

A DNA-Based Archival Storage System

James Bornholt^{*}

Randolph Lopez^{*}

Douglas M. Carmean[†]

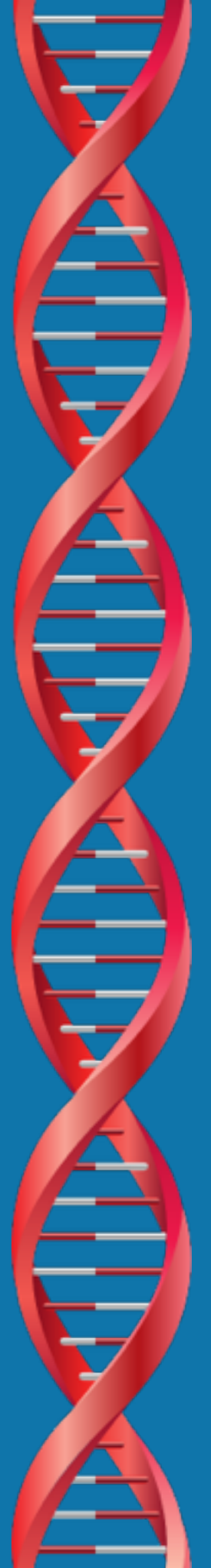
Luis Ceze^{*}

Georg Seelig^{*}

Karin Strauss[†]

^{*} University of Washington

[†] Microsoft Research



Facebook cold storage facility
1 exabyte (10⁹ GB)
66,000 square feet



DNA molecules as storage



DNA molecules as storage

Extremely dense
Theory: 1 exabyte in 1 in³



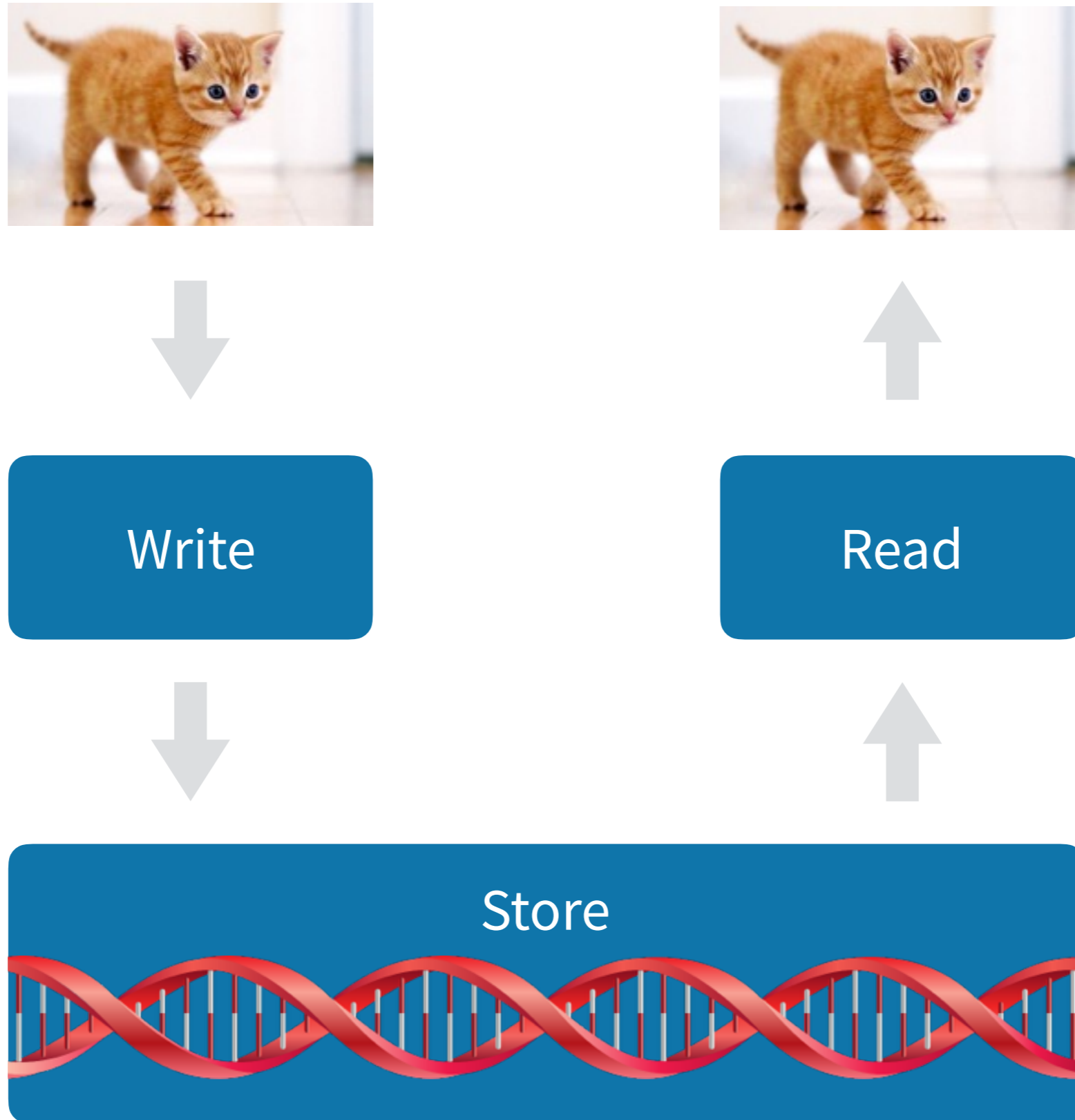
DNA molecules as storage

Extremely dense
Theory: 1 exabyte in 1 in³



Extremely durable
Half life > 500 years

A DNA-based archival storage system



A DNA-based archival storage system



Redundancy
and density



Write

Read



Store



A DNA-based archival storage system



Redundancy
and density



Write



Efficient
retrieval

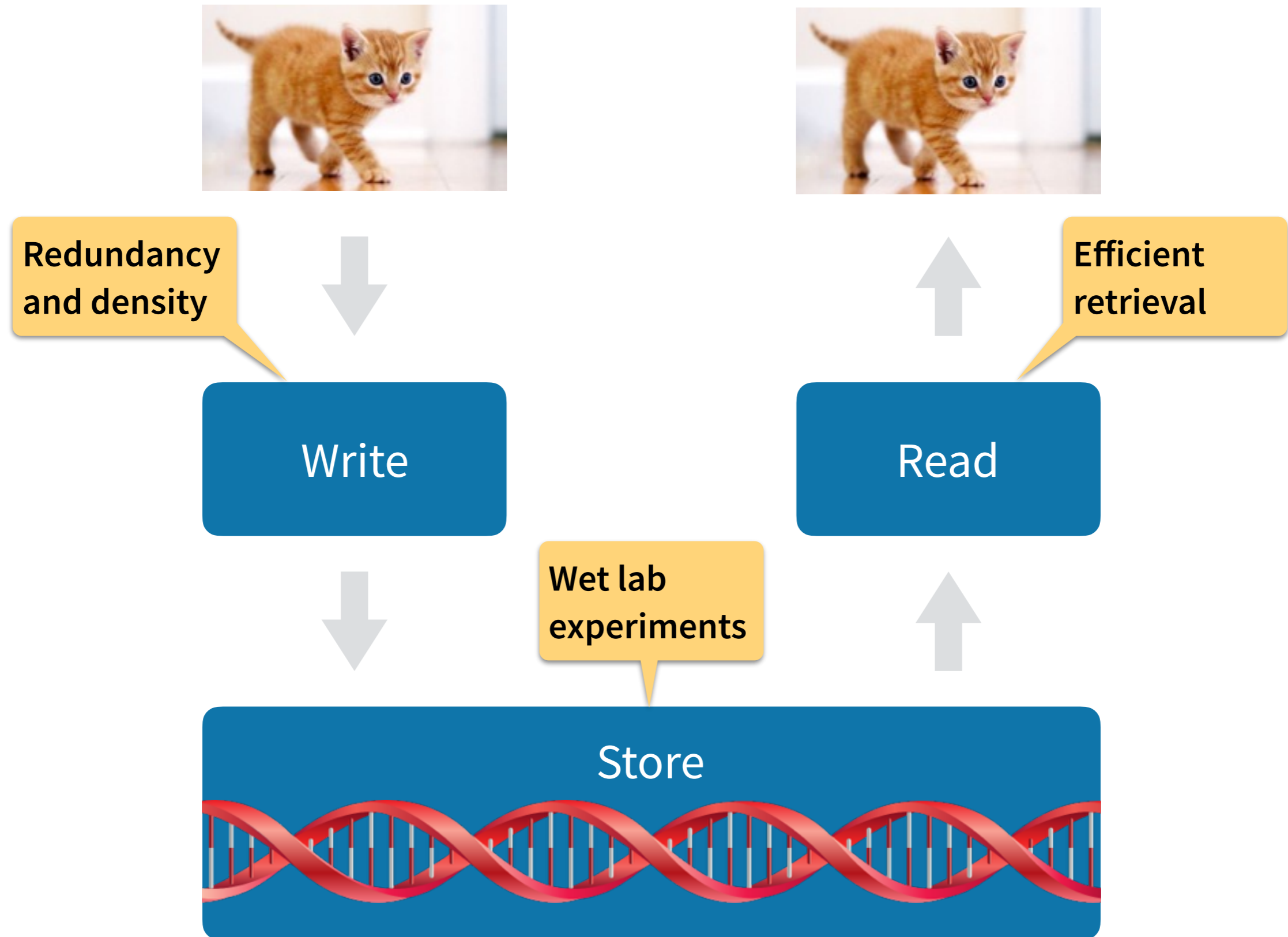
Read



Store



A DNA-based archival storage system



DNA manipulation



DNA molecules

Four nucleotides:

 Adenine

 Cytosine

 Guanine

 Thymine

DNA molecules

Four nucleotides:

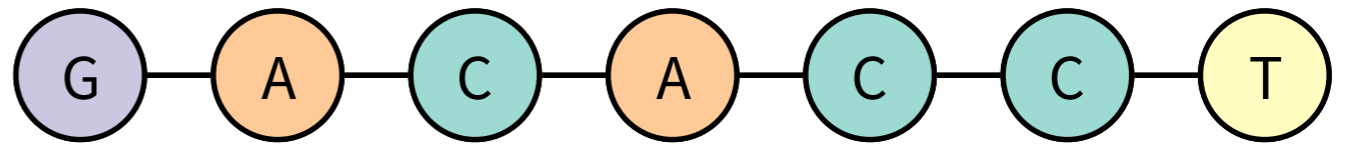
 Adenine

 Cytosine

 Guanine

 Thymine

DNA strand (oligonucleotide) is a linear sequence of these nucleotides



DNA molecules

Four nucleotides:

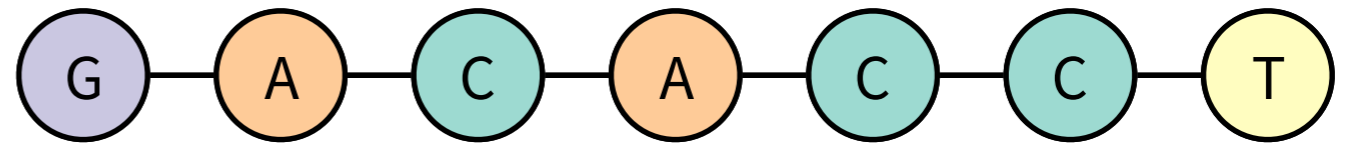
 Adenine

 Cytosine

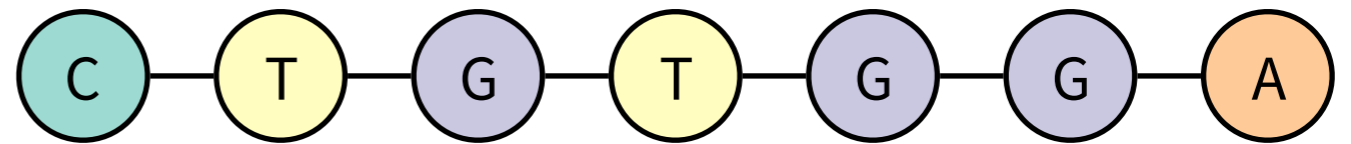
 Guanine

 Thymine

DNA strand (oligonucleotide) is a linear sequence of these nucleotides



Two strands can bind to each other if they are complementary:



DNA molecules

Four nucleotides:

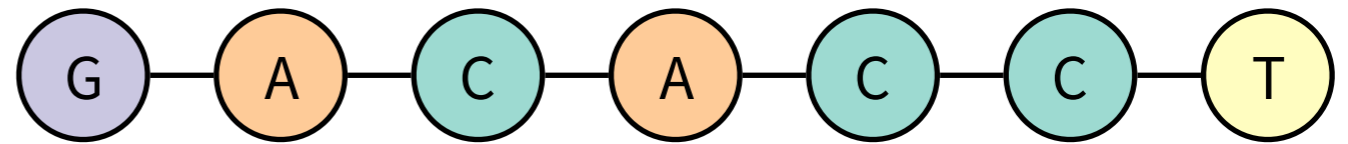
 Adenine

 Cytosine

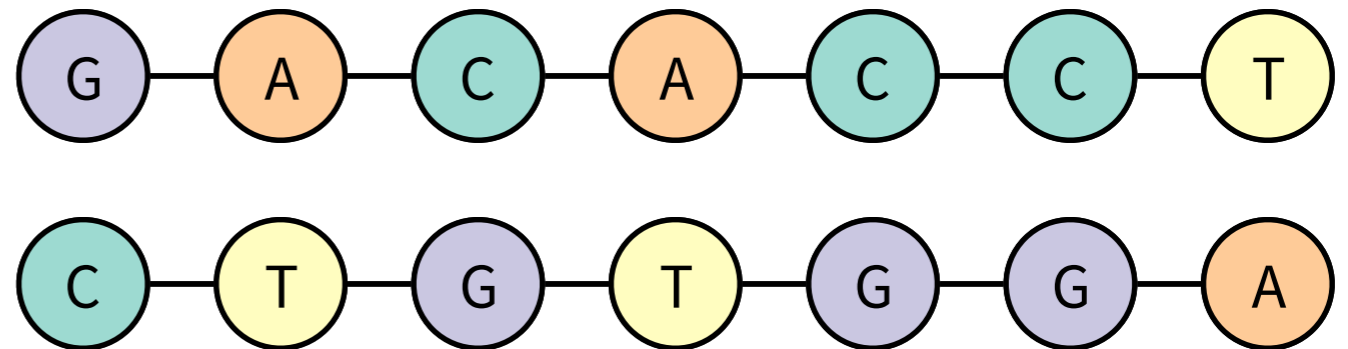
 Guanine

 Thymine

DNA strand (oligonucleotide) is a linear sequence of these nucleotides

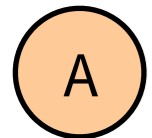
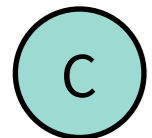
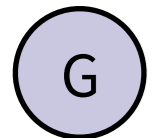
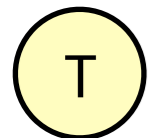


Two strands can bind to each other if they are complementary:

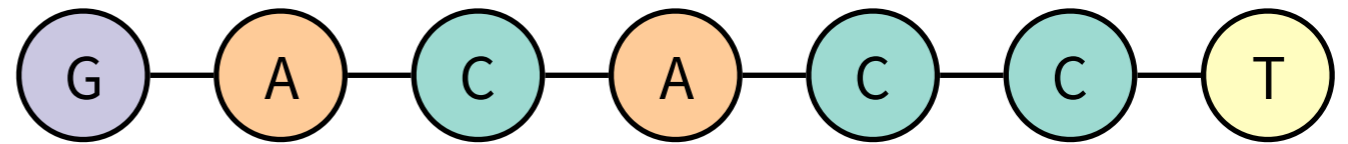


DNA molecules

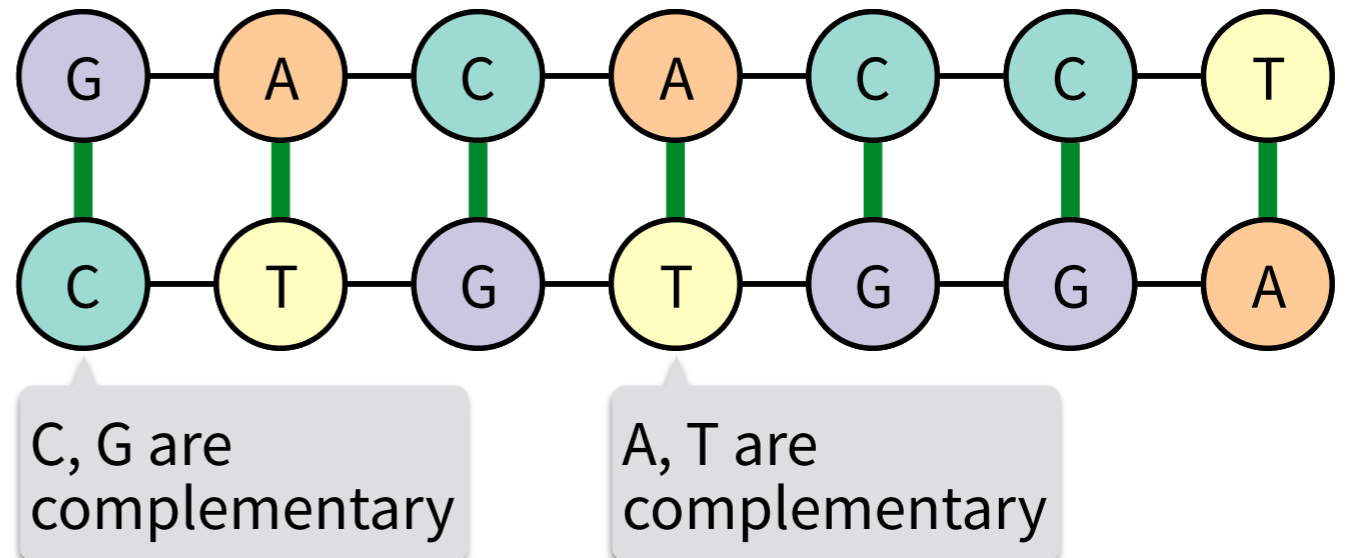
Four nucleotides:

-  Adenine
-  Cytosine
-  Guanine
-  Thymine

DNA strand (oligonucleotide) is a linear sequence of these nucleotides

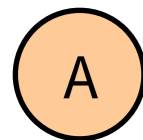
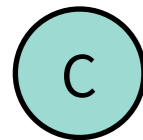
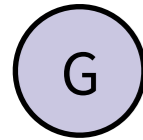
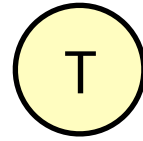


Two strands can bind to each other if they are complementary:

