradeoff

● Fundamental quantities from a coding theory perspective:
  ○ Writing cost (bases synthesized/message bit)
  ○ Reading cost (bases sequenced/message bit)
  ○ Note: “Coverage” (= bases sequenced/bases synthesized) doesn’t capture the actual reading cost.

● Fixed sequence length means asymptotic information capacity = 0!
  ○ Previous works assumed sequence length growing logarithmically in number of sequences
  ○ Does not capture the limitations posed by short sequence length
Simplified model for analysis

- **$nL$ information bits**
- **Encoding**
  - $nc_w$ sequences with $L$ bits each
- **BSC($\epsilon$)**
  - $nc_r$ "reads" sampled with replacement
- **Noisy reads**
Use a memoryless approximation and obtain asymptotically achievable tradeoff between $c_w$ and $c_r$. 

Simplified model for analysis

$nL$ information bits → Encoding → $nc_w$ sequences with $L$ bits each → BSC(ε) → $nc_r$ “reads” sampled with replacement → Noisy reads
Two strategies

Strategy 1: Inner/outer code separation

Strategy 2: Single large block code
Simulation results

- Separation strategy simulation: 0.5% error
- Large block code strategy simulation: 0.5% error
- Theoretically achievable bound: 0.5% error
- Theoretically optimal tradeoff: 0% error
Proposed framework
Proposed approach

Encoding

Binary file ➔ Large block LDPC encoding ➔ Segment and map to DNA ➔ Add sync marker (AGT) ➔ Attach BCH-protected index
Proposed approach

Encoding

<table>
<thead>
<tr>
<th>Index</th>
<th>BCH</th>
<th>Payload</th>
<th>AGT</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>~ 10 bp</td>
<td>~ 6 bp</td>
<td>~ 84 bp</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Proposed approach

**Encoding**

<table>
<thead>
<tr>
<th>Index</th>
<th>BCH</th>
<th>Payload</th>
<th>AGT</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>~ 10 bp</td>
<td>~ 6 bp</td>
<td>~ 84 bp</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Decoding**

- Reads
- Decode index using BCH
- Per-index MSA & consensus
- Recover partial payload using sync markers if consensus length incorrect
- LDPC decoding based on counts of A/C/G/T at each position
- Binary file
Results
Experimental Parameters

- Multiple parameter experiments, storing around 200 KB data each.

- CustomArray synthesis, length 150 including primers.

- Sequenced with Illumina iSeq.

- Total error rate around 1.3% (substitution: 0.4%, deletion: 0.85%, insertion: 0.05%) – cheaper and noisier synthesis as compared to previous works.
Experimental Results

Experimental Results


What happened in experiments 2 and 5?
Coverage variation

![Coverage variation graph](image)
Experimental Results

Higher redundancy codes much more robust!
Experimental Results

Higher redundancy codes much more robust!

More analysis in paper
Conclusions

- Introduced novel coding schemes for Illumina sequencing based DNA storage
  - Improved read/write cost tradeoff despite noisier synthesis

- Code and data: https://github.com/shubhamchandak94/LDPC_DNA_storage

- Biorxiv: https://www.biorxiv.org/content/10.1101/770032v1
Future work

- Possibilities for improvement:
  - Optimized LDPC codes, e.g., using protographs
  - Better codes for insertion/deletion: LDPC with markers, VT codes
  - Check out q-ary VT codes implementation: https://github.com/shubhamchandak94/VT_codes/
Future work

- Possibilities for improvement:
  - Optimized LDPC codes, e.g., using protographs
  - Better codes for insertion/deletion: LDPC with markers, VT codes
  - Check out q-ary VT codes implementation: [https://github.com/shubhamchandak94/VT_codes/](https://github.com/shubhamchandak94/VT_codes/)

- Plan to integrate these with random access and repeated reading.
Future work

- Possibilities for improvement:
  - Optimized LDPC codes, e.g., using protographs
  - Better codes for insertion/deletion: LDPC with markers, VT codes
  - Check out q-ary VT codes implementation: https://github.com/shubhamchandak94/VT_codes/

- Plan to integrate these with random access and repeated reading.
- Long term vision: Nanopore sequencing + cheaper and noisier synthesis techniques
Team and funding

SemiSynBio: Highly scalable random access DNA data storage with nanopore-based reading

Beckman Center Innovative Technology Seed Grant
Scalable Long-Term DNA Storage with Error Correction and Random-Access Retrieval

NSF
SRC
NIH

Shubham Chandak
Kedar Tatwawadi
Joachim Neu
Jay Mardia
Billy Lau
Matt Kubit
Peter Griffin

Tsachy Weissman
Mary Wootters
Hanlee Ji
Copyright 1878, by MUYBRIDGE.

THE HORSE IN MOTION.

MUYBRIDGE, AUTOMATIC ELECTRO-PHOTOGRAPHER.

"SALLIE GARDNER," owned by LELAND STANFORD; ridden by O. DOMM, running at a 4.40 gait over the Polo Aloe track, 19th June, 1878.

The negatives of these photographs were made at intervals of twenty-seventh of an inch, and about the twenty-fifth part of a second of time, they illustrate consecutive positions assumed during a single cycle of the race. The vertical lines were twenty-seventh of an inch, the horizontal lines represent elevation of one inch each.

The negatives were each exposed during the twelfth of a second, and are absolutely "stopped."
Thank You!

Biorxiv: https://www.biorxiv.org/content/10.1101/770032v1
Backup
We first compute the optimal tradeoff between $c_w$ and $c_r$ when $\epsilon = 0$, i.e., the reads are error-free. In this case, for large enough $n$, we can use the Poisson($\lambda$) approximation for the number of times each sequence is observed with $\lambda = c_r / c_w$. Since the probability of seeing zero copies of a sequence is $e^{-\lambda}$, this gives us an erasure channel with capacity $1 - e^{-\lambda}$ [20]. For reliable recovery, we need that the rate $1/c_w$ be less than the capacity. This gives us

$$c_r \geq c_w \log_e \frac{c_w}{c_w - 1}$$

$$P((k_0, k_1) \mid 0) = \frac{e^{-\lambda} \lambda^{k_0+k_1}}{(k_0 + k_1)!} \binom{k_0 + k_1}{k_0} (1 - \epsilon)^{k_0} \epsilon^{k_1}$$

$$LLR(k_0, k_1) = \ln \frac{P((k_0, k_1) \mid 0)}{P((k_0, k_1) \mid 1)} = (k_0 - k_1) \ln \frac{1 - \epsilon}{\epsilon}$$