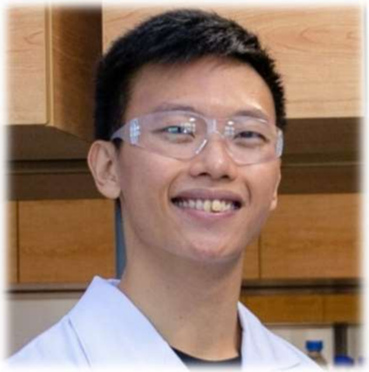
Puru Sharma



Cheng-Kai Lim

Dehui Lin

Efficiently Enabling Block Semantics and Data Updates in DNA Storage

Yash Pote

National University of Singapore

Djordje Jevdjic

Why storing data in DNA molecules?

1. Incredible density
   - 6-7 orders of magnitude ahead of best alternatives!
2. Unmatched durability
   - Thousands/millions/billions of years (vs. 3-5 years for disks/flash)
3. Never obsolete: R/W interfaces will only improve with time
4. Efficient random access
5. Convenient for many data-parallel & near-data computations

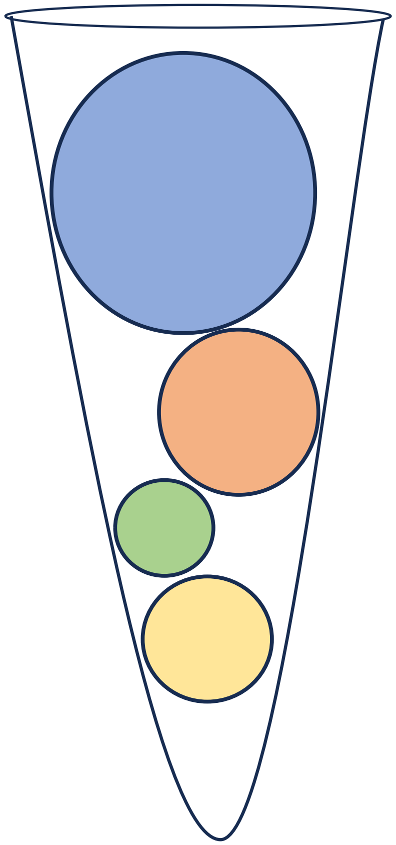
Key Problems with DNA Storage

1. Expensive R/W interfaces
   - Writing cost: $1K - $10K/MiB
   - Reading cost: $10 - $10K/MiB

   → Architectures to minimize the amount of data read/written

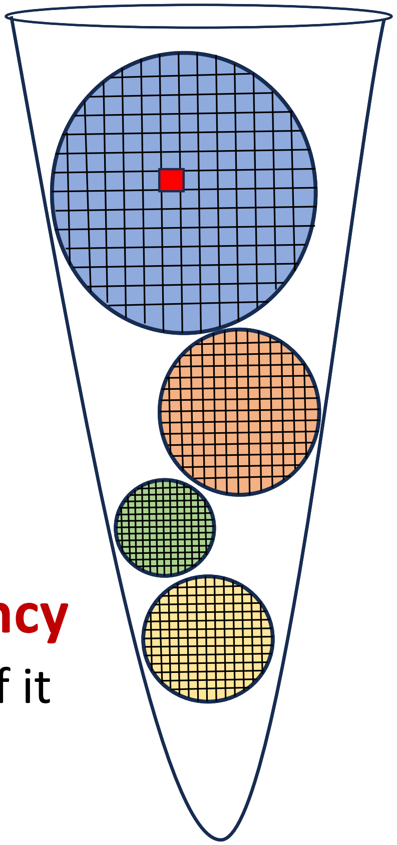
2. Limited number of addresses per test-tube
   - Only ~3000 unique objects can be retrieved at random
   - Key reason: arbitrary size of objects

3. No Practical Update Mechanism
   - Impractical to “edit” existing molecules



Our Proposal: Block-Based Architecture

- Enables ~3000 ~~objects~~ **partitions** of arbitrary size in a tube
  - Any whole partition can be retrieved at random
- Each partition internally blocked into **fixed-size** units
  - Fixed size allows for **millions** of blocks within each partition
  - Each block can be individually retrieved and written to at random
- Orders of magnitude reduction in **read/write cost** and **latency**
  - Instead of a giant partition, we can retrieve/update a small part of it

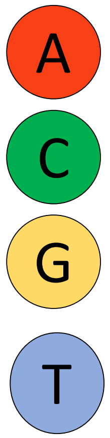


Outline

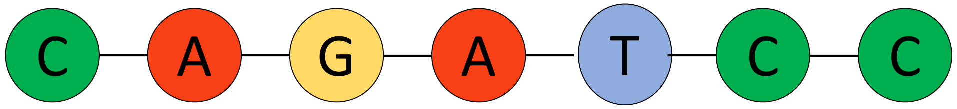
- Introduction
- **DNA Storage Basics**
- Limitations of Object Store semantics
- Block Semantics
- Data Updates
- Evaluation
- Conclusion

DNA Molecules

4 nucleotides



Synthetic DNA molecule



- Artificially created string of nucleotides
- No biological meaning

$log_2|\{A, C, G, T\}|$ = 2 bits of data per nucleotide

Storing Data in short DNA strings

1011000100101101… $\xrightarrow{\text{encoding}}$ GTACAGTC… $\xrightarrow{\text{synthesis}}$

Problem: **Artificial DNA molecules limited in length!**

- Practical length: a few hundred nucleotides
- Solution: split big data into smaller **ordered** chunks! [Bornholt et al, ASPLOS'16]