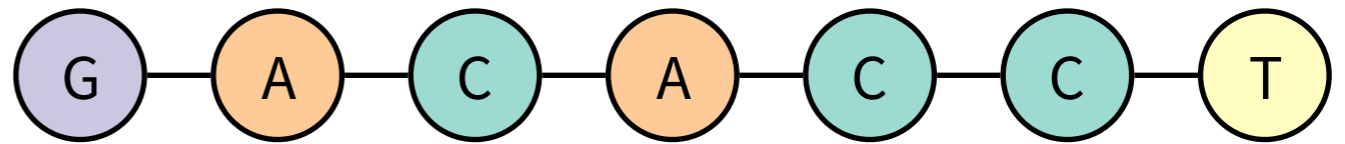


DNA molecules

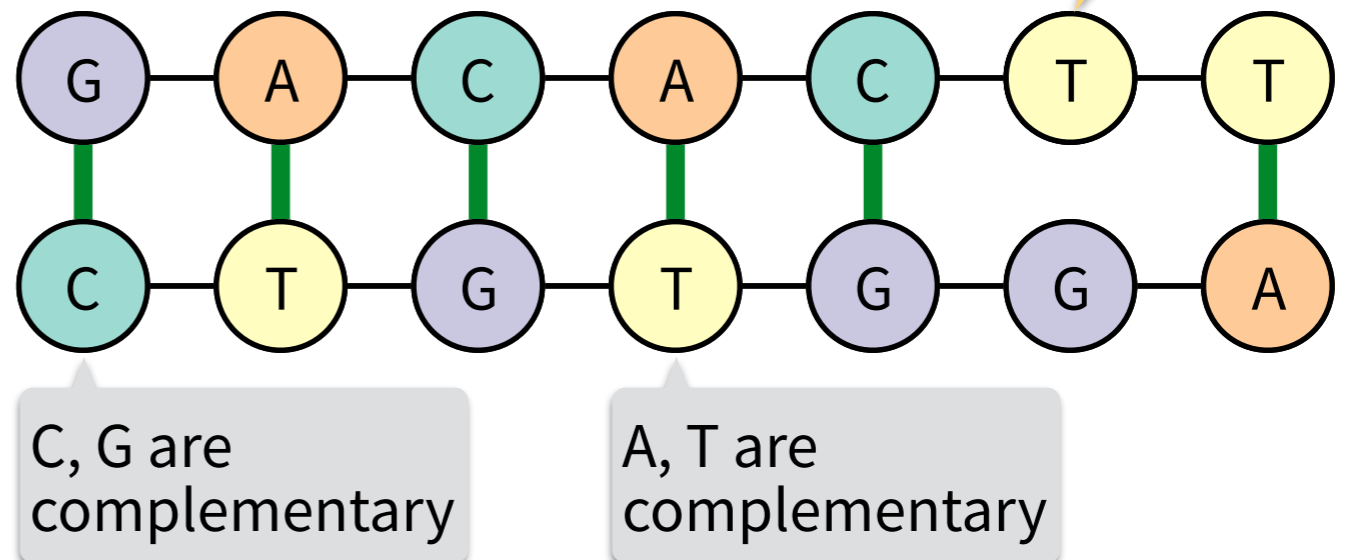
Four nucleotides:

-  Adenine
-  Cytosine
-  Guanine
-  Thymine

DNA strand (oligonucleotide) is a linear sequence of these nucleotides

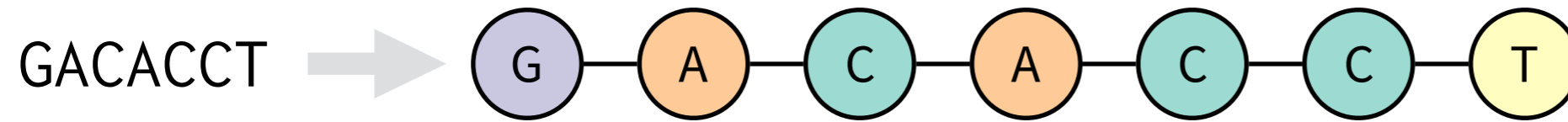


Two strands can bind to each other if they are complementary:



DNA manipulation

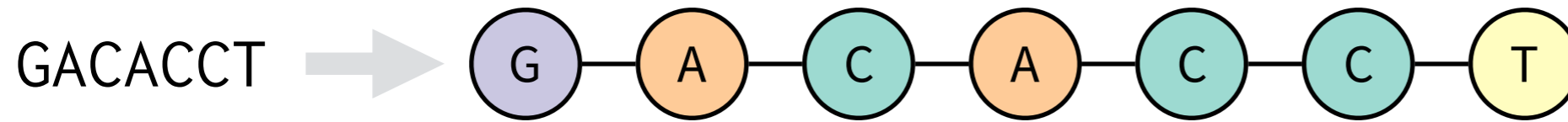
Synthesis: manufacturing DNA strands



- Chemical synthesis process appends one nucleotide at a time
- Maximum practical length ~200 nts
- Typically produces thousands of copies of the strand

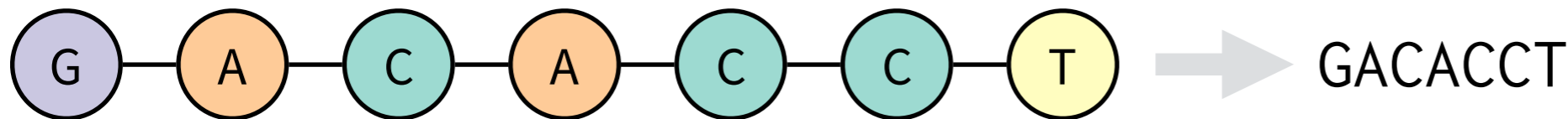
DNA manipulation

Synthesis: manufacturing DNA strands



- Chemical synthesis process appends one nucleotide at a time
- Maximum practical length ~200 nts
- Typically produces thousands of copies of the strand

Sequencing: reading DNA strands



- Produces *many* reads of a strand
- Much higher throughput than synthesis

An archival storage system

System overview

Archival storage system structured as a key-value store

System overview

Archival storage system structured as a key-value store

```
put(key, value)
```

System overview

Archival storage system structured as a key-value store

```
put(key, value)
```

```
get(key)
```

System overview

Archival storage system structured as a key-value store

```
put(key, value)
```

```
get(key)
```


System overview

Archival storage system structured as a key-value store

cat.jpg



↓ 01001...

`put(key, value)`

`get(key)`

System overview

Archival storage system structured as a key-value store

cat.jpg



↓ 01001...

`put(key, value)`

↓ ACGAT...

`get(key)`

System overview

Archival storage system structured as a key-value store

cat.jpg



↓ 01001...

put(key, value)

↓ ACGAT...

Synthesis

get(key)

System overview

Archival storage system structured as a key-value store

cat.jpg



↓ 01001...

put(key, value)

↓ ACGAT...

Synthesis



get(key)

System overview

Archival storage system structured as a key-value store

cat.jpg



↓ 01001...

put(key, value)

↓ ACGAT...

Synthesis

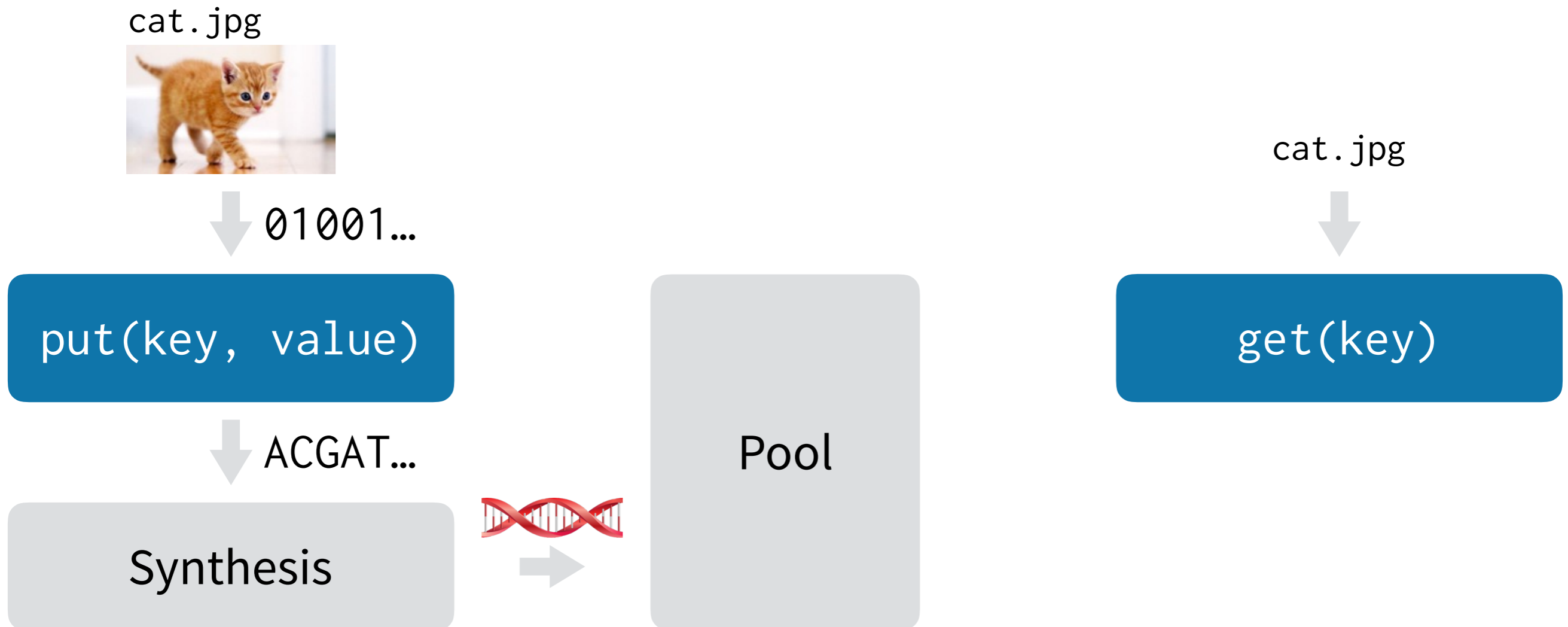


Pool

get(key)

