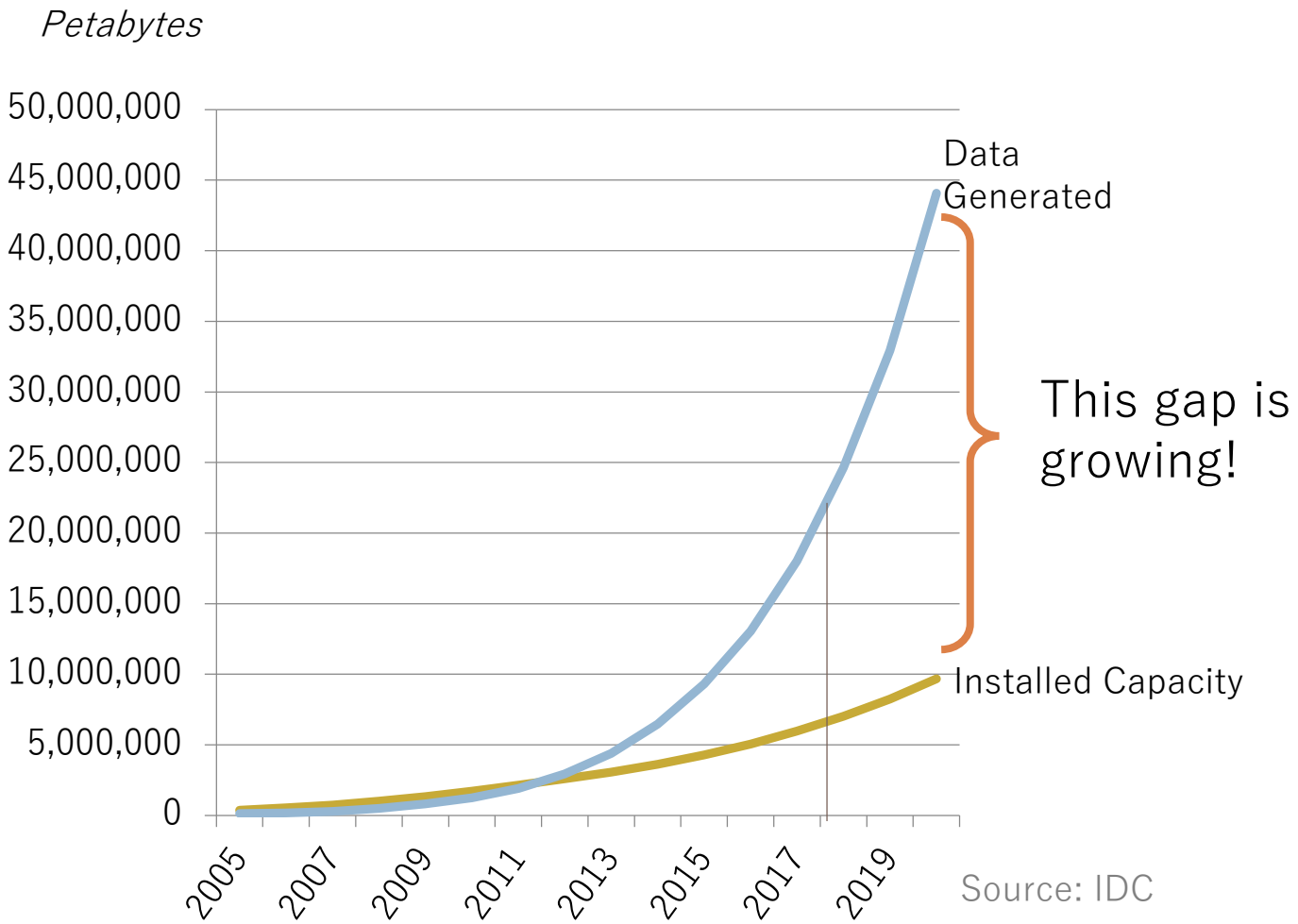
Algorithmic Challenges in DNA Data Storage