00 ↔ A

01 ↔ C

10 ↔ G

11 ↔ T

1011000100101101… $\xrightarrow{\text{encoding}}$

AGTAC

CAGTC

GCGTA

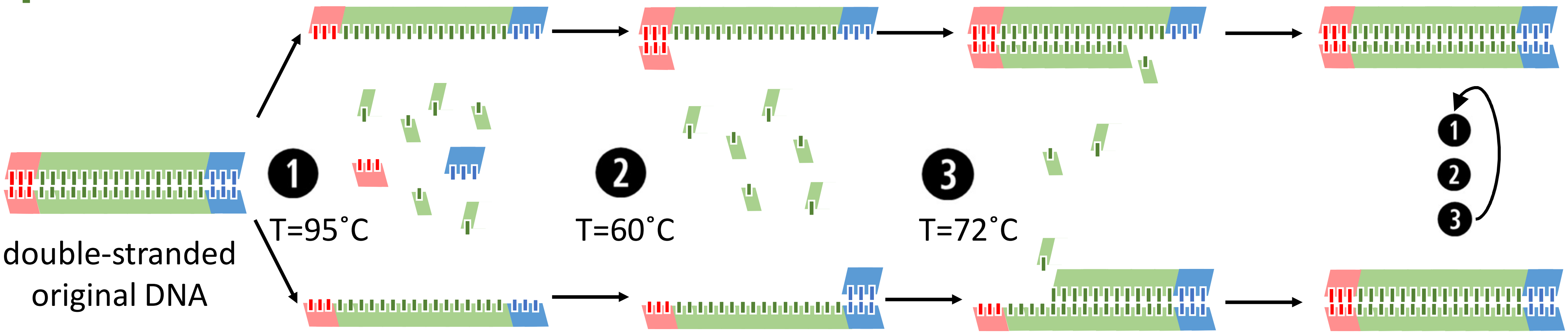
TAAGC → ordering metadata (index)

How to retrieve the entire object at random?

Polymerase Chain Reaction (PCR)

primers: short DNA sequences identifying regions of interest

free DNA bases



T=95˚C

T=60˚C

T=72˚C

double-stranded original DNA

Every PCR cycle doubles the number of DNA molecules between the primers
→ exponential replication

Random Access using PCR*

primers

GAC AA ACGAGGATTCAACCTCG
GAC AC ACCGAGGATTCAACTCG
GAC AG CACACGGGGCCTTATCG
GAC AT AAATCGGTTACCGGTCG
GAC CA TACCATGACGAAGCTCG
GAC CC GATTCAACACGAGTTCG
GAC CG CTTAGGACTAATCG TCG
GAC CT ACAATTGAAGCTAGTCG