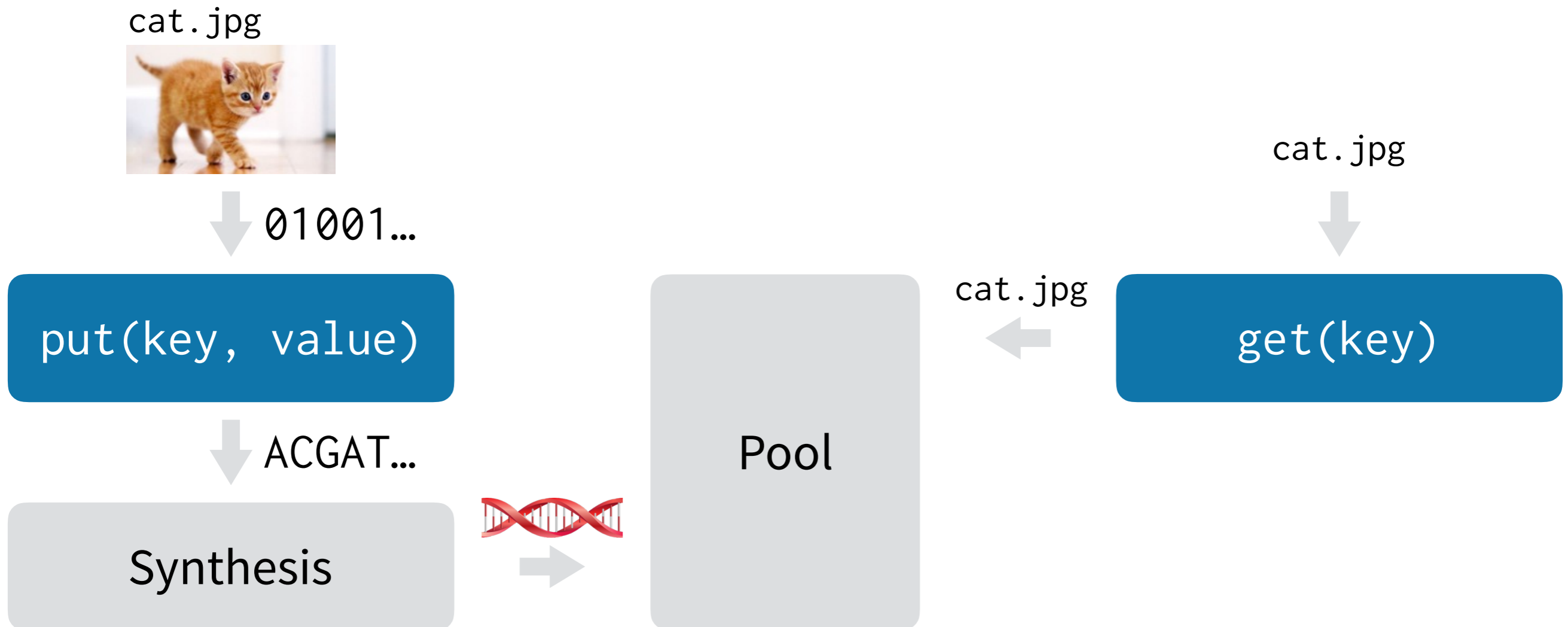
System overview

Archival storage system structured as a key-value store



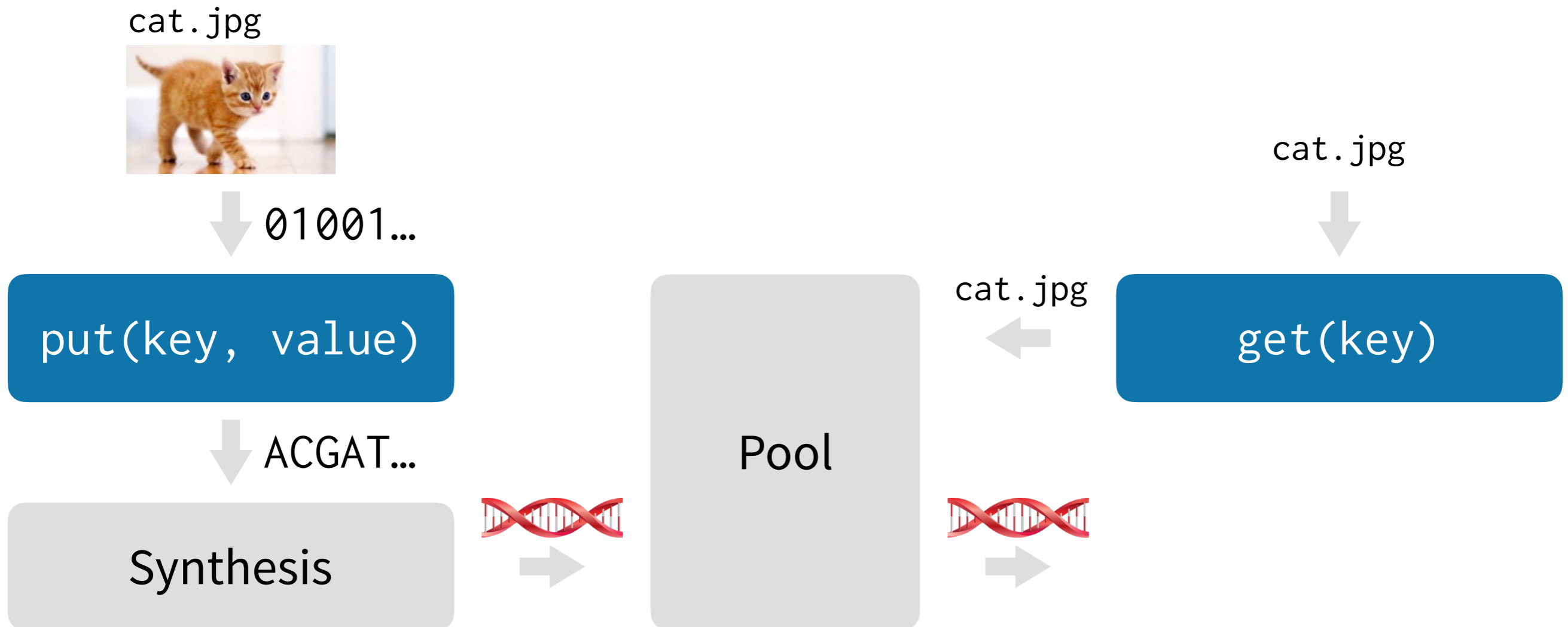
System overview

Archival storage system structured as a key-value store



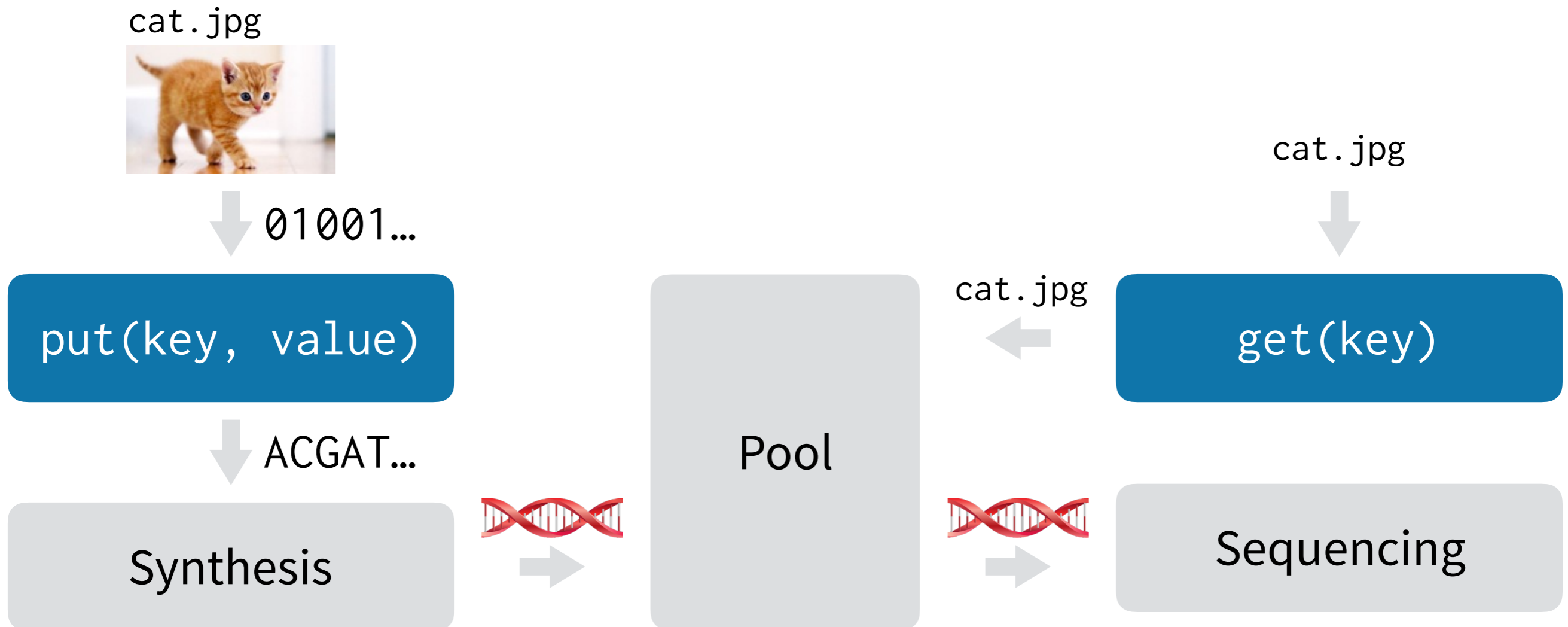
System overview

Archival storage system structured as a key-value store



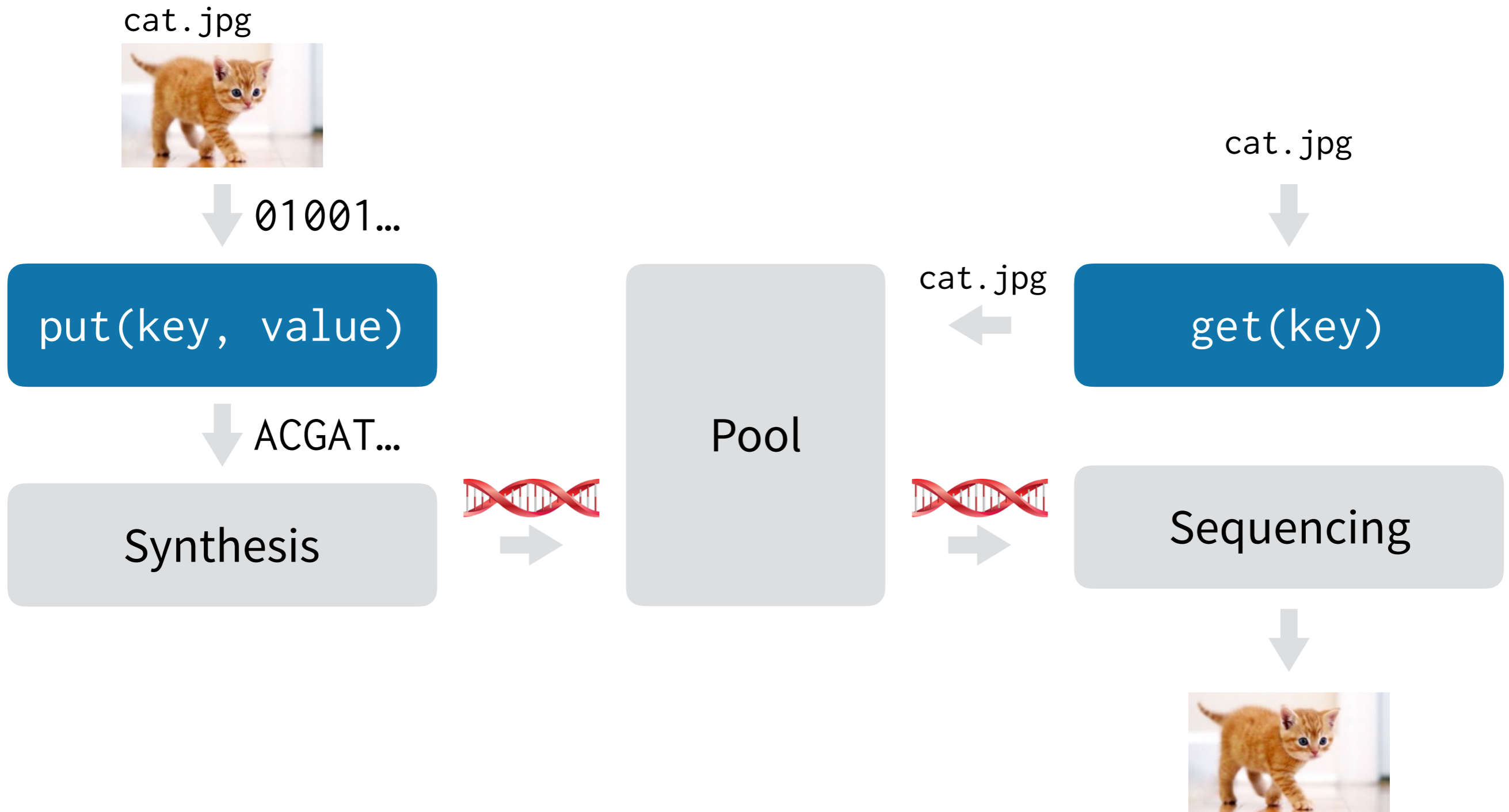
System overview

Archival storage system structured as a key-value store



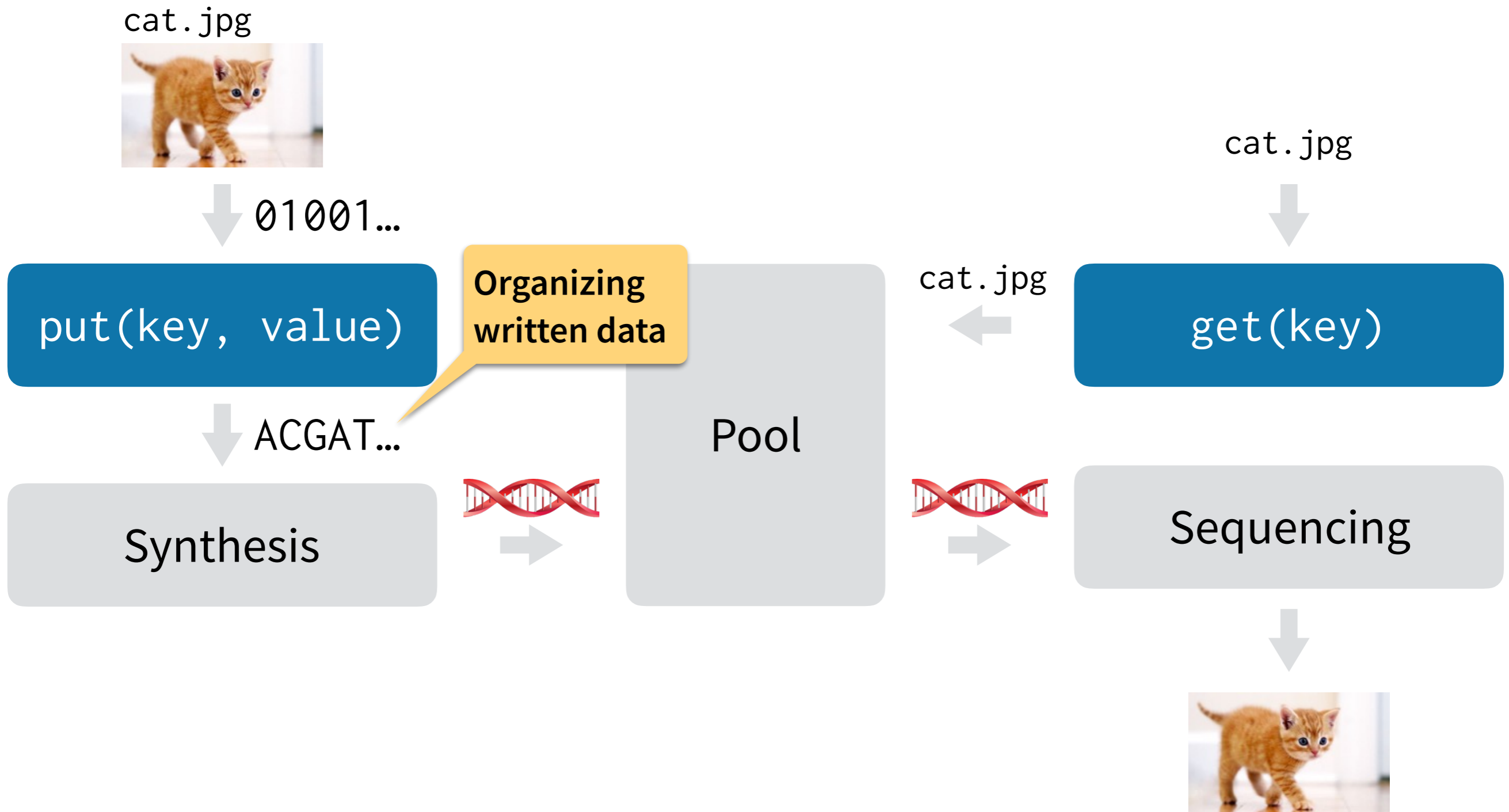
System overview

Archival storage system structured as a key-value store



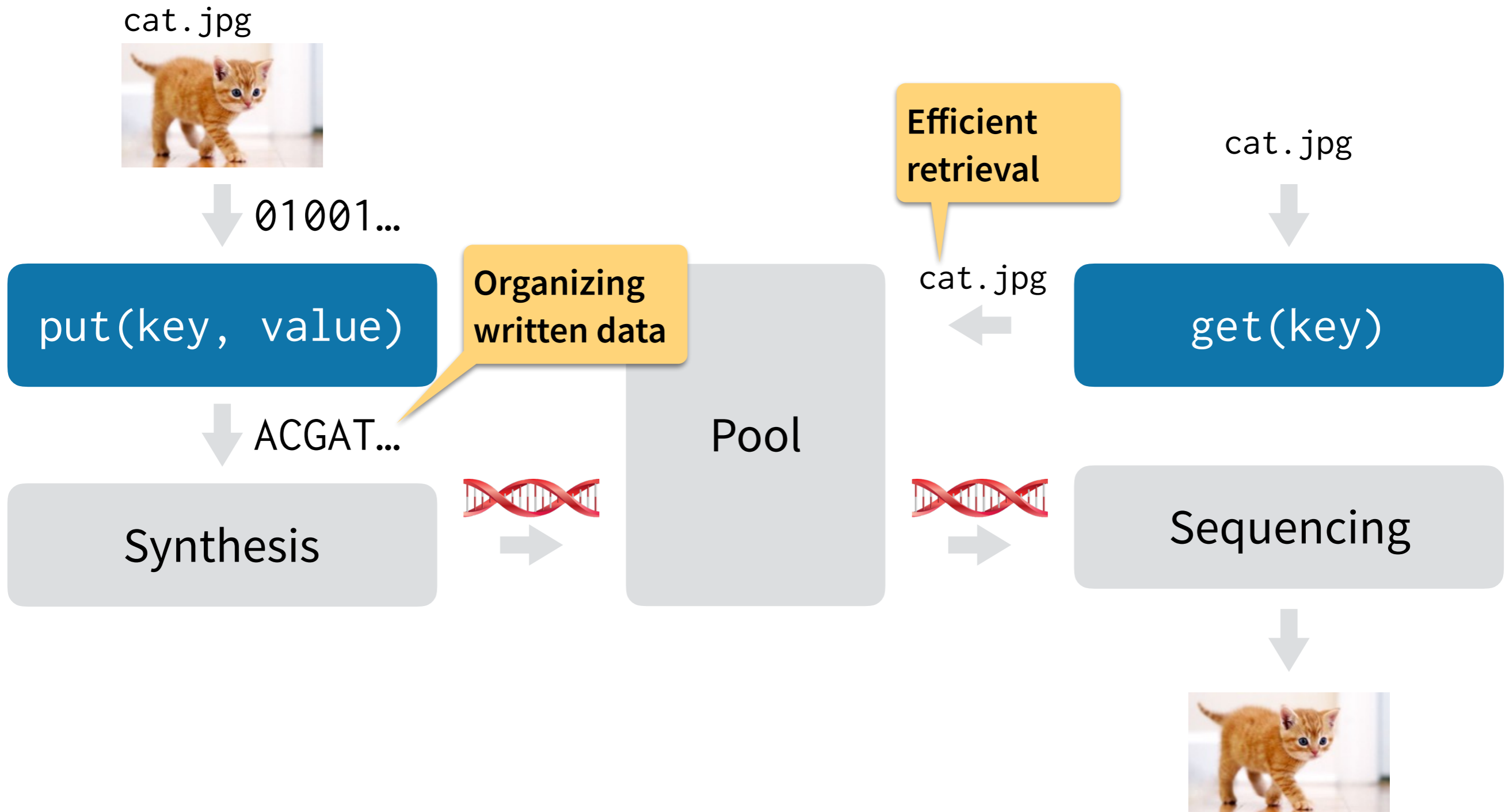
System overview

Archival storage system structured as a key-value store



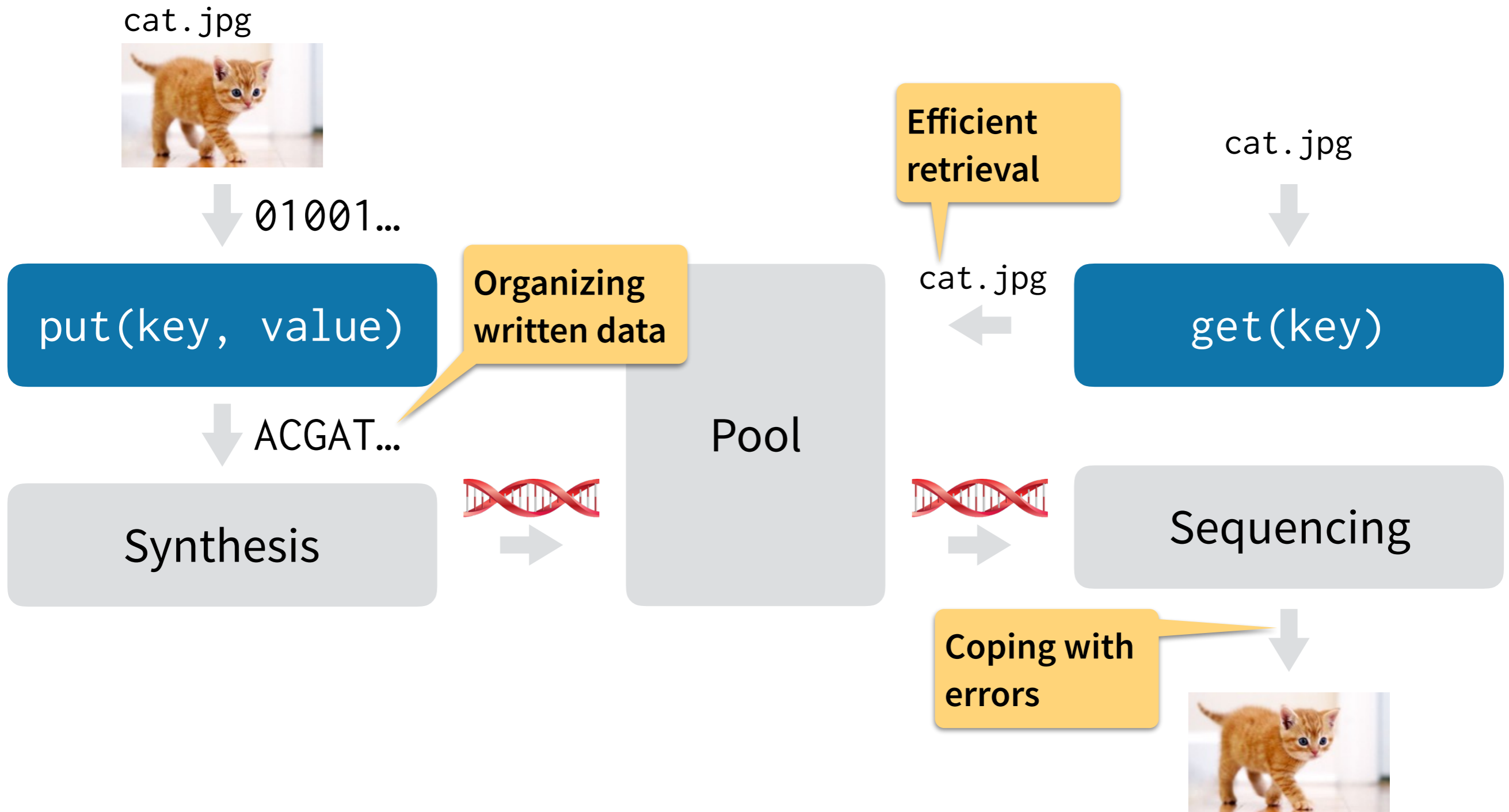
System overview

Archival storage system structured as a key-value store



System overview

Archival storage system structured as a key-value store



Writing data to DNA

The easy way: convert base 2 to base 4

10100011 10010001 11100111 11000101 10010100 10111101

Writing data to DNA

The easy way: convert base 2 to base 4

10100011 10010001 11100111 11000101 10010100 10111101



2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

Writing data to DNA

The easy way: convert base 2 to base 4

10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

Writing data to DNA

The easy way: convert base 2 to base 4

10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes

Writing data to DNA

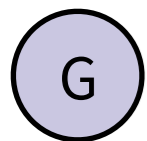
The easy way: convert base 2 to base 4

10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes



Writing data to DNA

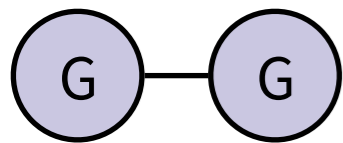
The easy way: convert base 2 to base 4

10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes



Writing data to DNA

The easy way: convert base 2 to base 4

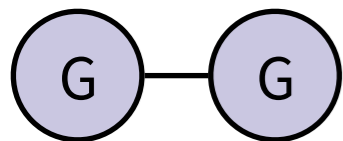
10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes

$P[\text{Attach}] = 99\%$



Writing data to DNA

The easy way: convert base 2 to base 4

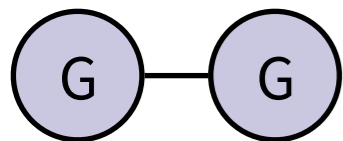
10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes

$P[\text{Attach}] = 99\%$



99%

Writing data to DNA

The easy way: convert base 2 to base 4

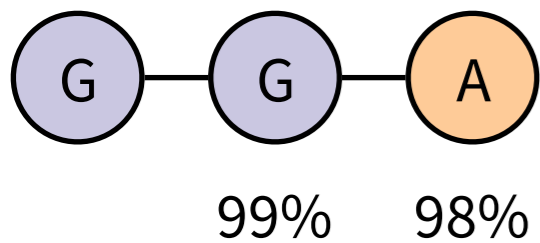
10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes

$P[\text{Attach}] = 99\%$



Writing data to DNA

The easy way: convert base 2 to base 4

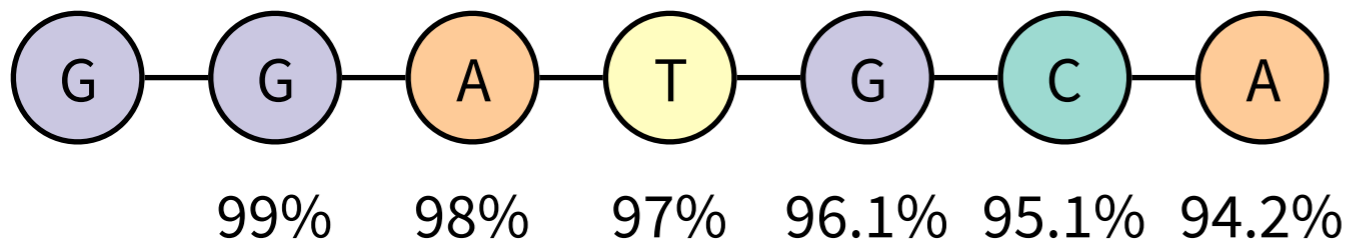
10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes

$P[\text{Attach}] = 99\%$



Writing data to DNA

The easy way: convert base 2 to base 4

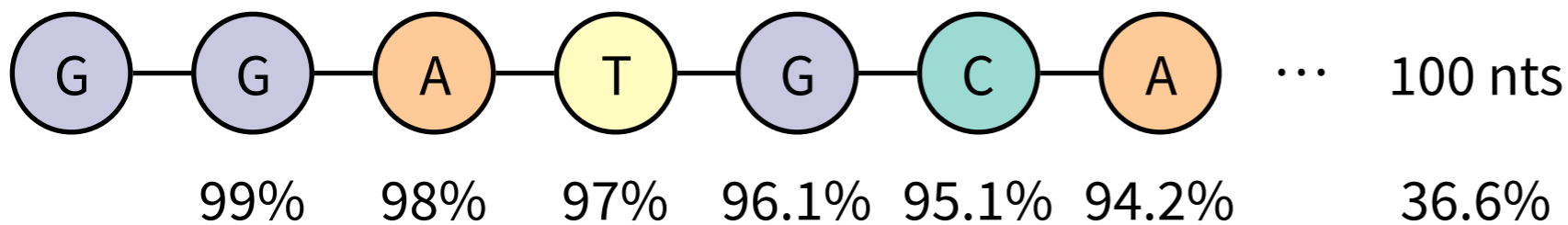
10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes

$P[\text{Attach}] = 99\%$



Writing data to DNA

The easy way: convert base 2 to base 4

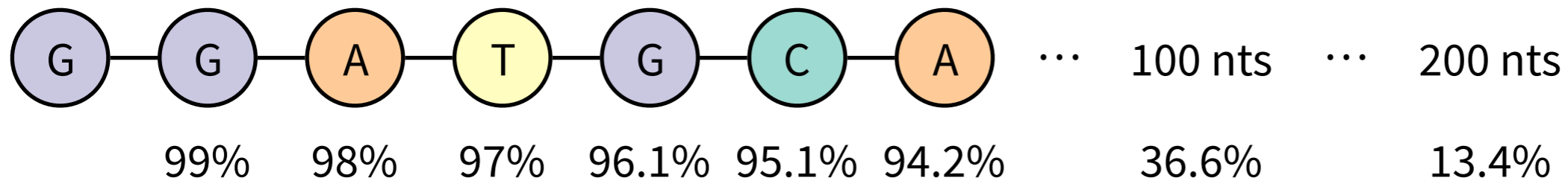
10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

But this approach isn't feasible for more than a few bytes

$P[\text{Attach}] = 99\%$



Chunking data

Break binary data into chunks stored in separate strands

10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C T G C T T A C C G C C A G T T C

Chunking data

Break binary data into chunks stored in separate strands

10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C

T G C T T A C C

G C C A G T T C

Chunking data

Break binary data into chunks stored in separate strands

10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1

G G A T G C A C A A A A
T G C T T A C C A A A C
G C C A G T T C A A A G

Addresses
within the value

Chunking data

Break binary data into chunks stored in separate strands

10100011 10010001 11100111 11000101 10010100 10111101

2 2 0 3 2 1 0 1 3 2 1 3 3 0 1 1 2 1 1 0 2 3 3 1



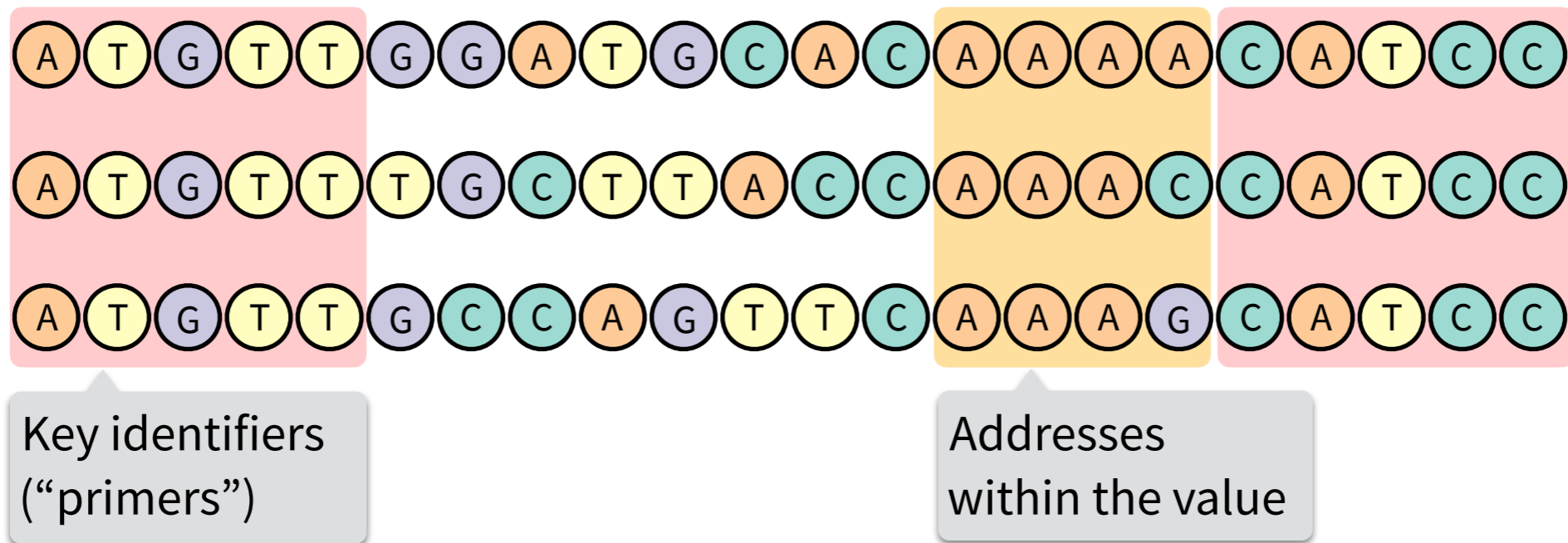
Key identifiers
("primers")

Addresses
within the value

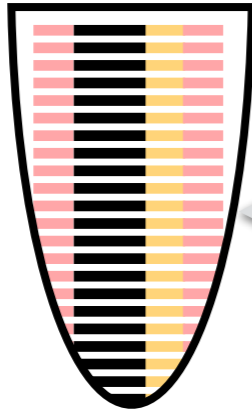
Efficient reads



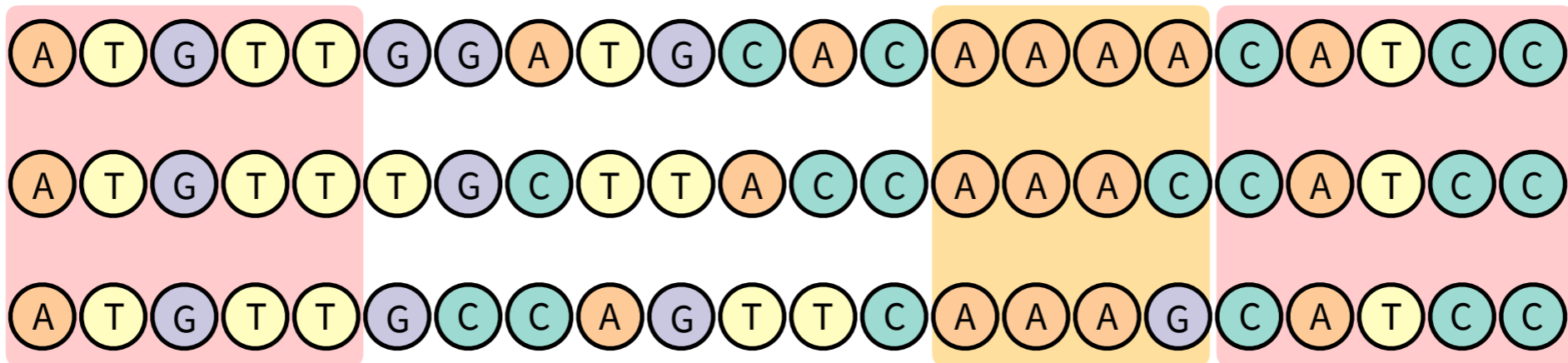
Efficient reads



Efficient reads



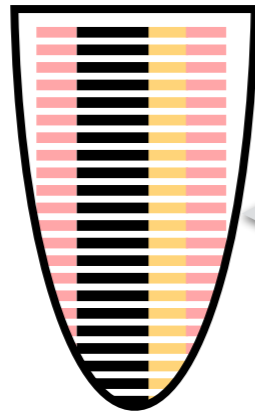
Pool containing stored strands for all keys & values!



Key identifiers
("primers")

Addresses
within the value

Efficient reads

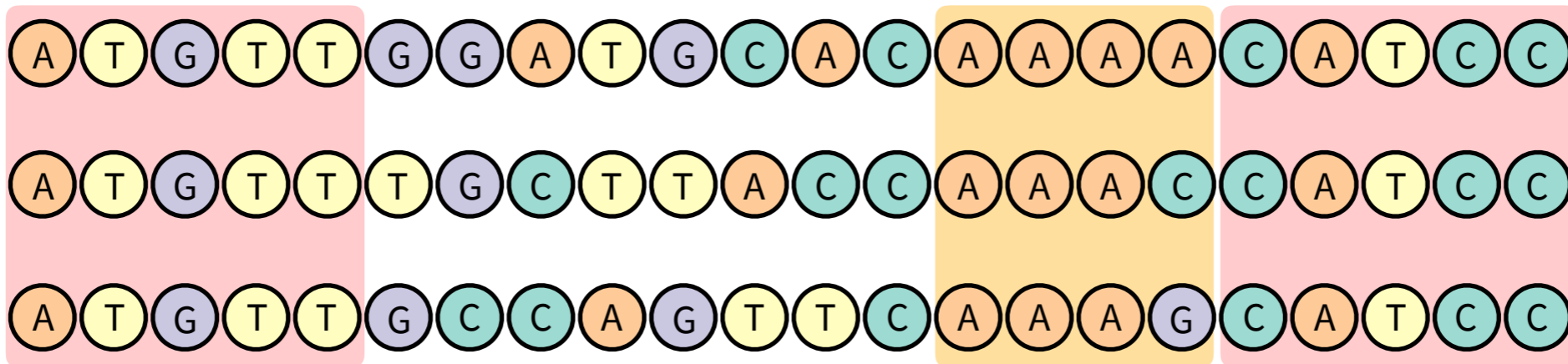


Pool containing stored strands for all keys & values!

cat.jpg



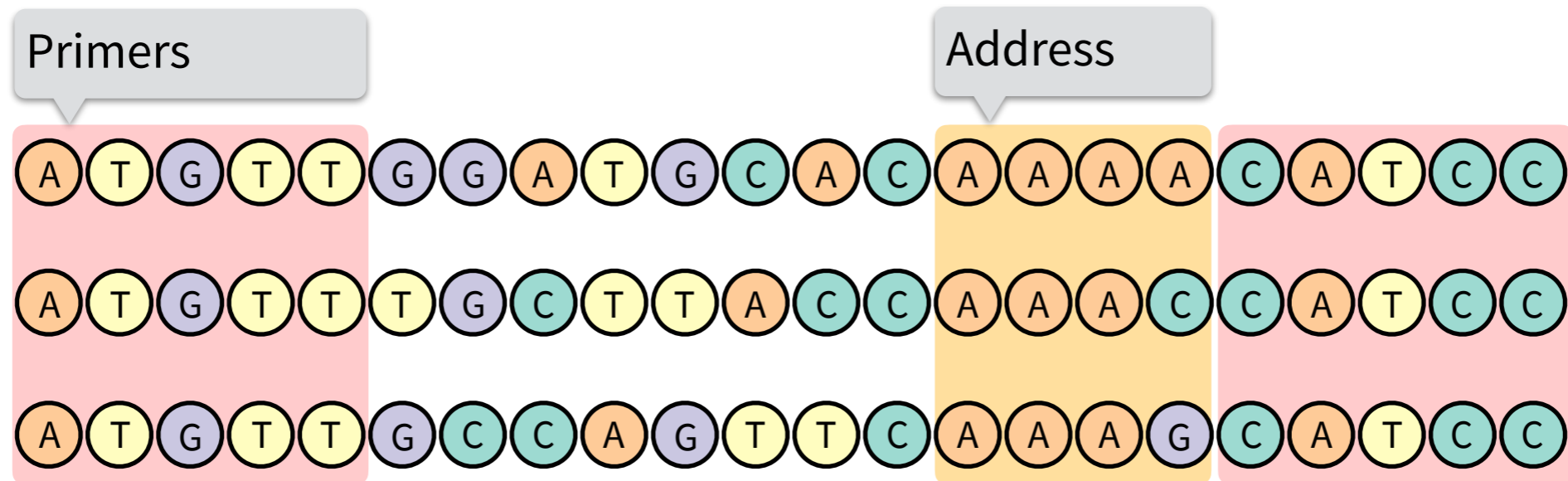
get(key)



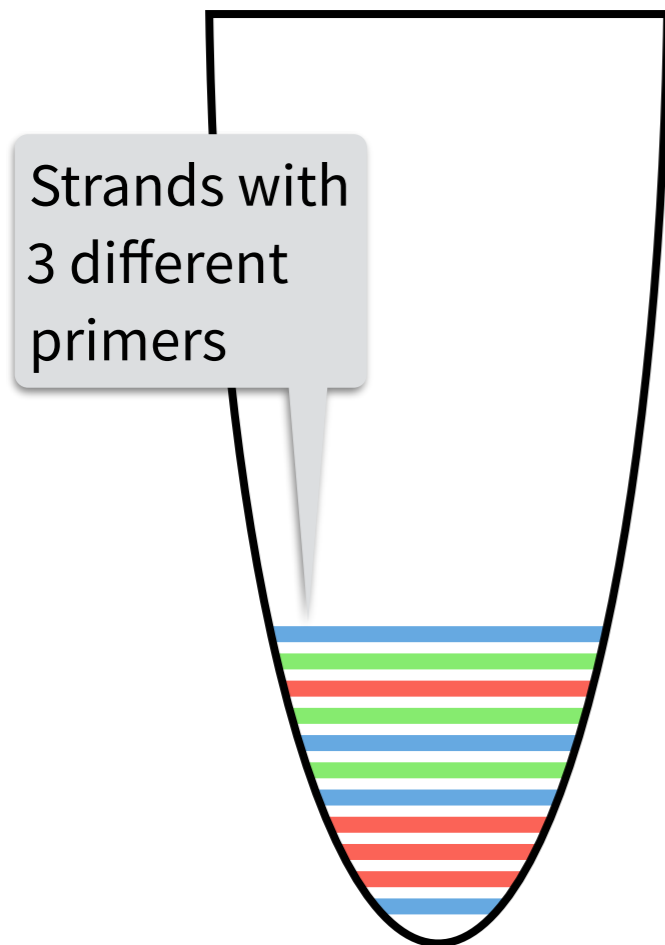
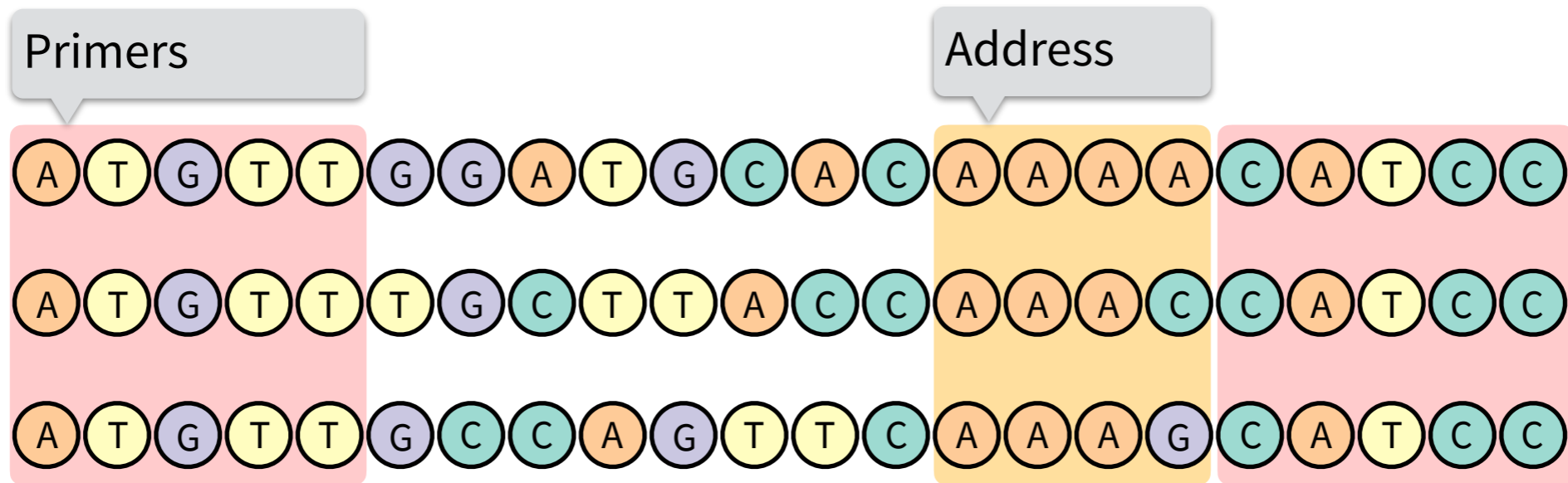
Key identifiers ("primers")

Addresses within the value

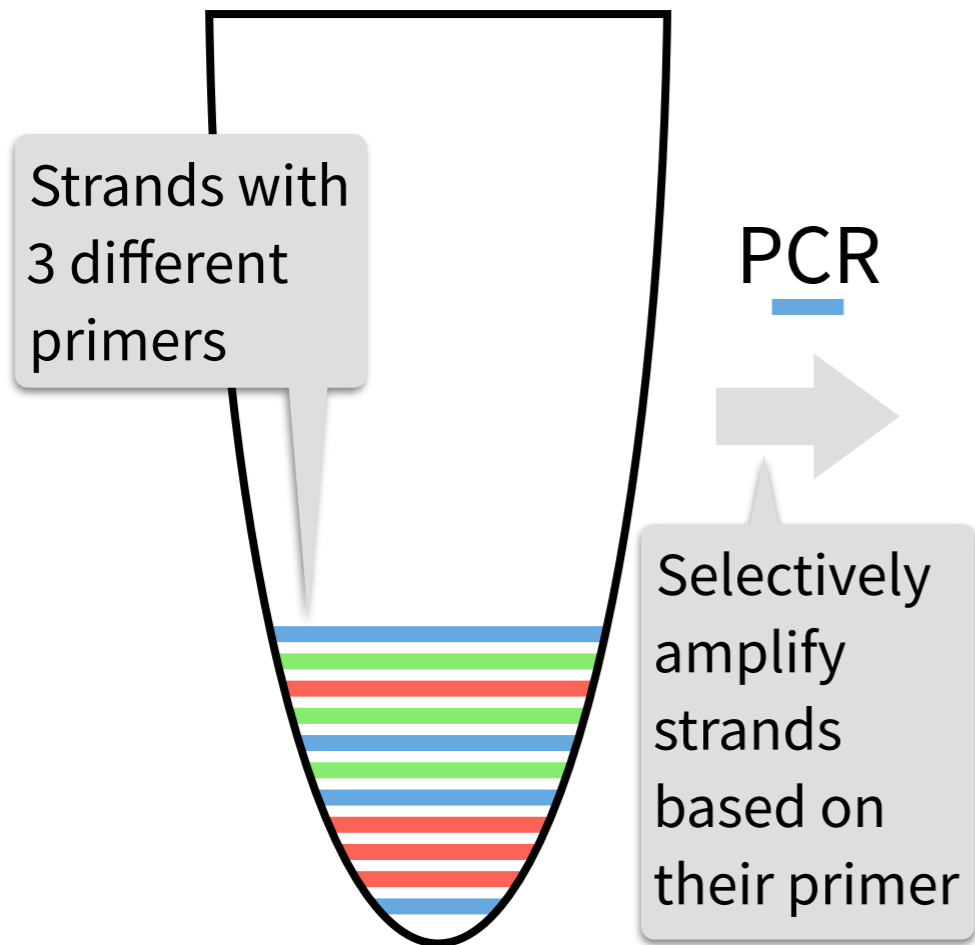
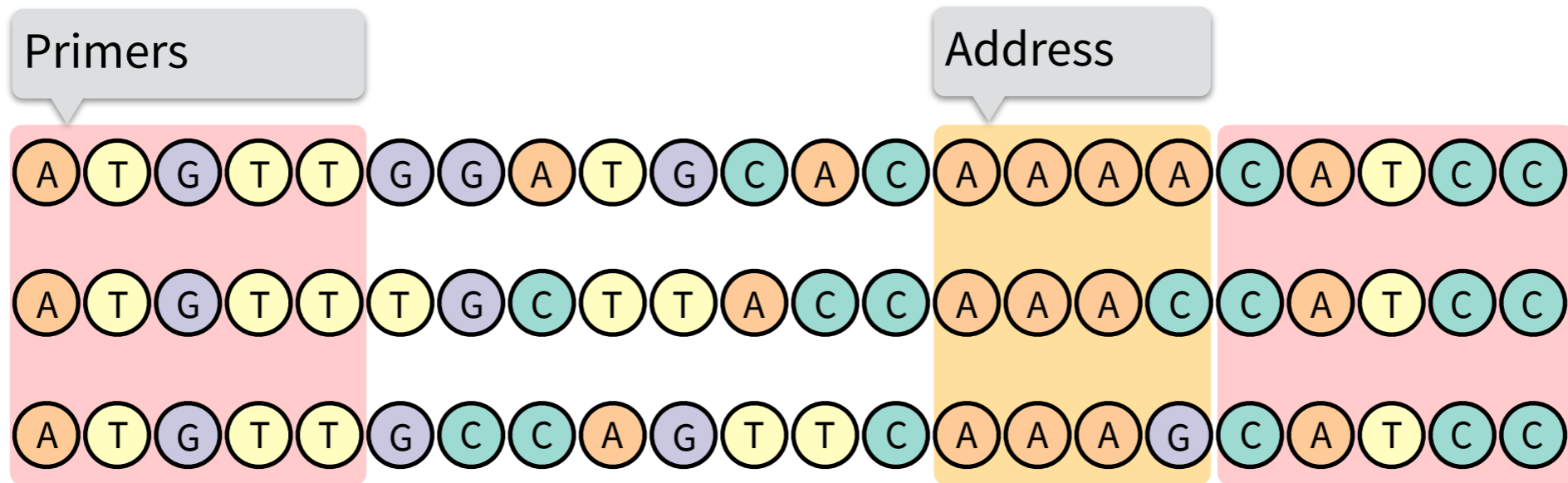
Random access