Cyrus Rashtchian

Data Science Fellow
Computer Science & Engineering and QI

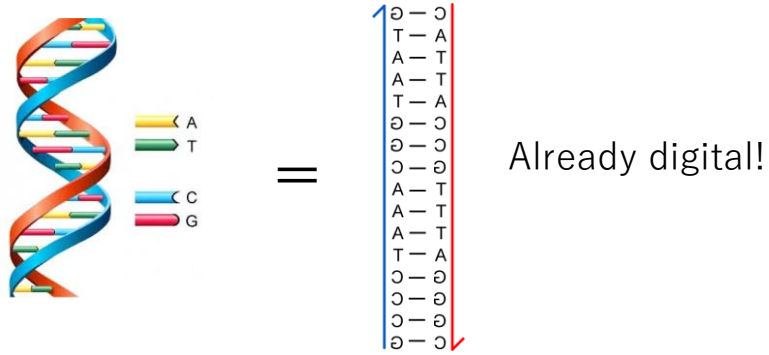
UC San Diego

Storage capacity is growing too slowly



Source: IDC

Can we bridge this hardware gap with “wetware”?



Why DNA?

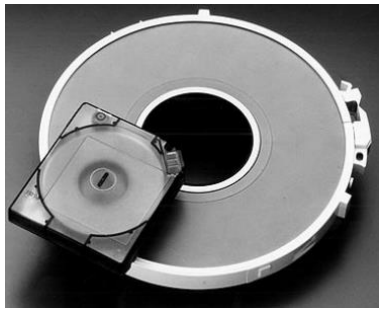
- DNA is the information system for life
- Extremely **durable** – oldest recovered genome over 700,000 years
  - half-life > 500 years

Millar, C. D. and Lambert, D. M. *Nature* 499, 34–35

DNA ⇅ Amino Acids ↓ Protein

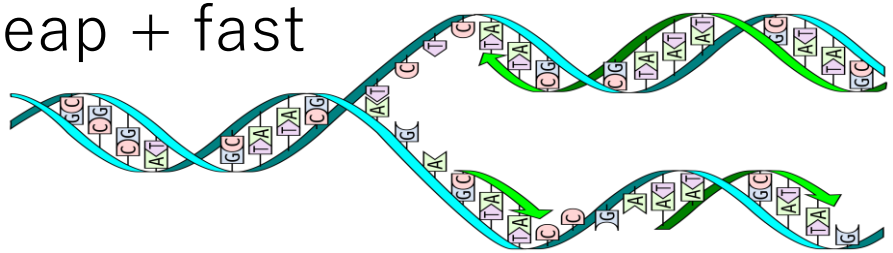
DNA ⇅ Bits ↓ File

- Readers **never obsolete**



- 10^3-10^4 times **denser** than magnetic tape

- **Copying**: cheap + fast




DNA Data Storage
Dense: 1 exabyte in 1 in^3
Durable: 100+ years
Photo: Tara Brown / UW