object #1

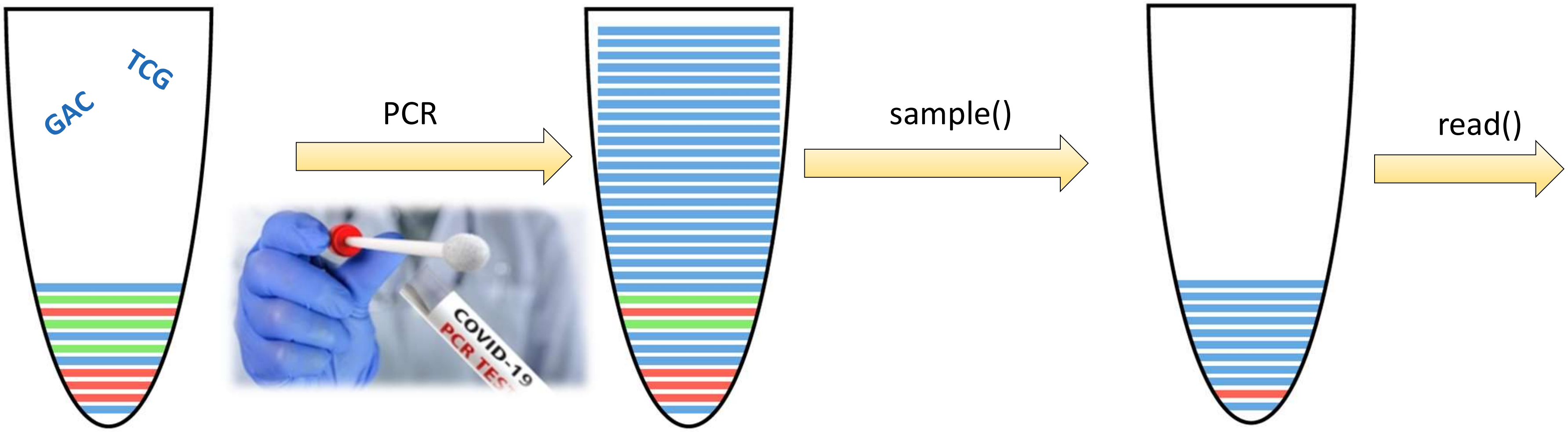
CTT A GACCAGGATTCGT AGG
CTT C CGATTCGATCGAC AGG

object #2

TAC A AGCTTCGATTCGG GTA
TAC C ATCGATCGTGCTA GTA
TAC G CGTAATCGGACTC GTA
TAC T GATCGGCTATTCC GTA

object #3

index



*Bornholt et al, ASPLOS’16

Primer Constraints

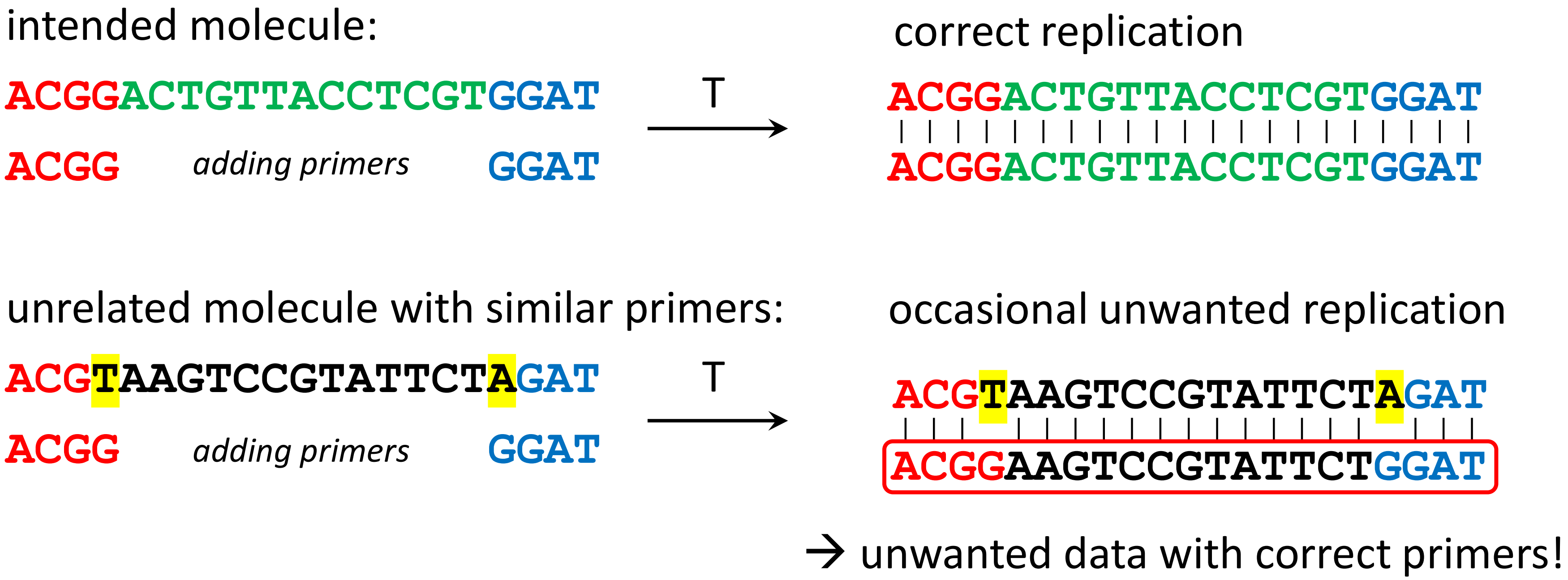
Typical primer length is 20 → $4^{20} = 2^{40}$ possible primers

Unfortunately, primers have strict constraints:

1. Balanced GC-content: **`#G + #C == #A + #T`**
2. Max homopolymer length of 4: `ACGTAG`$\underbrace{\texttt{TTTTTT}}_{\text{homopolymers}}$`ACG`
3. Minimum pairwise edit distance of 8
   - To avoid replication of unrelated data (a.k.a. ***mispriming***)
   - Significantly reduces the size of the primer set!

Largest primer library contains only ~6000 primers → 3000 objects

PCR *Mispriming* – replication of unwanted data



intended molecule:

```
ACGGACTGTTACCTCGTGGAT
ACGG    adding primers    GGAT
```

T →

correct replication

```
ACGGACTGTTACCTCGTGGAT
|||||||||||||||||||||
ACGGACTGTTACCTCGTGGAT
```

unrelated molecule with similar primers:

```
ACGTAAGTCCGTATTCTAGAT
ACGG    adding primers    GGAT
```

T →

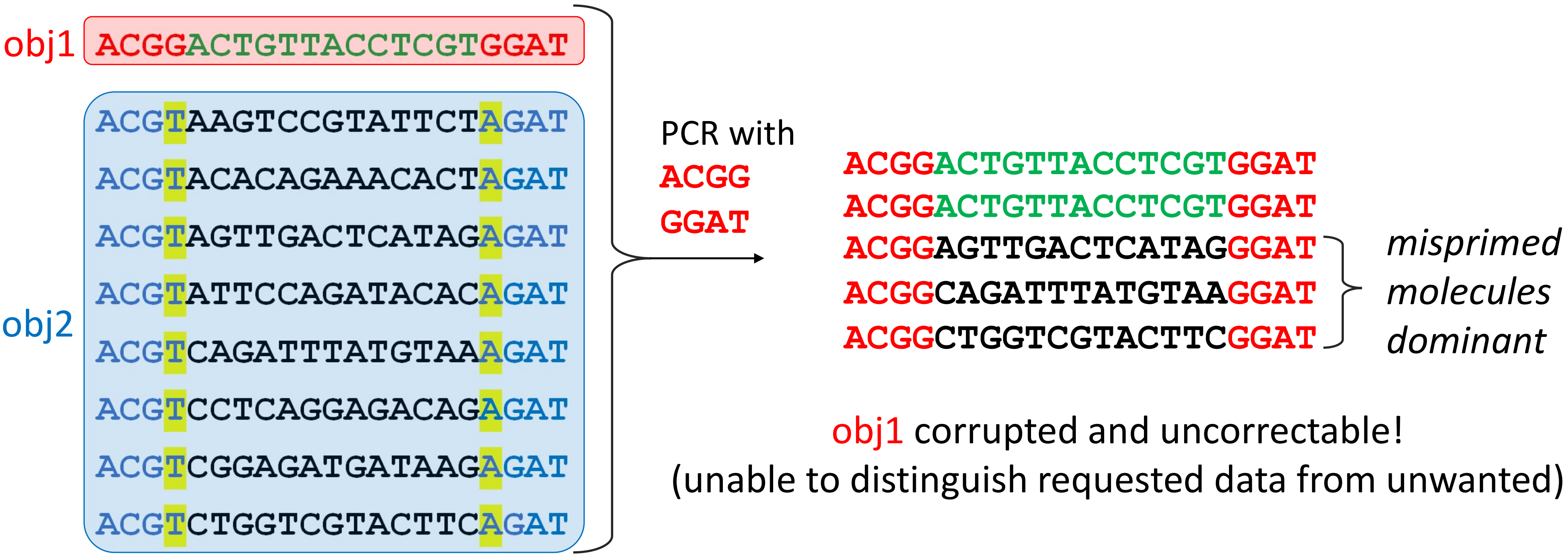
occasional unwanted replication

```
ACGTAAGTCCGTATTCTAGAT
||| ||||||||||||| |||
ACGGAAGTCCGTATTCTGGAT
```

→ unwanted data with correct primers!