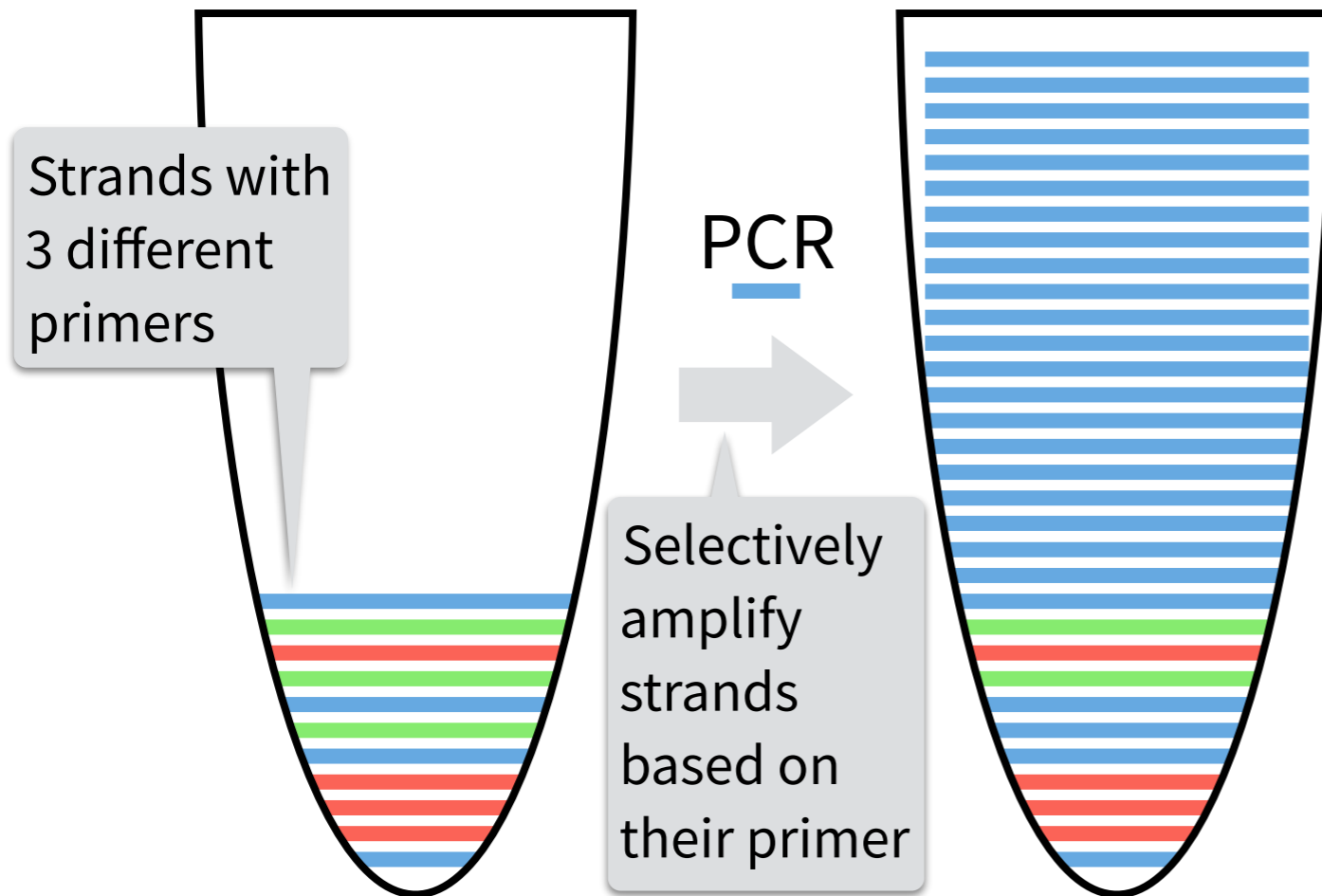
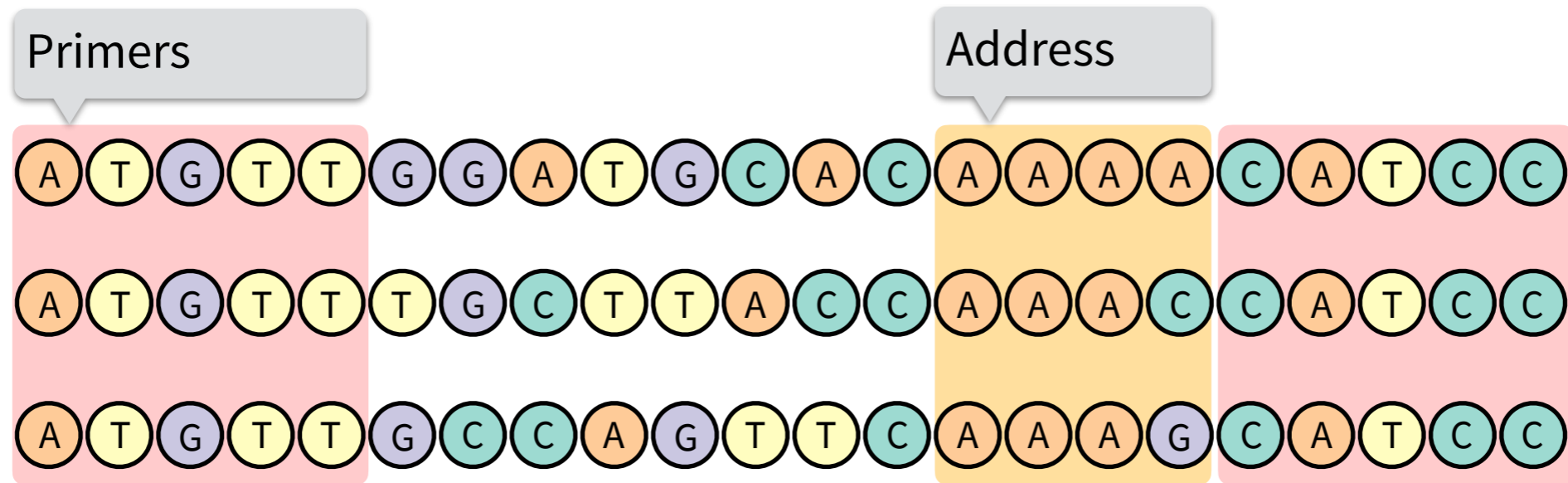
Random access



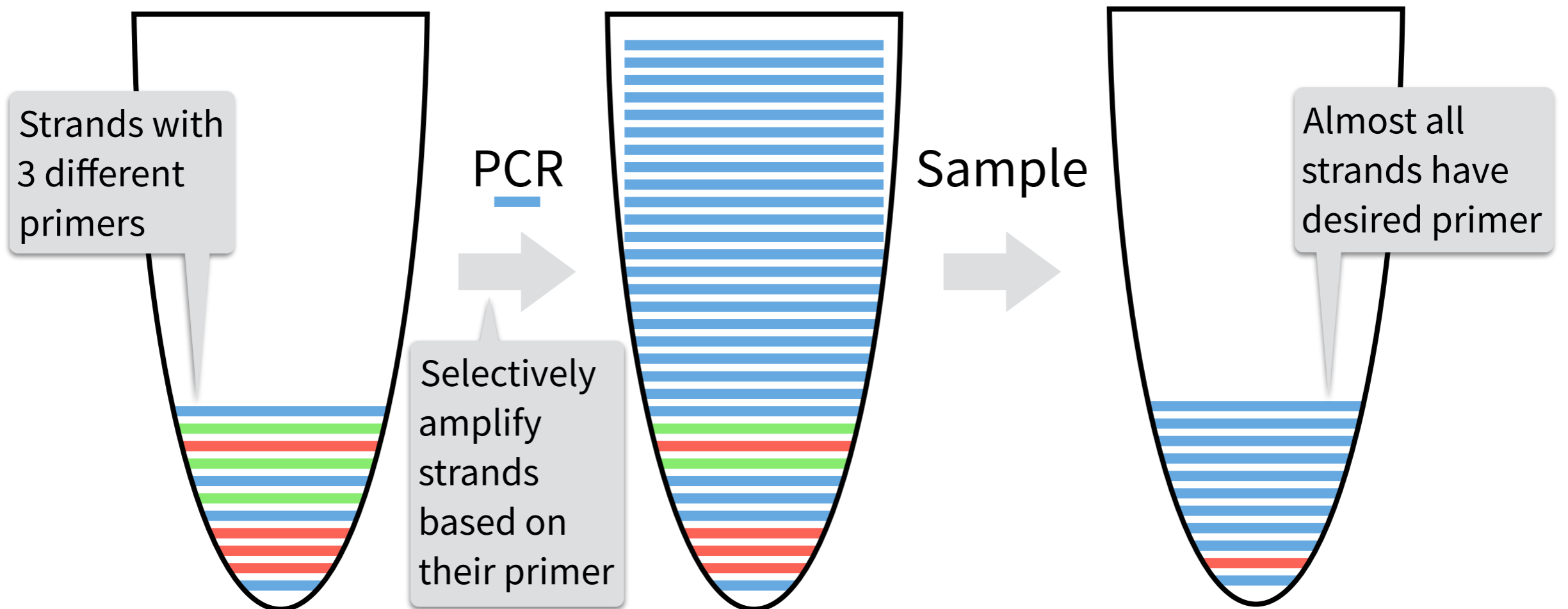
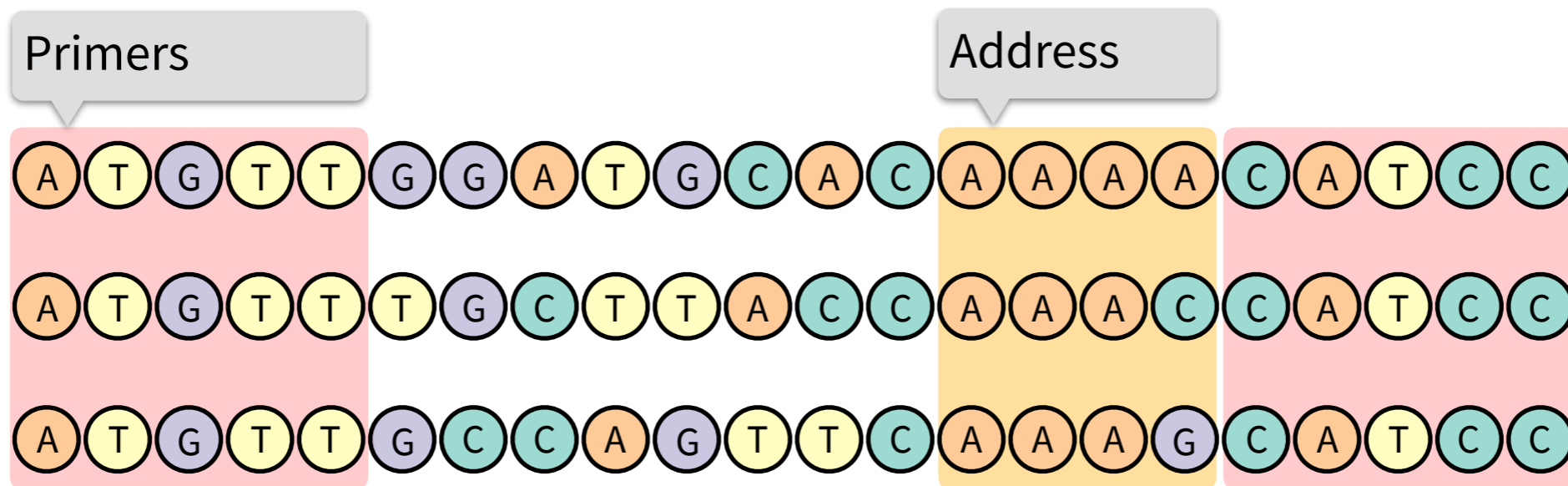
Random access



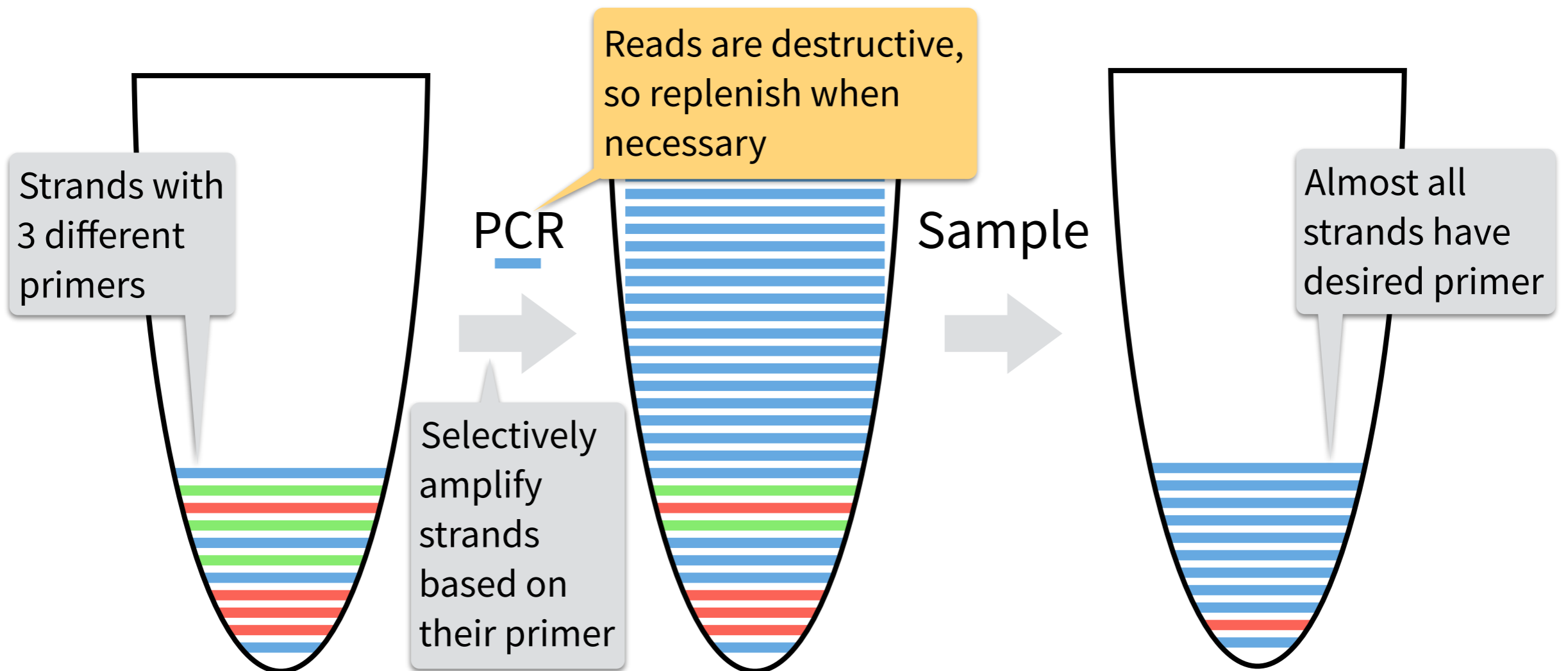
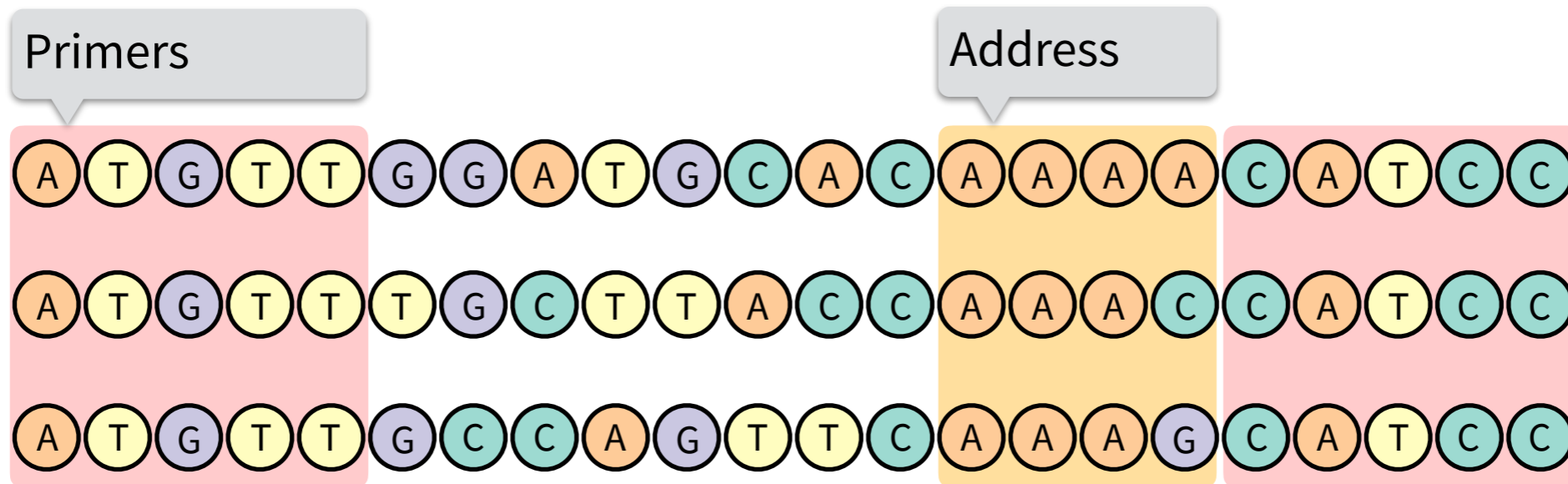
Random access