Ultimate Storage Hierarchy

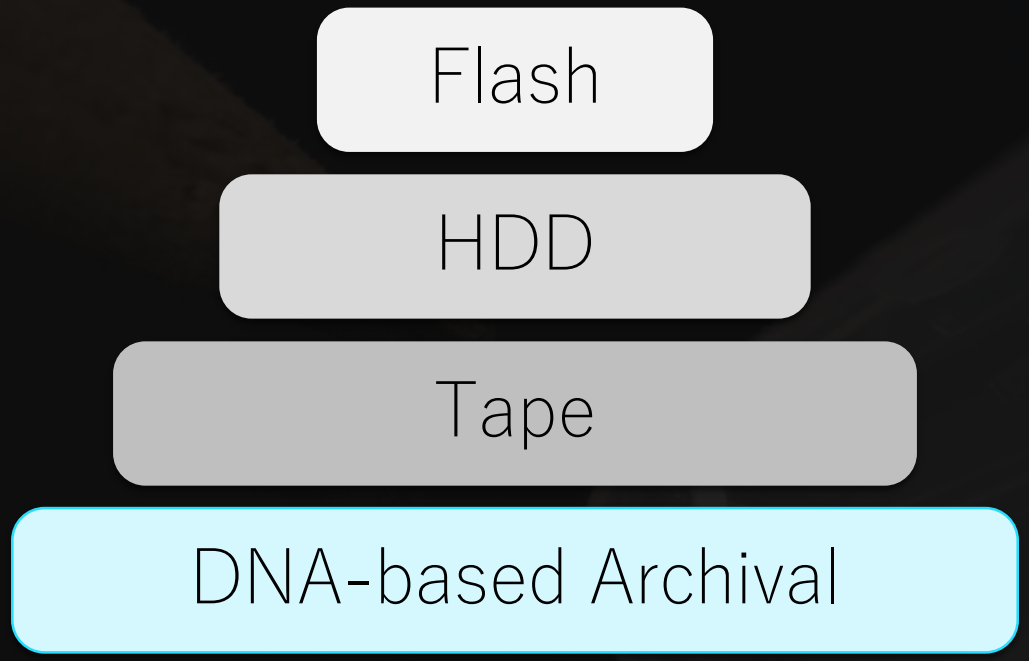


Photo: Tara Brown / UW

History

1960s – *"There is plenty of room at the bottom"* -- Feynman

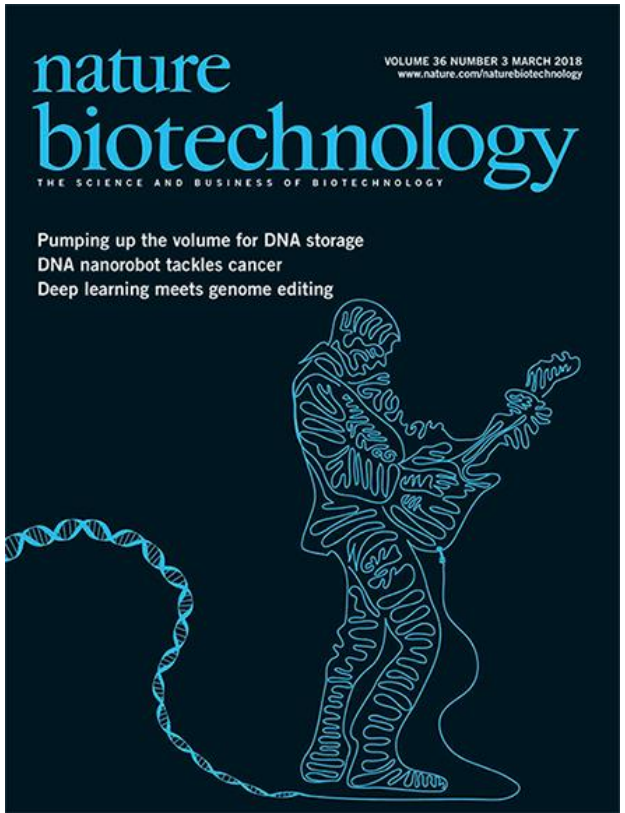
…

2012 – Church, Gao, Kosuri, *Science*
2013 – Goldman *et. al.*, *Nature*

} Visionary, Small-scale (KBs)

2015 – Grass *et. al.;* Yazdi *et. al.*
2016 – Bornholt *et. al.;* Blawat *et. al.;* Erlich-Zielinski

} Error-correction
Random Access
Biochemical Advances



Organick *et. al.*, … **R.** …, *Nature Biotech,* March 2018

- 400MB+
- Large-scale random access
- Robust data retrieval
- New algorithmic ideas