***Misprimed* molecules can be exponentially replicated**

Mispriming and Irregular Object Sizes



obj1 ACGGACTGTTACCTCGTGGAT

obj2

ACGTAAGTCCGTATTCTAGAT
ACGTACACAGAAACACTAGAT
ACGTAGTTGACTCATAGAGAT
ACGTATTCCAGATACACAGAT
ACGTCAGATTTATGTAAAGAT
ACGTCCTCAGGAGACAGAGAT
ACGTCGGAGATGATAAGAGAT
ACGTCTGGTCGTACTTCAGAT

PCR with
ACGG
GGAT

ACGGACTGTTACCTCGTGGAT
ACGGACTGTTACCTCGTGGAT
ACGGAGTTGACTCATAGGGAT
ACGGCAGATTTATGTAAGGAT
ACGGCTGGTCGTACTTCGGAT

misprimed molecules dominant

obj1 corrupted and uncorrectable!
(unable to distinguish requested data from unwanted)

Maximum extent of *mispriming* uncontrollable due to arbitrary object sizes

Key Insights

Arbitrary object size causes severe problems:

- Mispriming must be avoided at all cost
  - Else, it can spiral out of control due to arbitrary object sizes

→ primers maintain high pairwise distance

→ unacceptably small set of viable primers

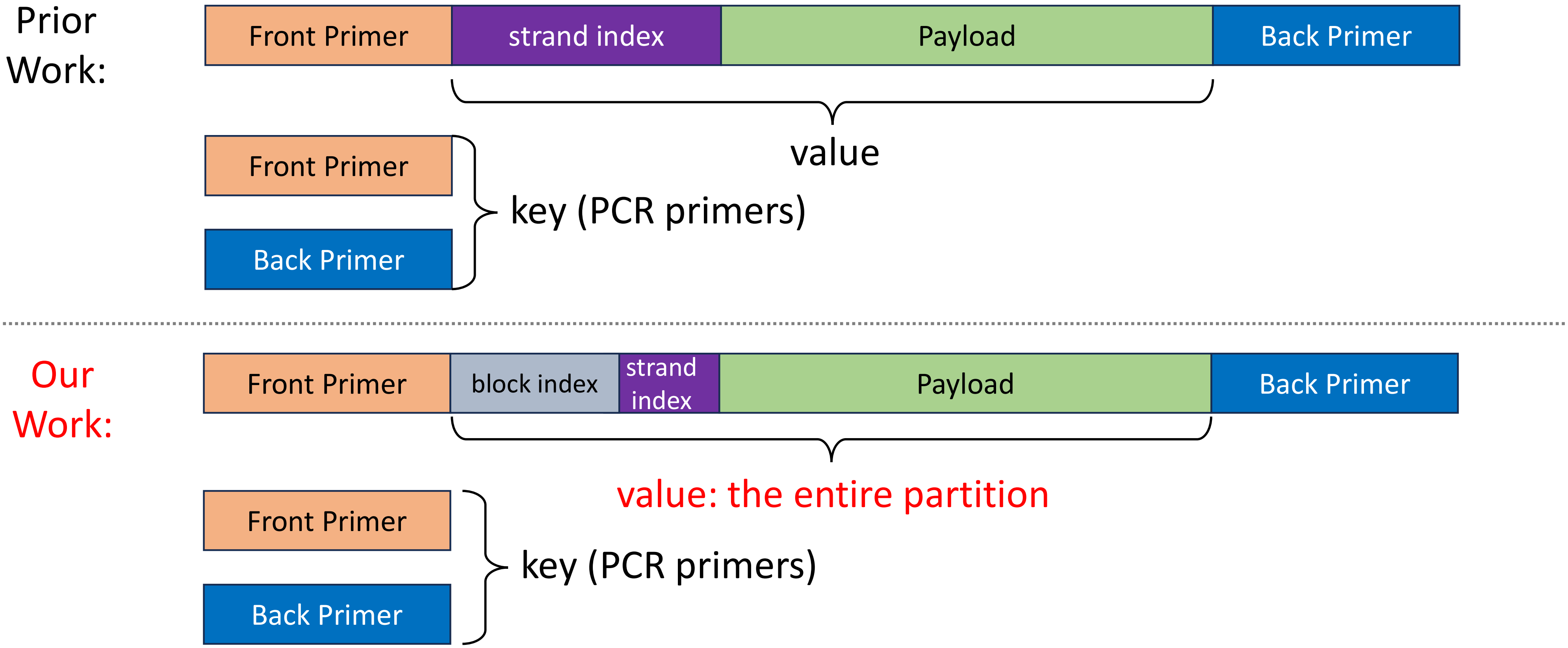
Key idea:

- Maintain uniform object sizes to allow for controllable amount of mispriming
  - Limited mispriming can be dealt with through **error correction**
- Relax the distance requirement → significantly increase the number of primers

Outline

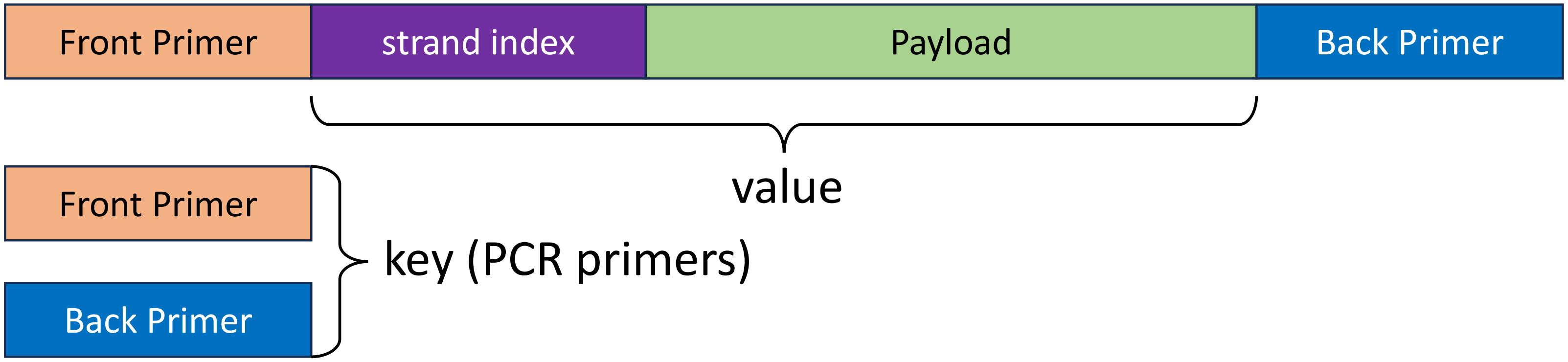
- Introduction
- DNA Storage Basics
- Limitations of Object Store semantics
- **Block Semantics**
- Data Updates
- Evaluation
- Conclusion