Random access

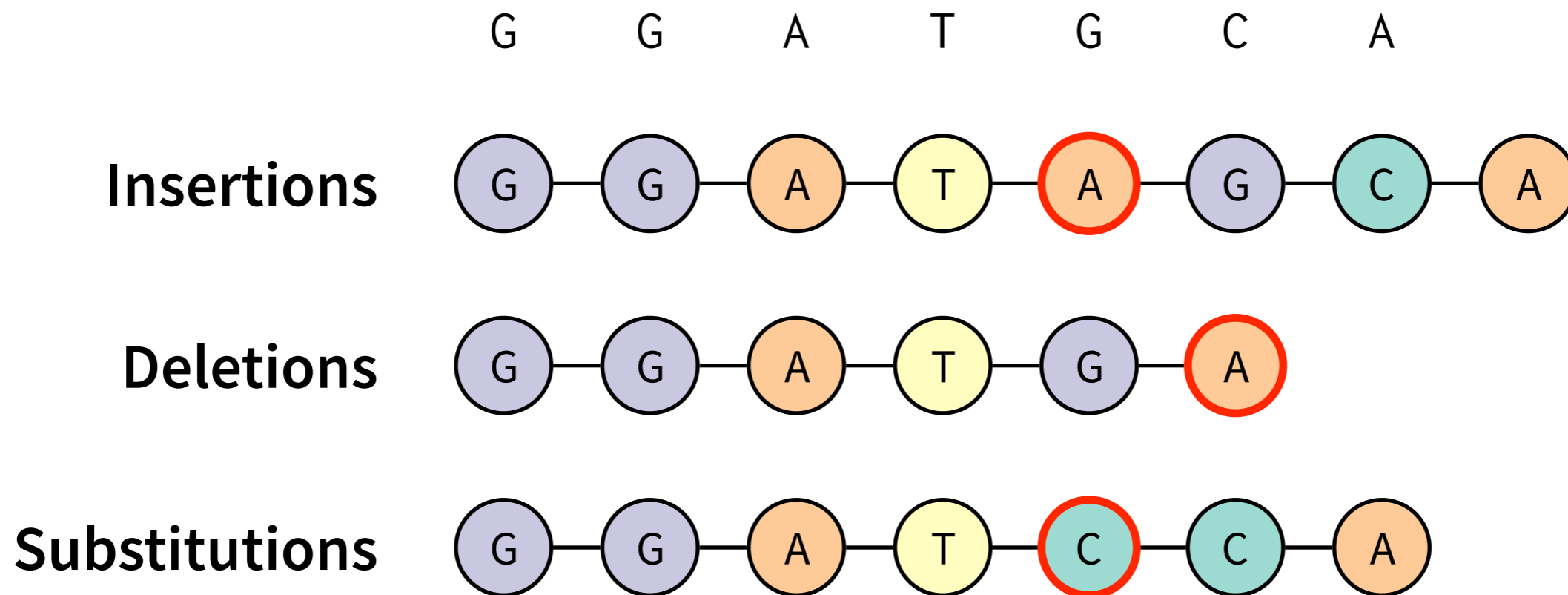


Random access



Error correction

Both synthesis and sequencing are error prone:



Error rates ~1%
per nucleotide!

Logical redundancy

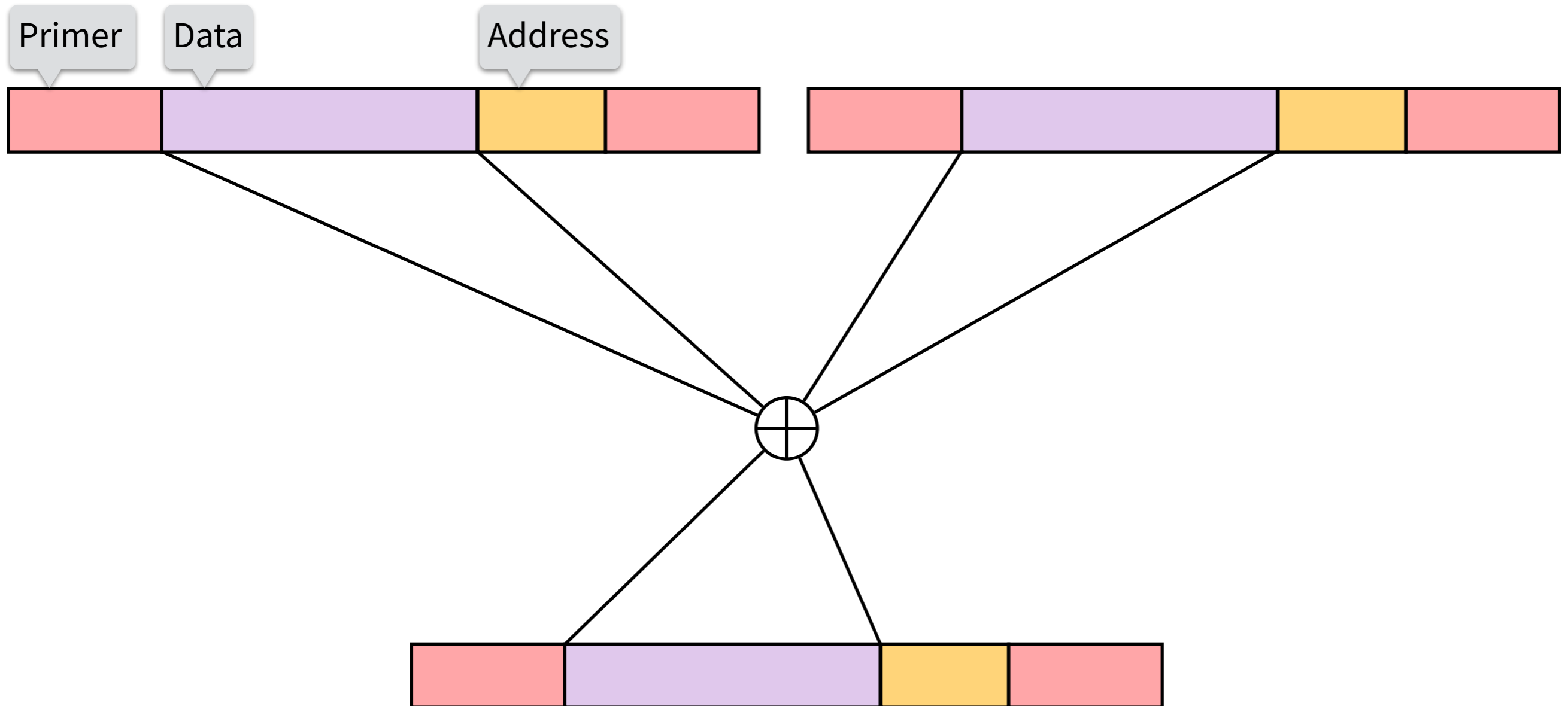
Logical redundancy



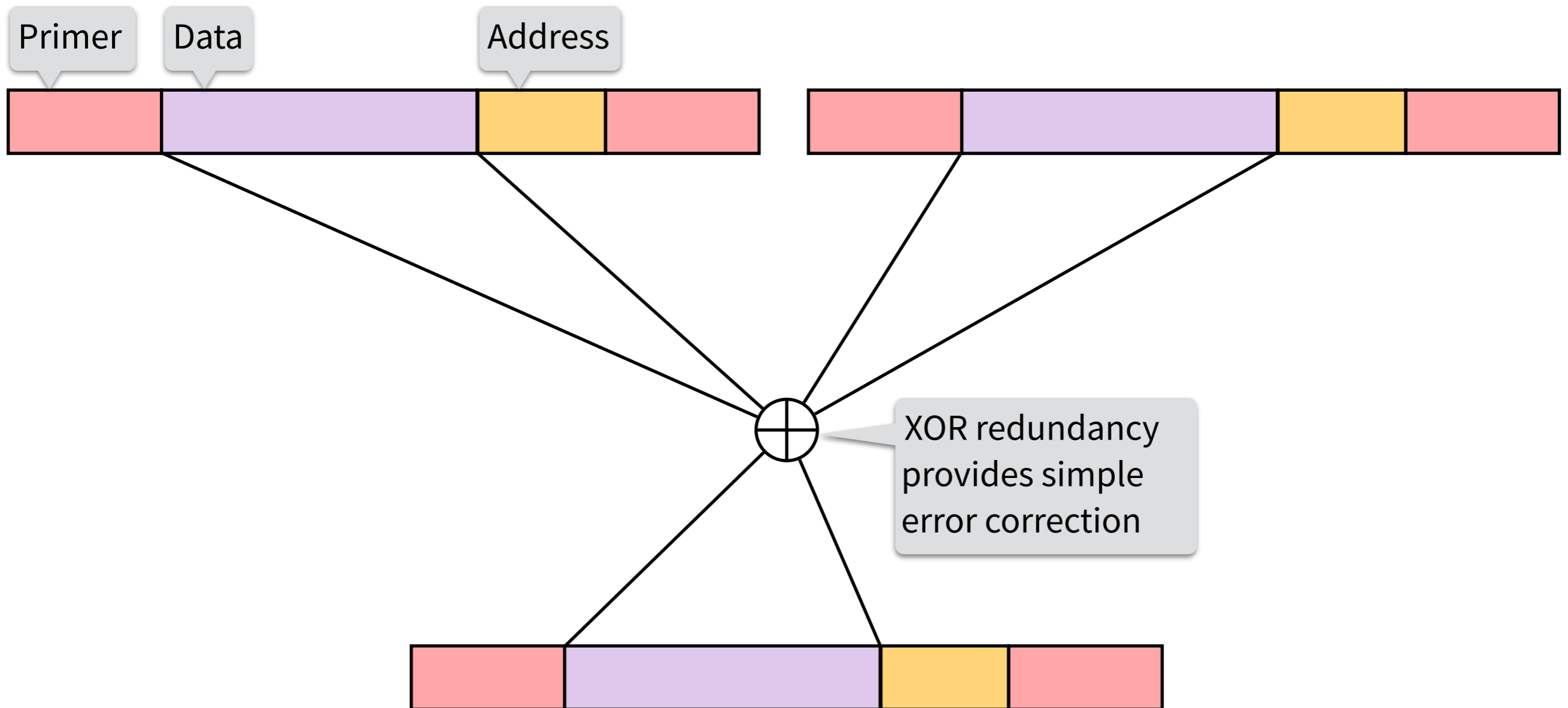
Logical redundancy



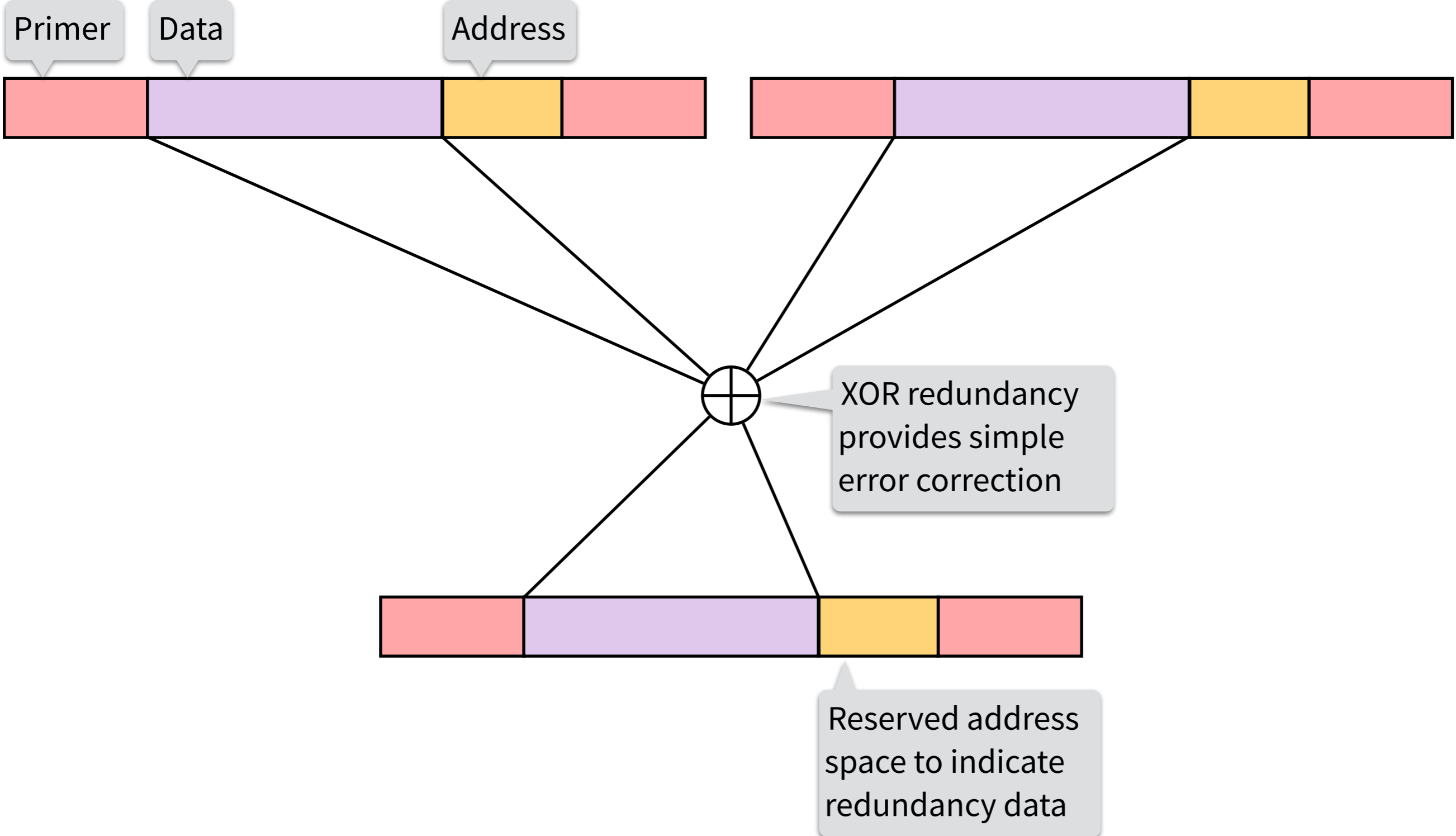
Logical redundancy



Logical redundancy



Logical redundancy



Wet lab results

The process

The process



The process



The process



catcatgg

The process



catcatgg



The process



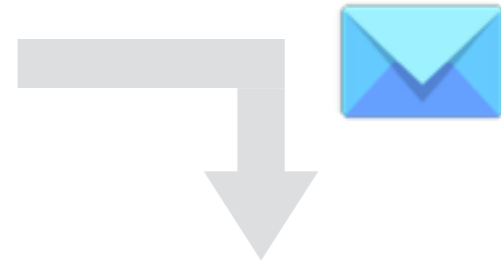
catcatgg



The process



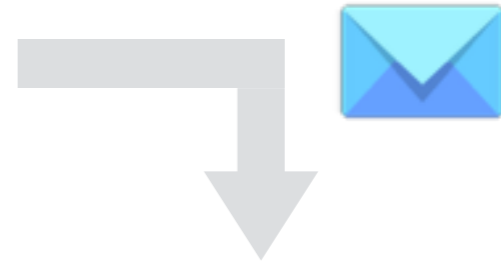
catcatgg



The process



catcatgg



FedEx

The process



catcatgg



catcatg**c**



FedEx

The process



catcatgg



catcatgc



FedEx

The process



catcatgg



catcatgc

Throughput
MBs/week



FedEx

Decoding

Encoded and synthesized 3 files (151 kB):



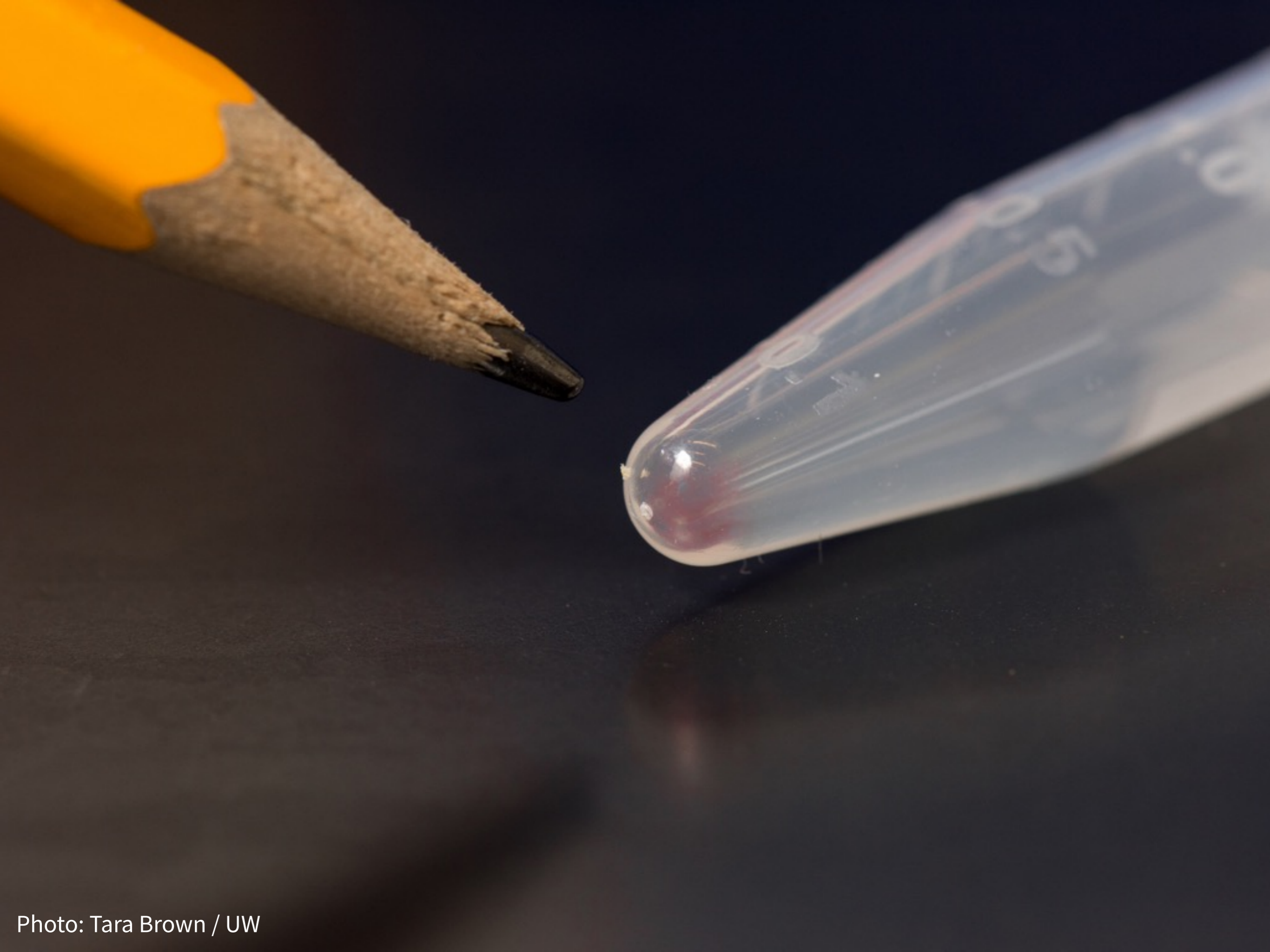


Photo: Tara Brown / UW

Decoding

Encoded and synthesized 3 files (151 kB):