MISL

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

Microsoft Research

Today

Part 1: Storing Digital Data in Synthetic DNA

Part 2: Clustering for Data Retrieval

Part 3: Future Directions: Algorithmic and Molecular

DNA Data Storage

DNA Data Storage

Encode

```
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
        ...

TCGTATCGTCGCGTACGT
```

DNA Data Storage

Encode

```
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
        ...
TCGTATCGTCGCGTACGT
```

Synthesize
(write)

DNA Data Storage

Encode

```
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
        ...
TCGTATCGTCGCGTACGT
```

Synthesize
(write)

Amplify &
Sequence
(read)

```
ACGTCCAATACGGATCC-
GC-TTCATCGATTATCAAG
        ...
AGTATCGTCGCGTACGT
```

DNA Data Storage

Encode

```
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
        ...
TCGTATCGTCGCGTACGT
```

Synthesize
(write)

Amplify &
Sequence
(read)

```
ACGTCCAATACGGATCC-
GC-TTCATCGATTATCAAG
        ...
AGTATCGTCGCGTACGT
```

Insertions, Deletions,
and Substitutions
(2-12% error)

DNA Data Storage

Encode

```
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
        ...
TCGTATCGTCGCGTACGT
```

Synthesize
(write)

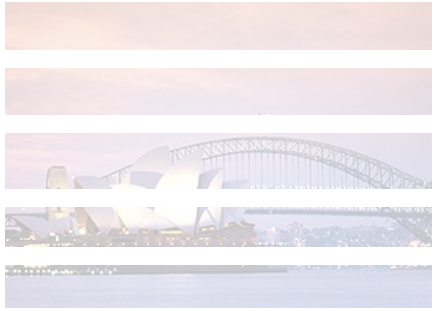
Amplify &
Sequence
(read)

Insertions, Deletions,
and Substitutions
(2-12% error)

```
ACGTCCAATACGGATCC-
GC-TTCATCGATTATCAAG
        ...
AGTATCGTCGCGTACGT
```

Decode

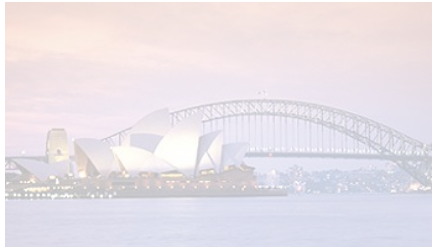
DNA Data Storage



```
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
        ...
TCGTATCGTCGCGTACGT
```

Insertions, Deletions, and Substitutions (2-12% error)

```
ACGTCCAATACGGATCC-
GC-TTCATCGATTATCAAG
        ...
AGTATCGTCGCGTACGT
```



Edit Distance: min # in/del/sub

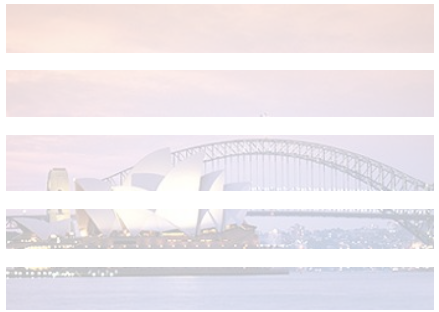
```
- E L E P H A N T
T E L E P H O N E
```

ED = 3

```
1010101010
0101010101
```

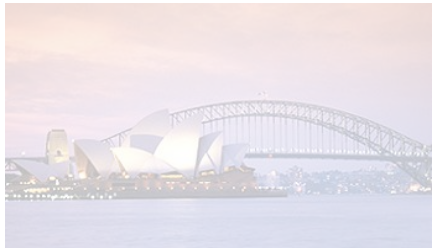
ED = 2

DNA Data Storage



```
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
        ...
TCGTATCGTCGCGTACGT
```