Our Proposal: Block-Based DNA Storage



Our Proposal: Block-Based DNA Storage

Prior Work:



Front Primer | strand index | Payload | Back Primer

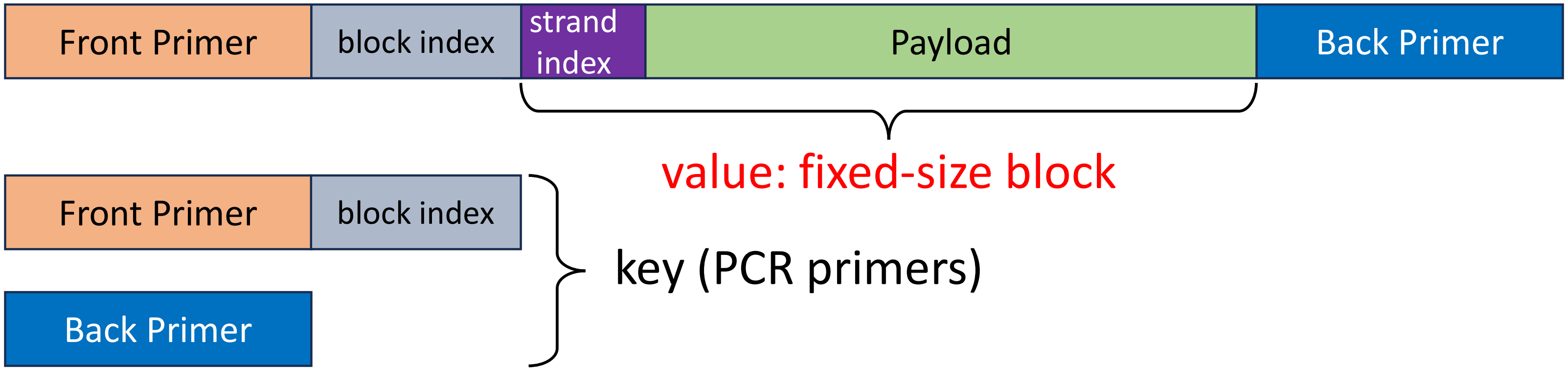
value

Front Primer

Back Primer

key (PCR primers)

Our Work:



Front Primer | block index | strand index | Payload | Back Primer

value: fixed-size block

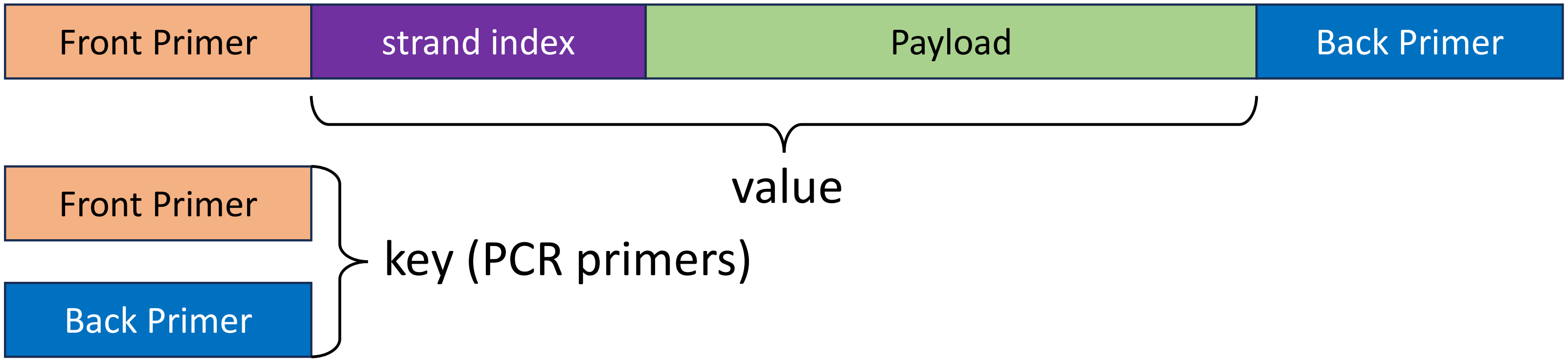
Front Primer | block index

Back Primer

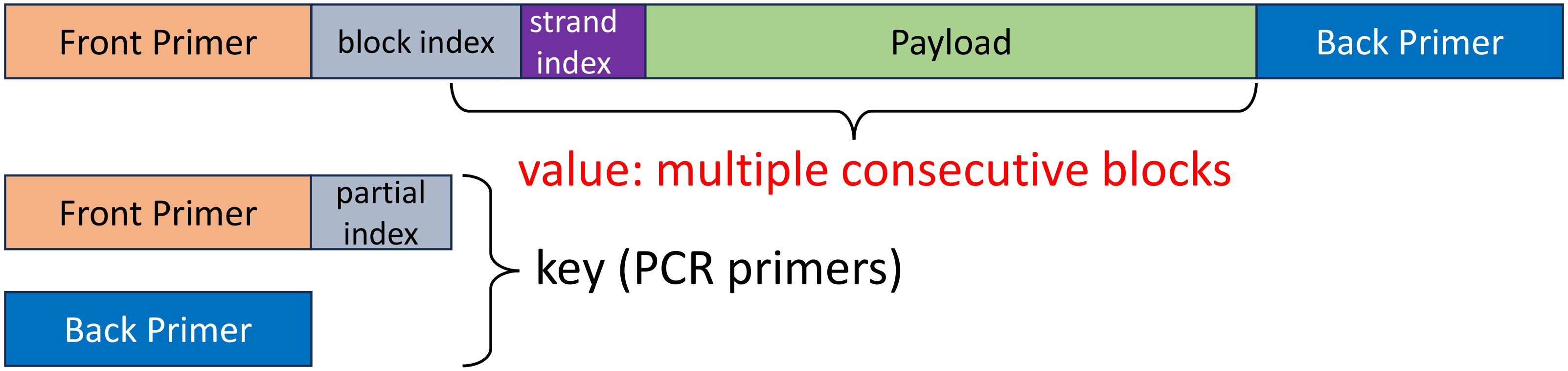
key (PCR primers)