Decoding

Encoded and synthesized 3 files (151 kB):



Selected and PCRred one file for random access (42 kB):



Decoding

Encoded and synthesized 3 files (151 kB):



Selected and PCRred one file for random access (42 kB):



Sequenced and decoded the resulting amplified pool:

Decoding

Encoded and synthesized 3 files (151 kB):



Selected and PCRred one file for random access (42 kB):

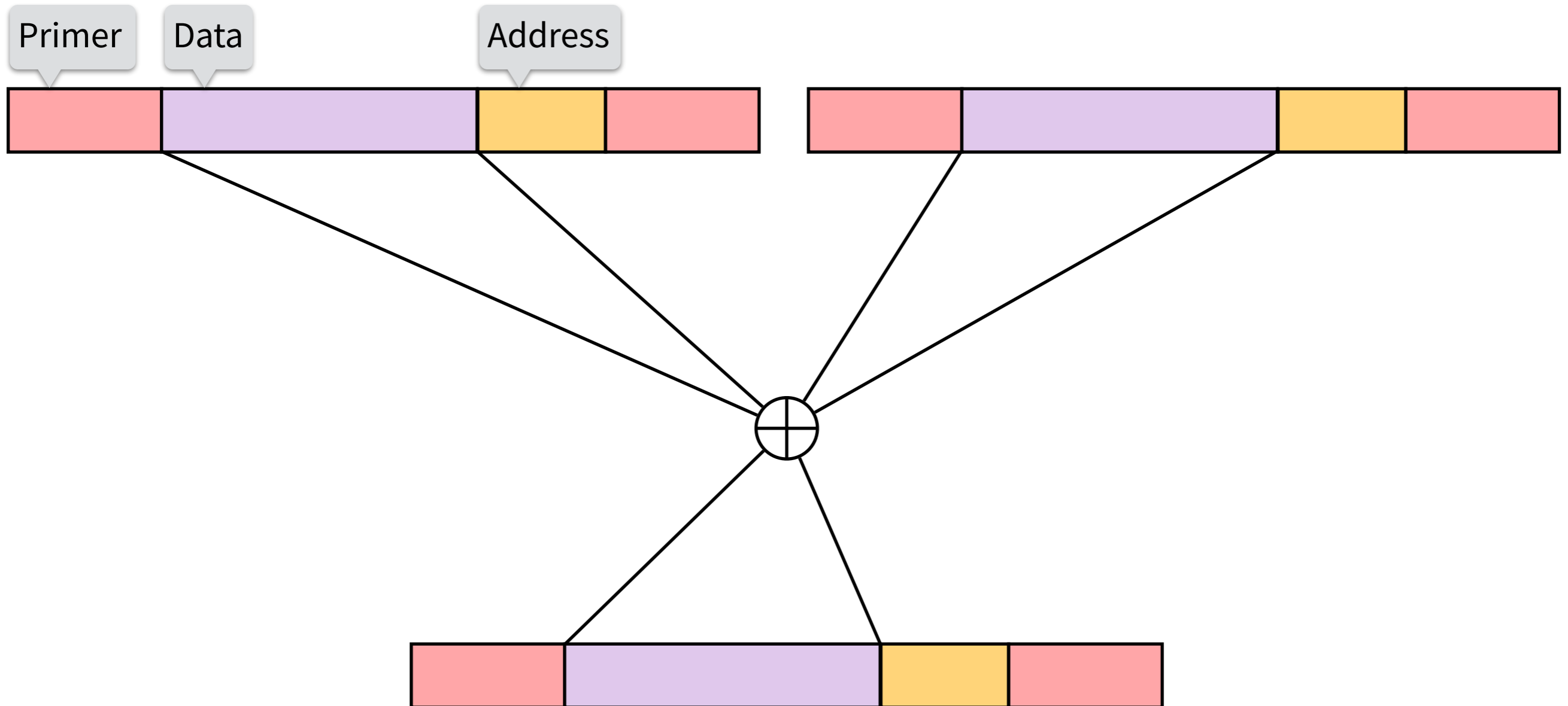


Sequenced and decoded the resulting amplified pool:

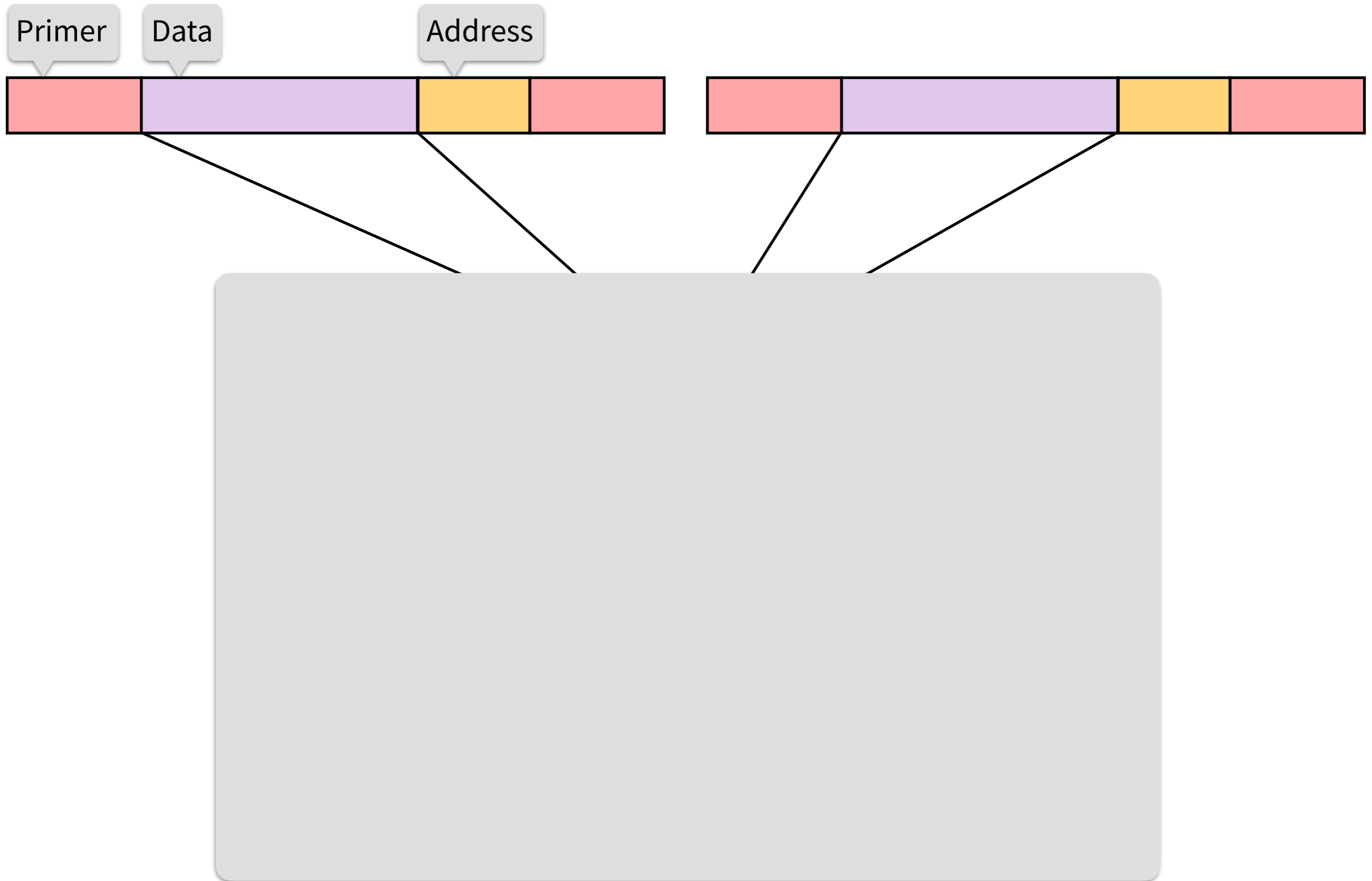


Recovered *every bit* despite errors in synthesis and sequencing

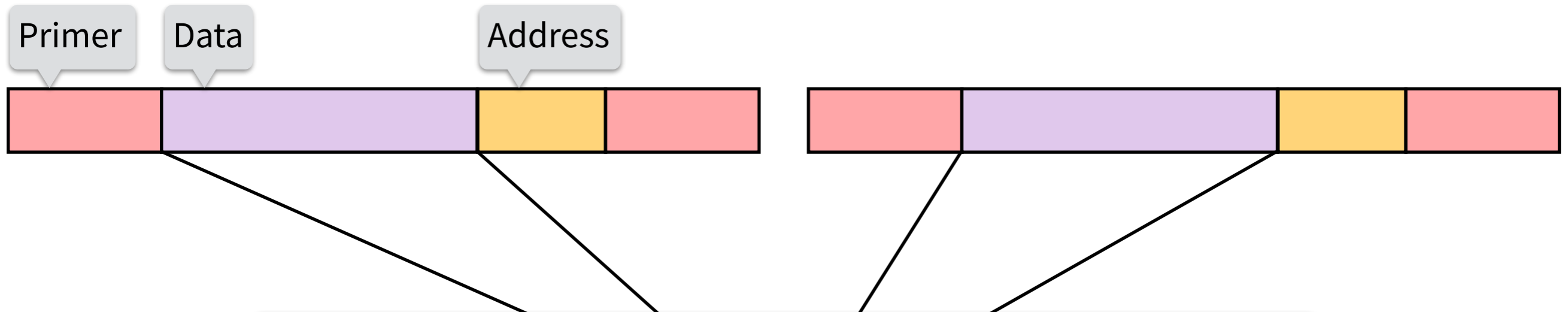
The importance of redundancy



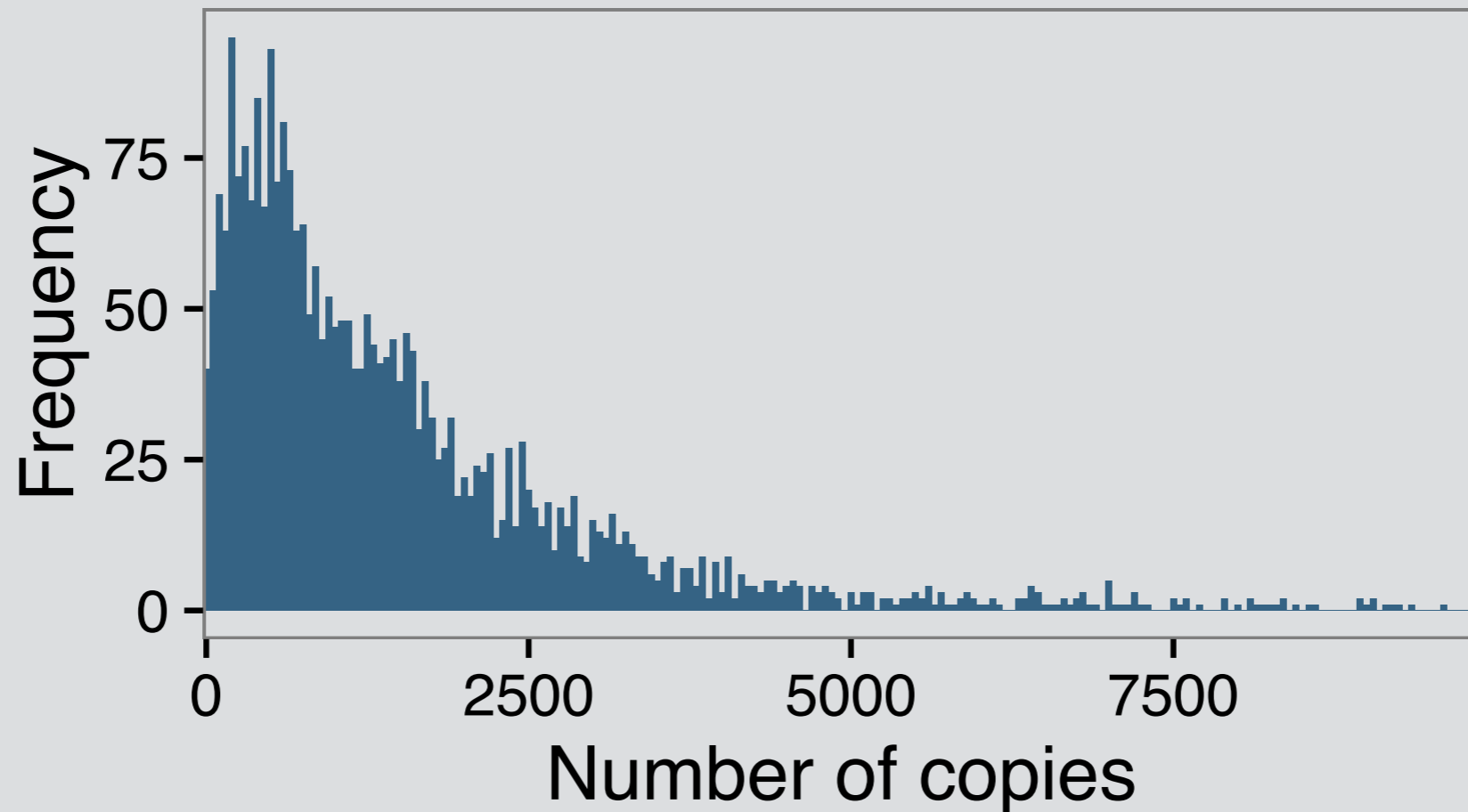
The importance of redundancy



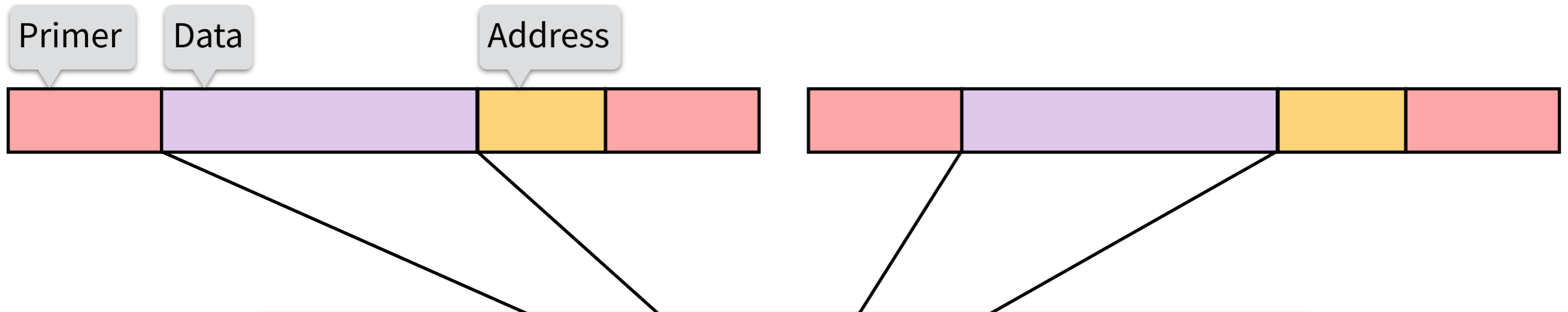
The importance of redundancy



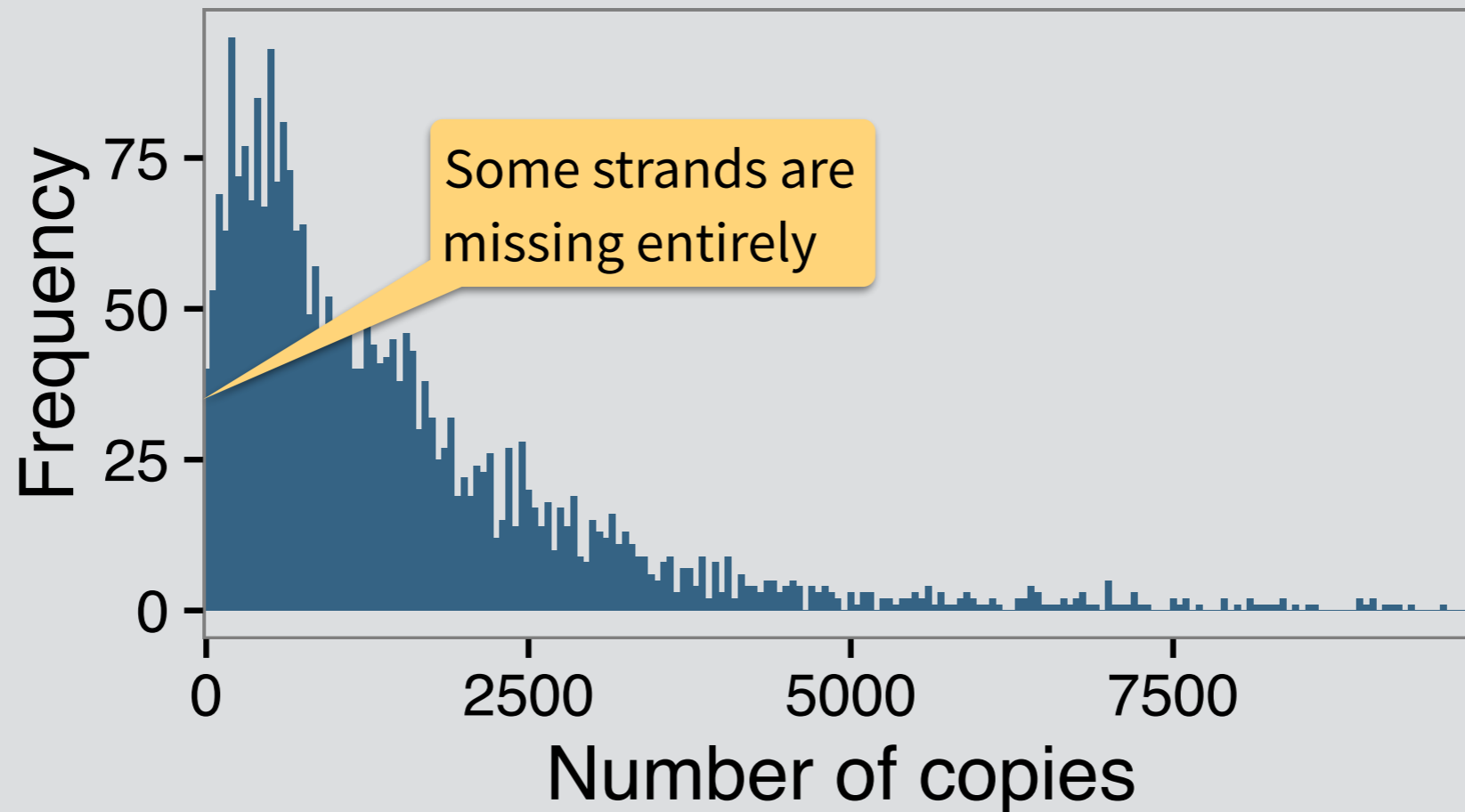
If we ignore redundancy data, we cannot recover the file.