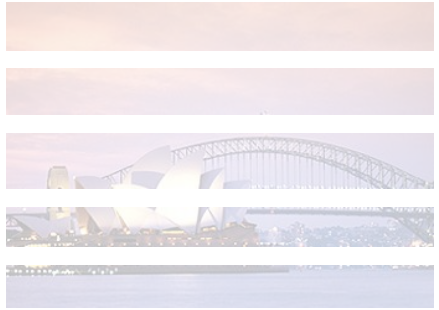

```
ACGTCCAATACGGATCC-
GC-TTCATCGATTATCAAG
        ...
AGTATCGTCGCGTACGT
```



each strand ≈ 10 Bytes

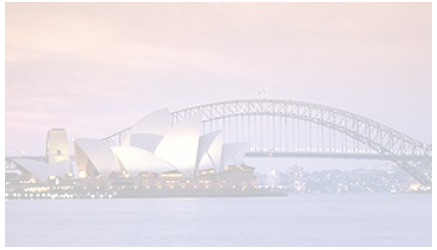
1GB ≈ 100M strands

DNA Data Storage



```
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
        ...

TCGTATCGTCGCGTACGT
```

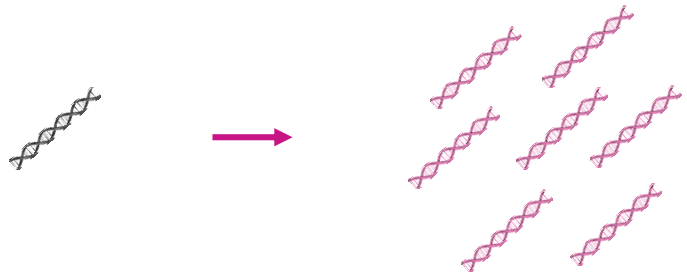

```
ACGTCCAATACGGATCC-
GC-TTCATCGATTATCAAG
        ...

AGTATCGTCGCGTACGT
```



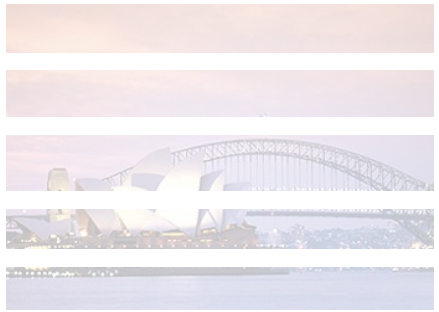
each strand ≈ 10 Bytes

1GB ≈ 100M strands

copy strands ≈ 10 times