Sequential Access with Partially-Elongated Primers

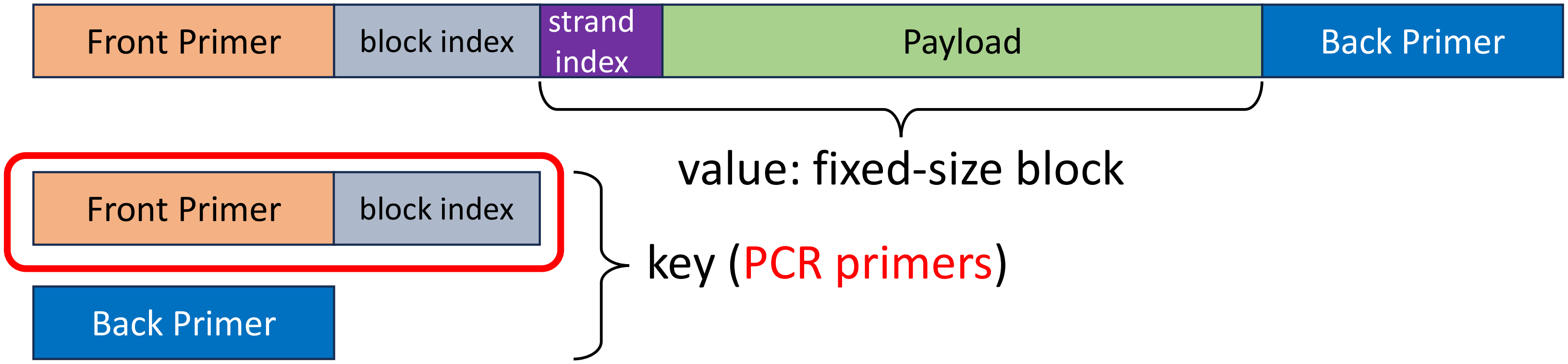
Prior Work:



Our Work:



PCR with Elongated Primers



Front Primer | block index | strand index | Payload | Back Primer

value: fixed-size block

Front Primer | block index

Back Primer

key (PCR primers)

All possible elongated primers must comply with primer constraints!

However, for block 0 (AAAAAA):

Front Primer | AAAAAA

→ Too many homopolymers

→ GC content not balanced

Block Indexes need PCR-compatible Encoding

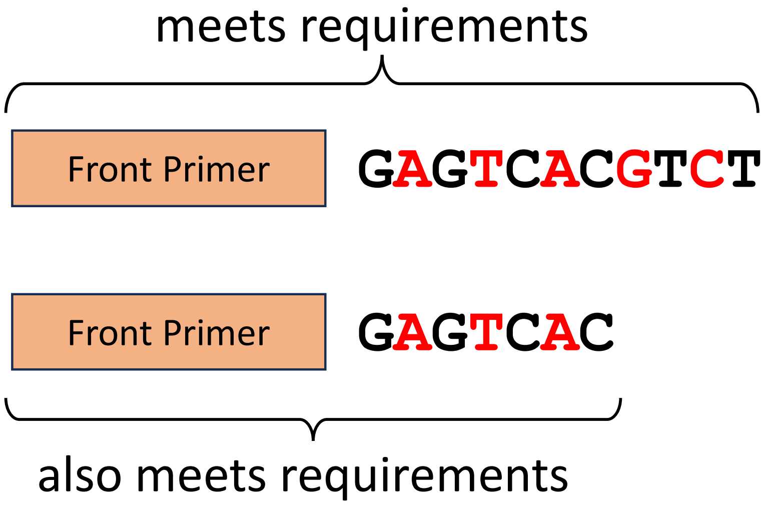
Sparse Encoding of Block Indexes

Add a suitable **padding** base between neighboring index bases

- In a manner that satisfies the constraints

AAAAAA → AGACACAGAGA

GGCCTT → GAGTCACGTCT



meets requirements

Front Primer **GAGTCACGTCT**

Front Primer **GAGTCAC**

also meets requirements

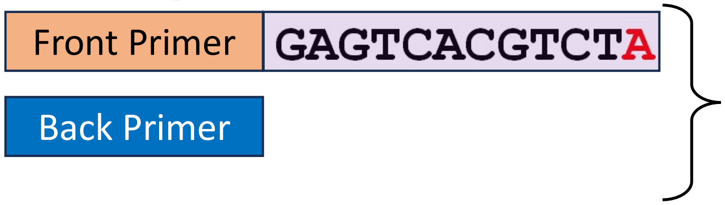
All possible elongations, including the partial ones, satisfy the PCR constraints

Outline

- Introduction
- DNA Storage Basics
- Limitations of Object Store semantics
- Block Semantics
- **Data Updates**
- Evaluation
- Conclusion

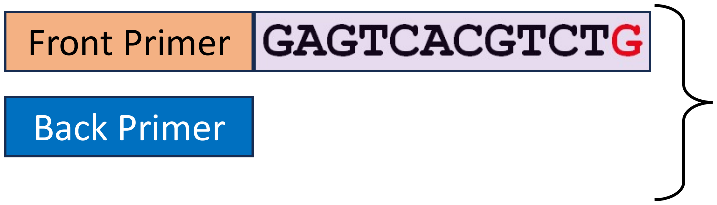
Practical Data Updates

Task: update block #42 in partition X?