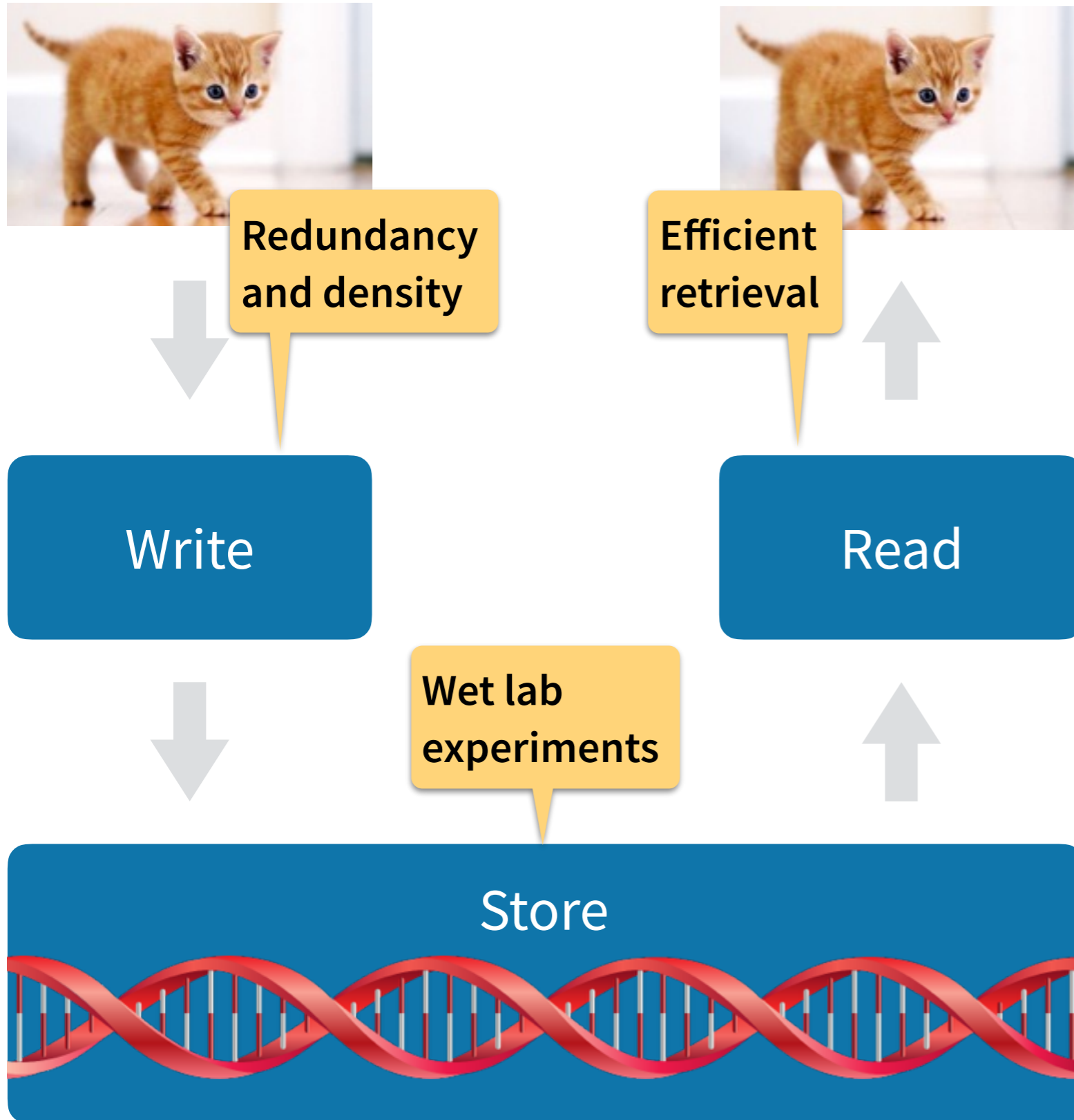
The importance of redundancy



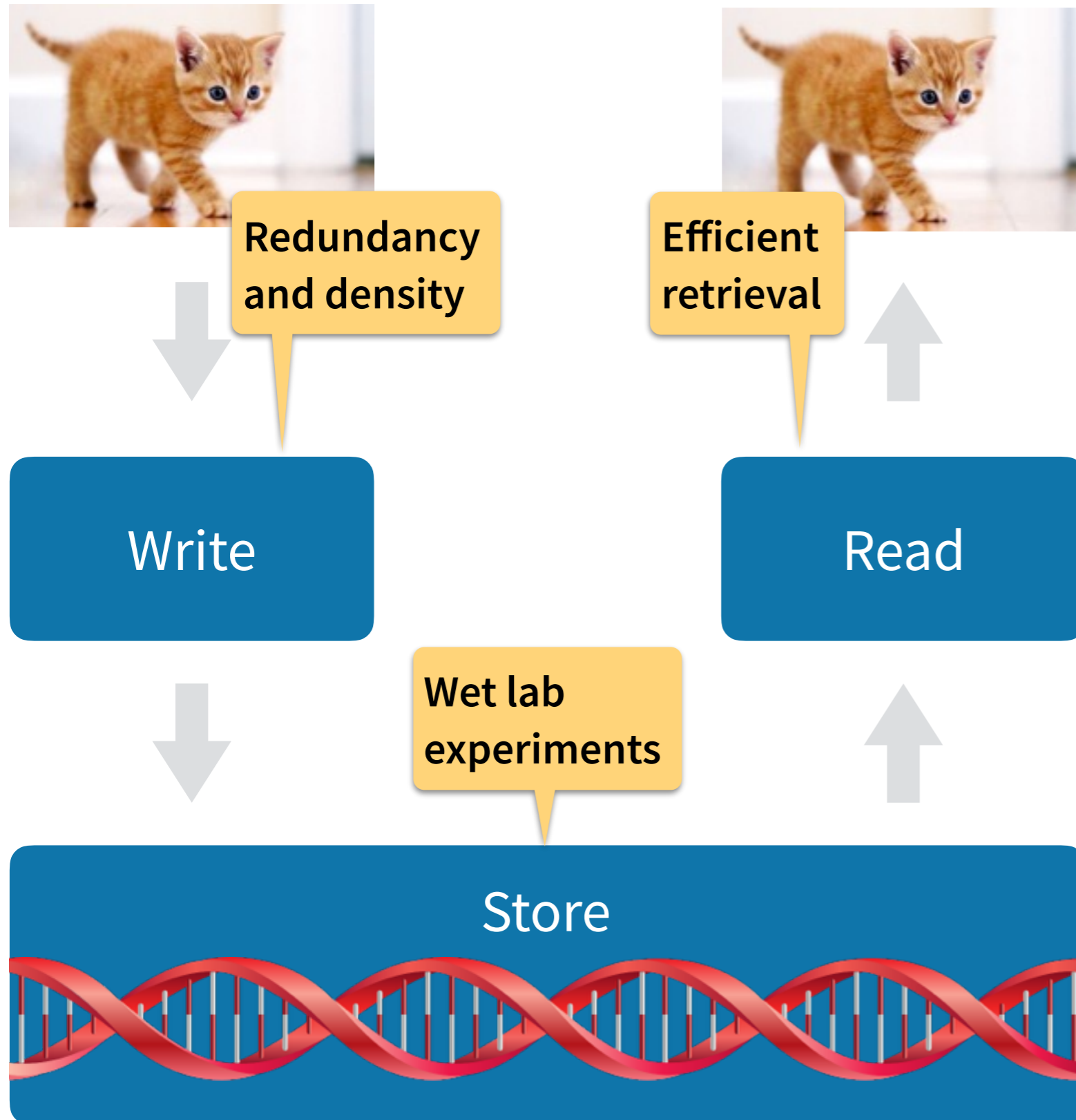
If we ignore redundancy data, we cannot recover the file.



A DNA-based archival storage system



A DNA-based archival storage system

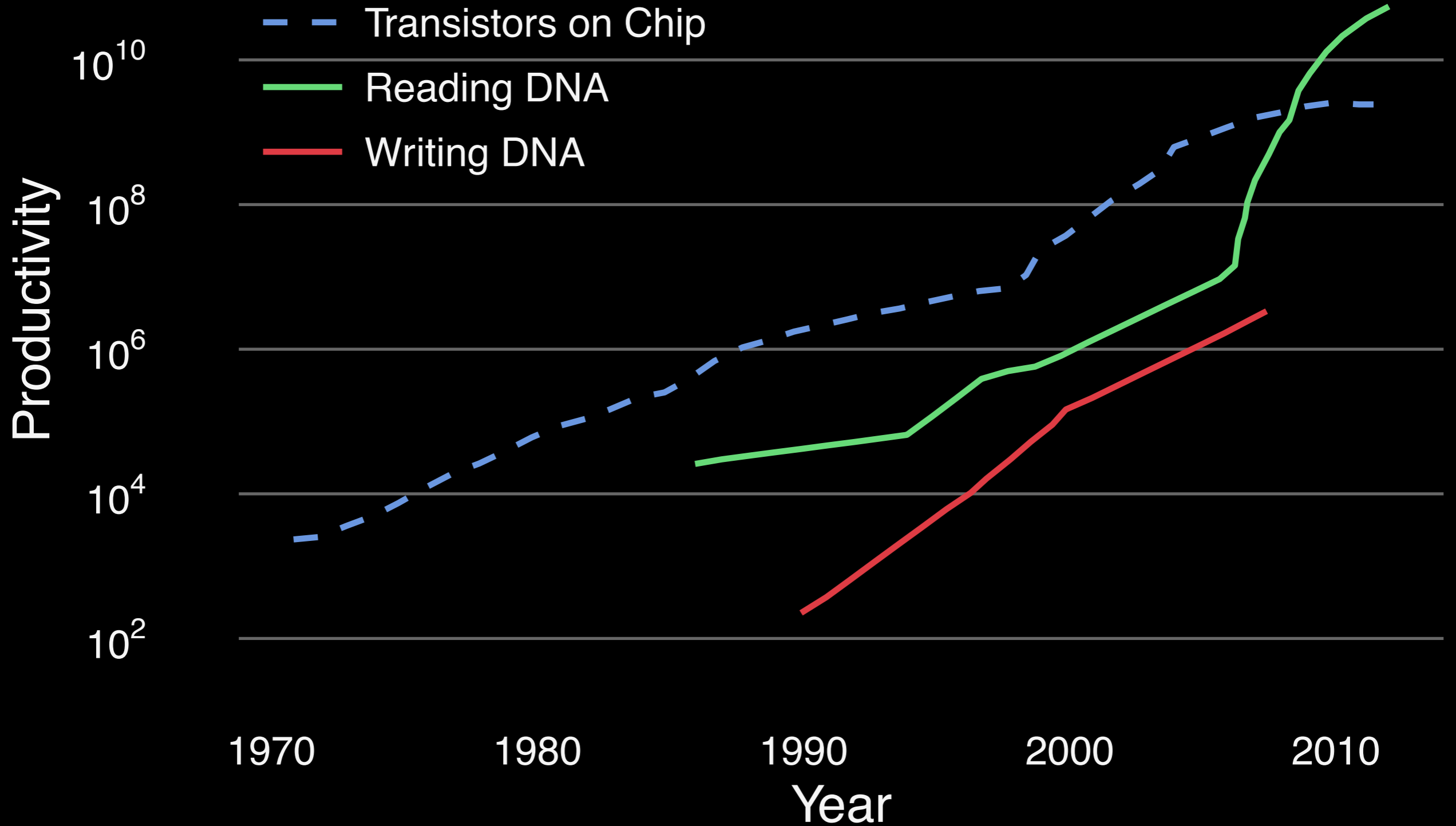


Also in the paper:

- Reliability-density trade-off
- Simulation of decay over time
- Error analysis
- Model of truncated strands

MBs/week → **GBs/second**

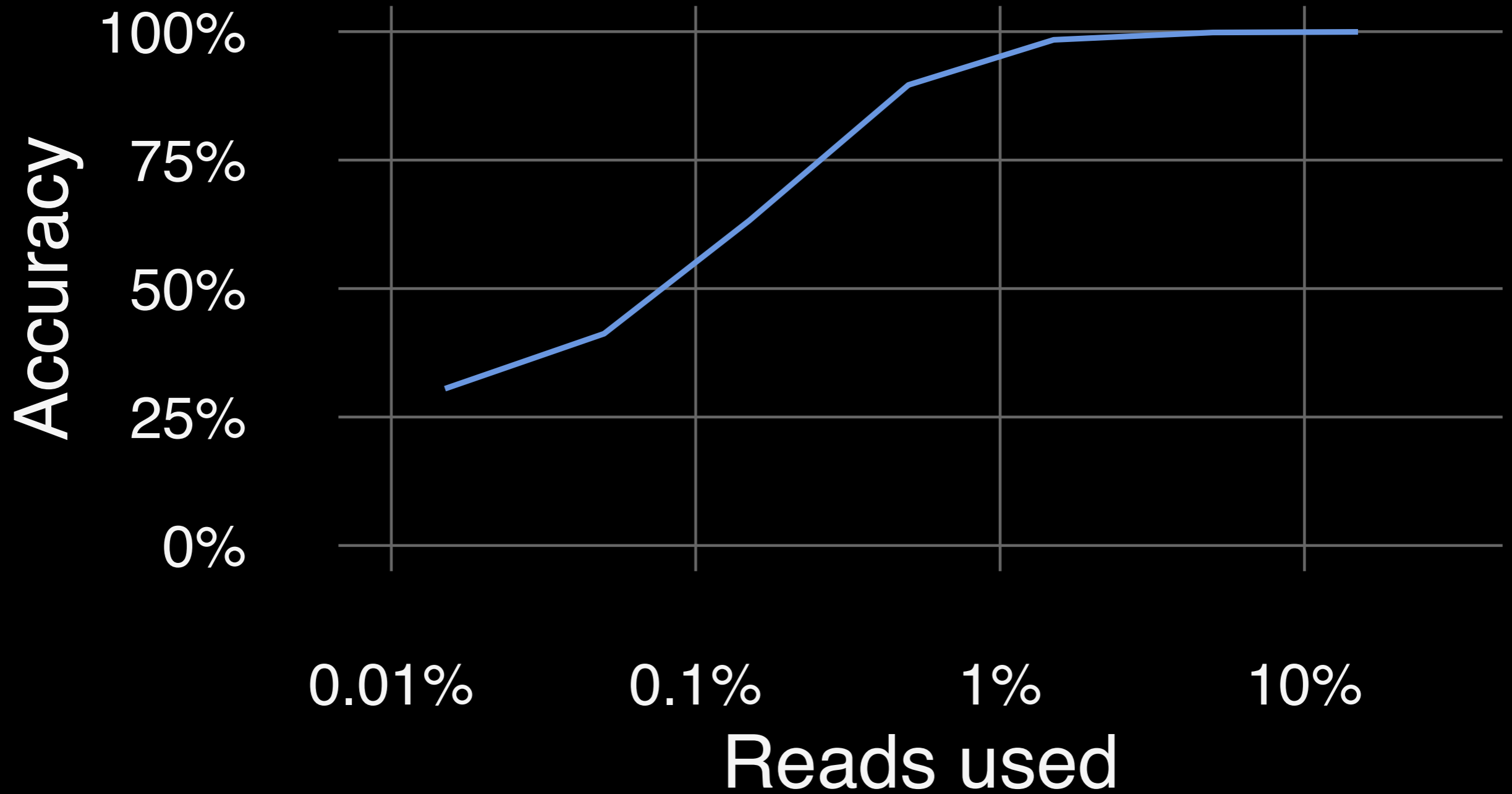
DNA productivity is growing



DNA technology is miniaturizing



We've just barely scratched the surface



Our community has seen these challenges before

Simulation

Cache locality

Latency-hiding
optimizations

Scheduling

Error correction

Spatial addressing

Circuit design

Programming
with errors

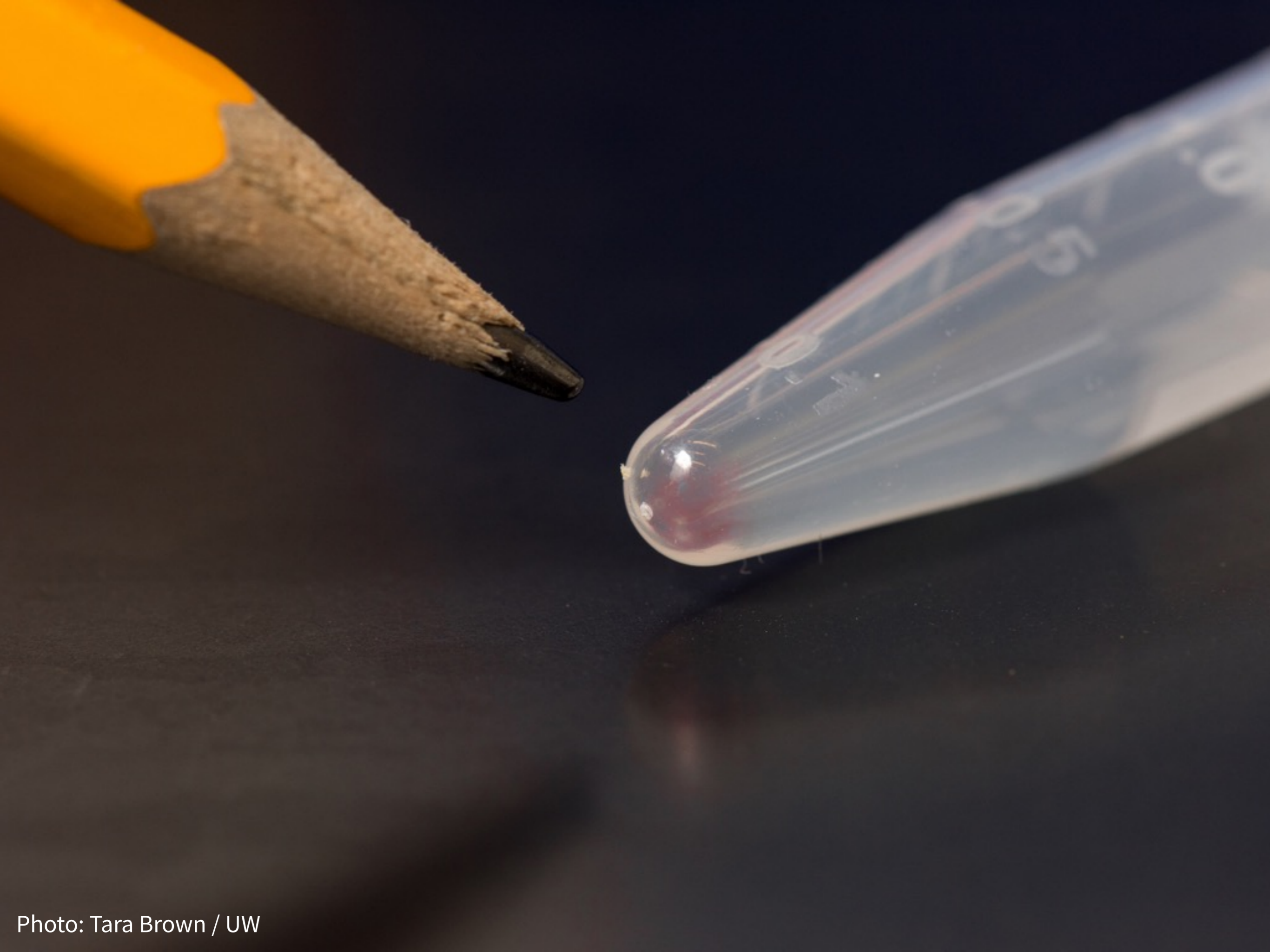


Photo: Tara Brown / UW