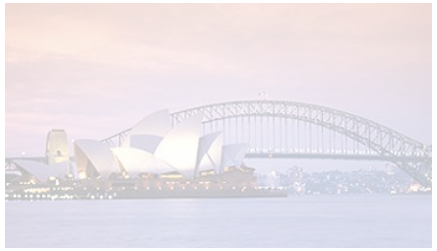


DNA Data Storage

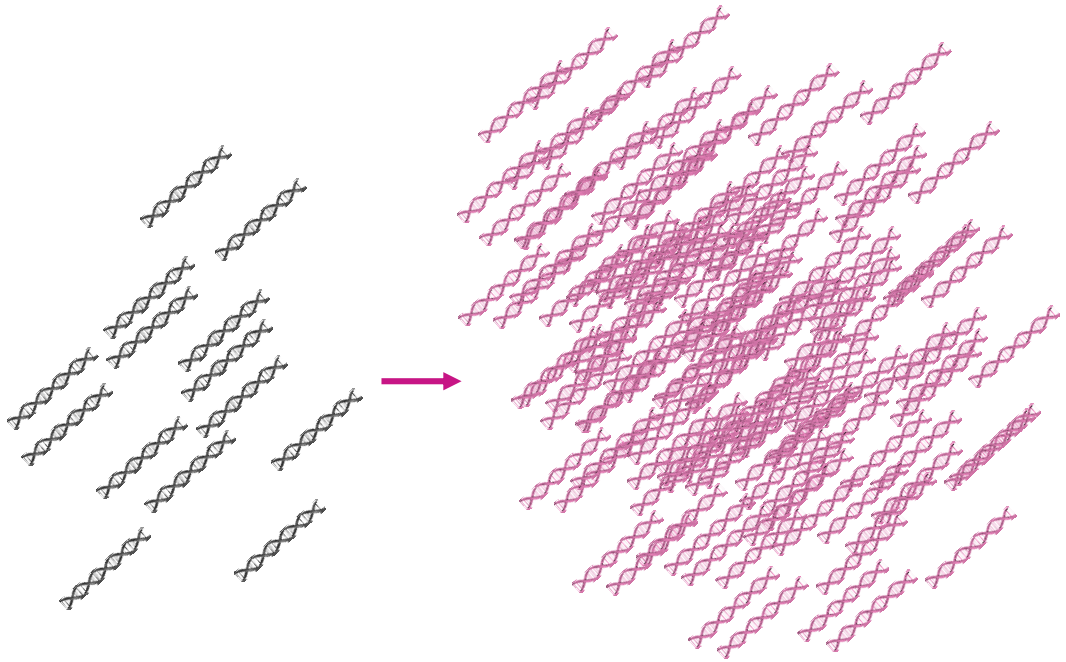


ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
...
TCGTATCGTCGCGTACGT

ACGTC**C**AATACGG**A**TCC-
GC-TTCATCGATTATCAA**G**
...
AGTATCGTCGCGTACGT

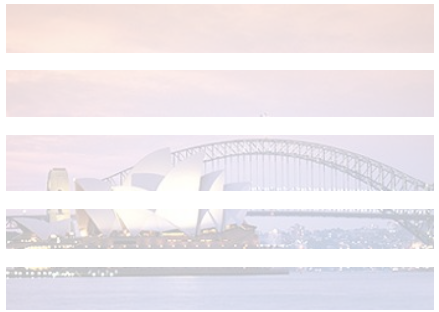


each strand ≈ 10 Bytes
1GB ≈ 100M strands



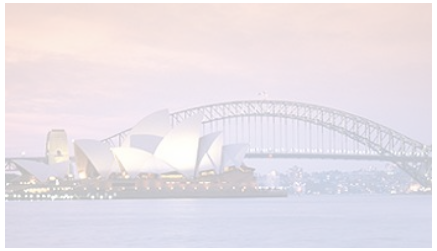
1 GB → 1 Billion Reads

DNA Data Storage



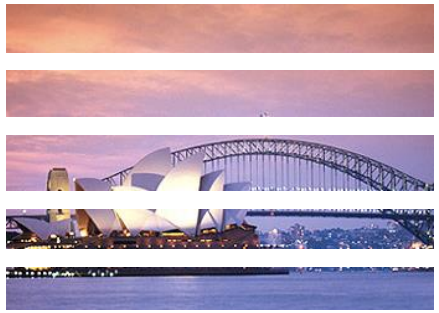
ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
...
TCGTATCGTCGCGTACGT