Key for original block #42

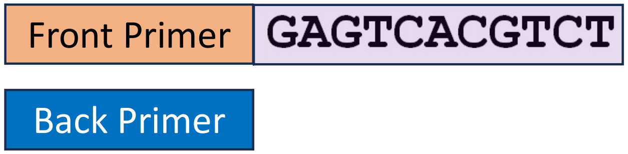
Create an encoded DNA patch for block #42.



Key for block #42 update patch

“persist” the update

Read block #42 and all its updates (and nothing else)



Decode and apply the patch in software, avoiding molecular edits

Evaluation Methodology

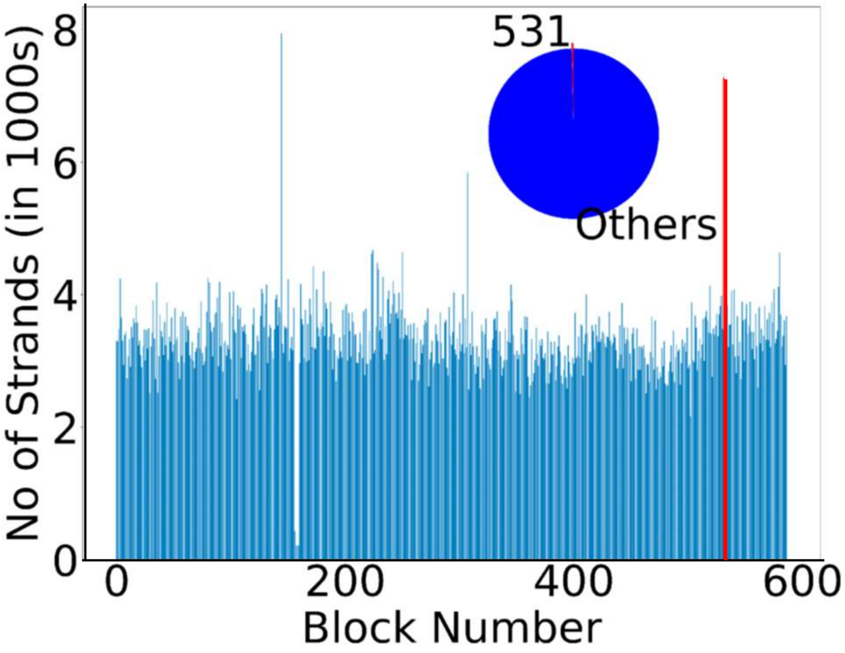
Synthesized ~12.000 DNA strands as 13 partitions

- One big partition (9000 strands): “Alice in Wonderland” book in plaintext
  - Organized in 1024 blocks, 256B each
  - 15 DNA strands/block, 4 of which are Reed-Solomon ECC
- 6 DNA update patches created for 6 blocks chosen at random
  - contain textual edits
  - encoded as a *diff* rather than the entire replacement block
  - ”persisted” by careful mixing with the original

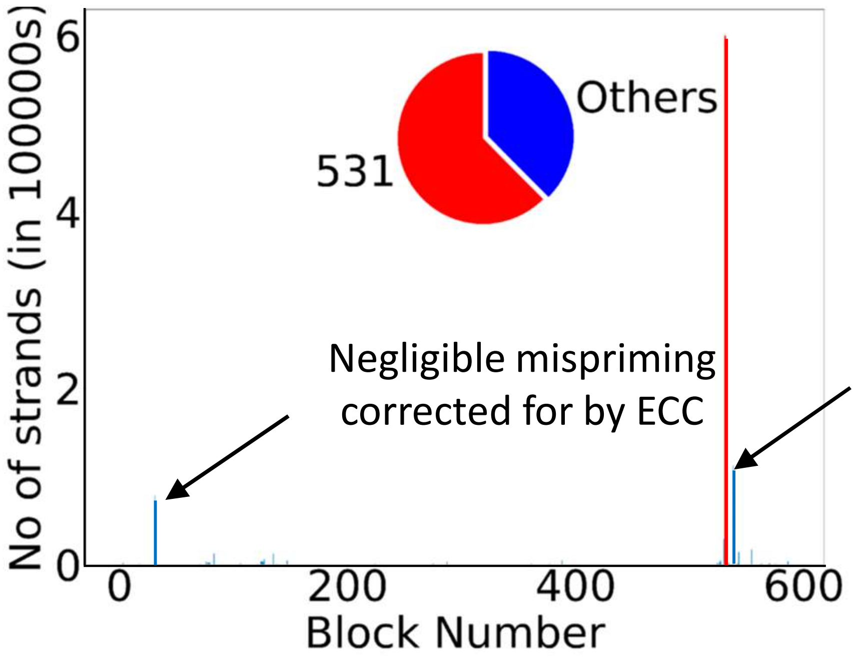
Experiment: retrieve an updated block using PCR with elongated primers

- Compare against the retrieval of the entire partition (conventional primers)

Result Highlights: Retrieving Block #531



reading the entire partition:
>99% unwanted data



Negligible mispriming
corrected for by ECC

reading the target block:
target data dominant

140x reduction in reading cost

Conclusions

- Arbitrary object size significantly reduces the number of addresses
    - Uniform object size can relax the addressing restrictions

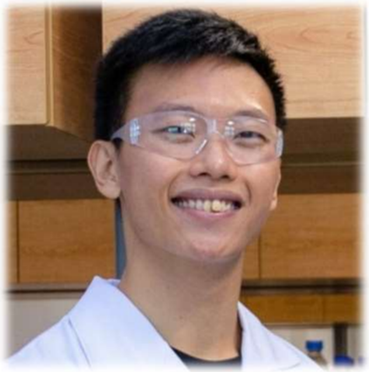
- Block-based architecture with elongated primers
    - 1024x more addresses within every partition
    - Convenient log-based data updates
    - Enables future DNA Storage File Systems

- Wetlab experiments: 140x reduction in sequencing cost (and latency)

- Check out the paper for more details and results:

Full Paper

Puru Sharma



Cheng-Kai Lim

Dehui Lin

Thank you! Questions?

Yash Pote

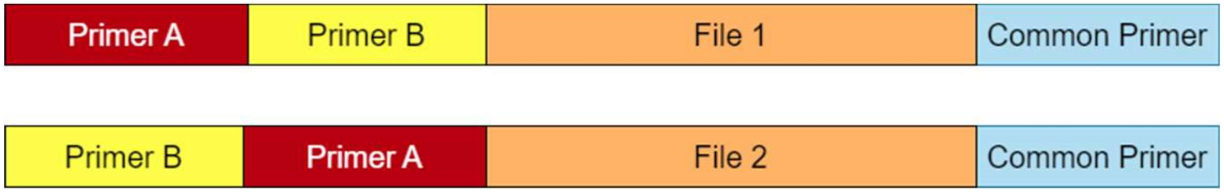
NUS National University of Singapore

Djordje Jevdjic

Backup Slides

Prior Work

- Nested Primers [1]



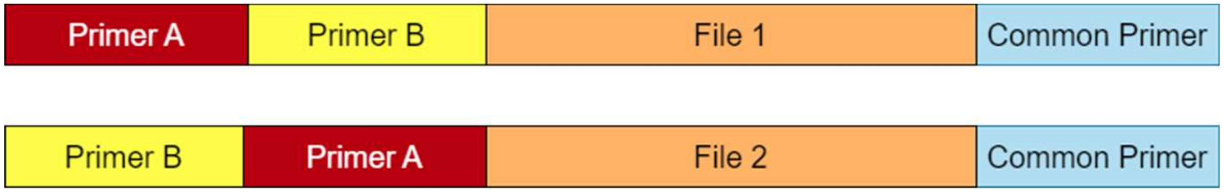
Primer A | Primer B | File 1 | Common Primer

Primer B | Primer A | File 2 | Common Primer

[1] Tomek, Kyle J., et al. "Driving the scalability of DNA-based information storage systems." ACS synthetic biology 8.6 (2019)

Prior Work

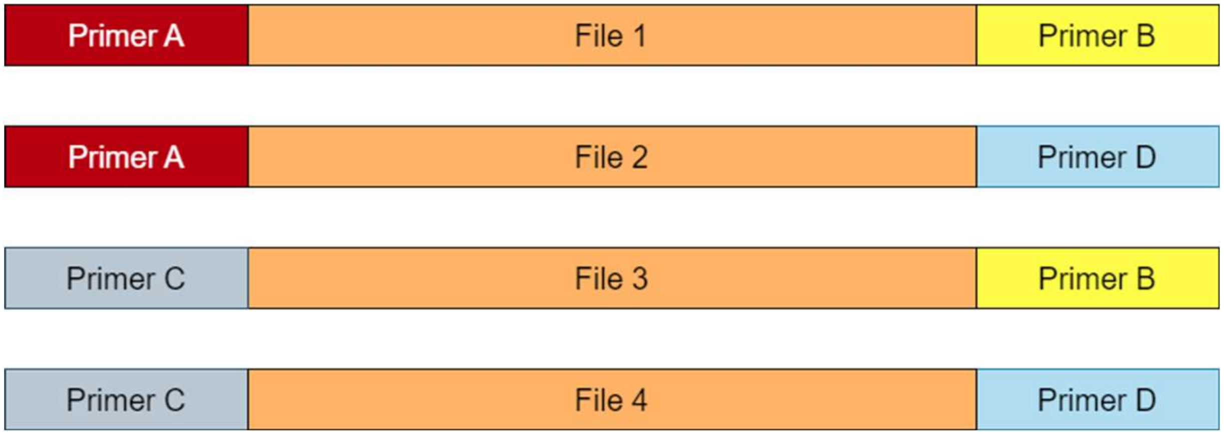
- Nested Primers [1]



Primer A | Primer B | File 1 | Common Primer

Primer B | Primer A | File 2 | Common Primer

- Combinatorial PCR [2]



Primer A | File 1 | Primer B

Primer A | File 2 | Primer D

Primer C | File 3 | Primer B

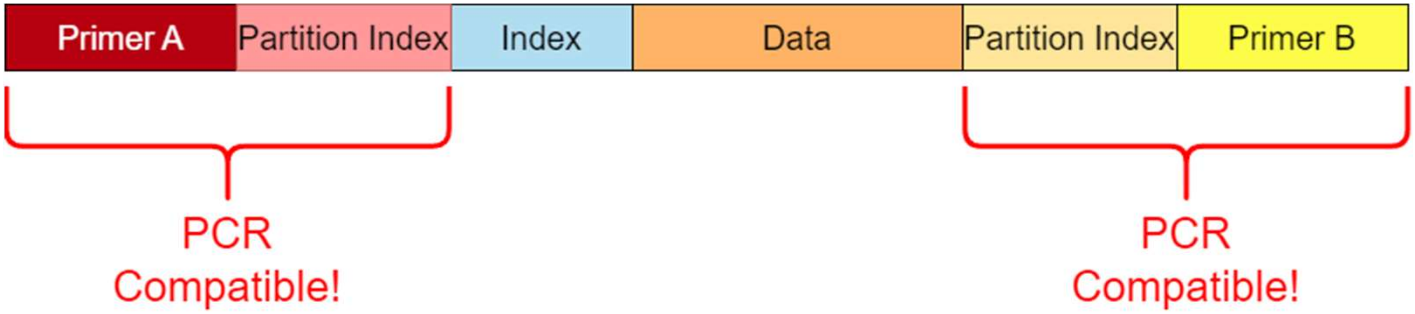
Primer C | File 4 | Primer D

[1] Tomek, Kyle J., et al. "Driving the scalability of DNA-based information storage systems." ACS synthetic biology 8.6 (2019)

[2] Winston, Claris, et al. "Combinatorial PCR method for efficient, selective oligo retrieval from complex oligo pools." ACS Synthetic Biology 11.5 (2022)

Future Work

- Study limitations of our PCR
- Increase number of partitions further
  - Extend both forward and reverse primers


Primer A
Partition Index
Index
Data
Partition Index
Primer B
PCR Compatible!
PCR Compatible!