ACGTC**C**AATACGG**A**TCC-
GC-TTCATCGATTATCAA**G**
...
AGTATCGTCGCGTACGT



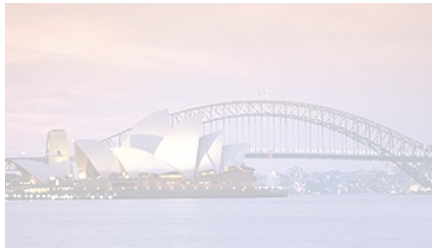
1 GB → 1 Billion Reads

DNA Data Storage



ACGTCGAATACGGCTCCA
GCATTCATCGATTATCAA
...
TCGTATCGTCGCGTACGT

ACGTCCAATACGGATCC-
GC-TTCATCGATTATCAAG
...
AGTATCGTCGCGTACGT



Randomize Data

111…111001…000000…
⊕ random string

001…001101…111011…

001…001 | 101…111 | 011 …

1 GB → 1 Billion Reads

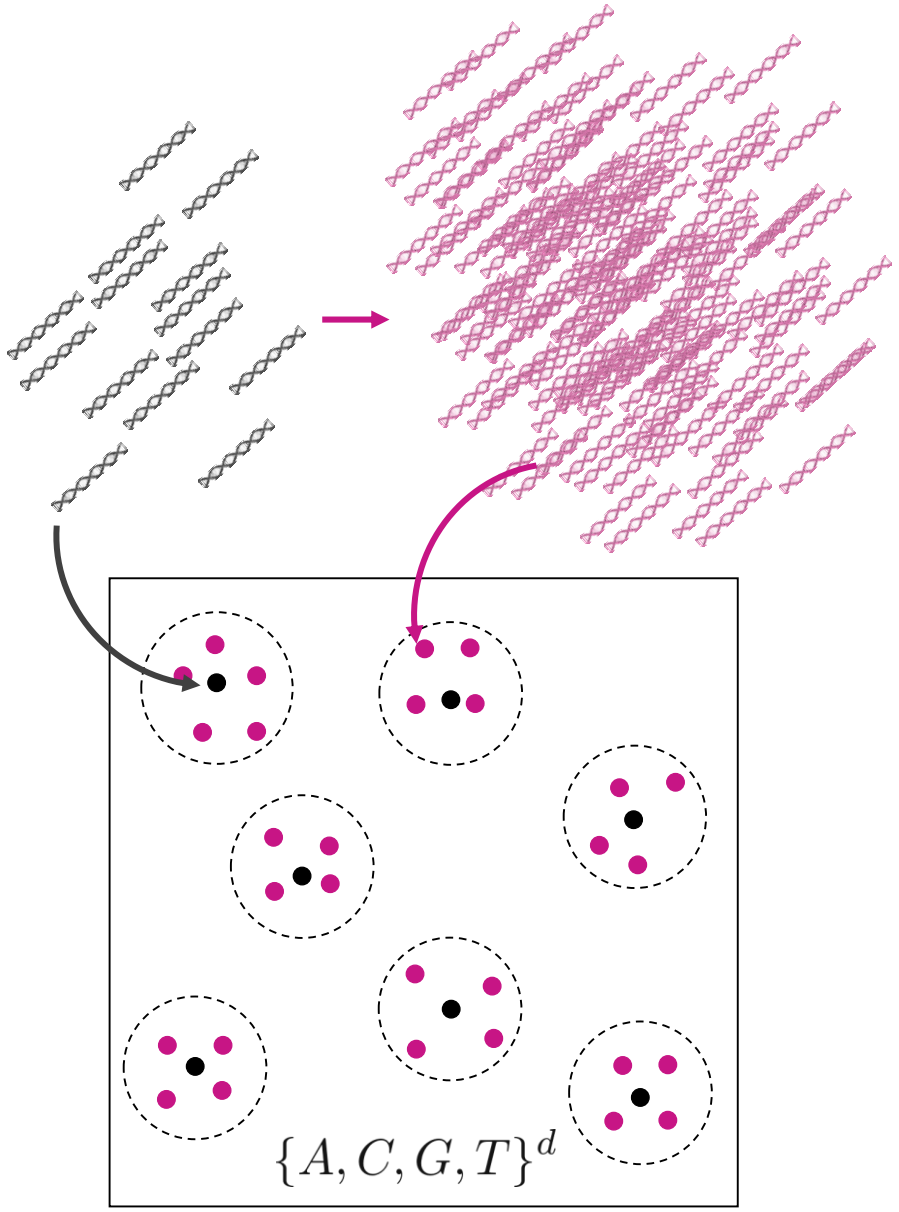
Geometry

Initial Strands = Cluster Centers

Reads = Noisy Copies

Randomization $\Longrightarrow$ Well-separated

Clusters → Centers → Decode → Original file



Clustering Challenges

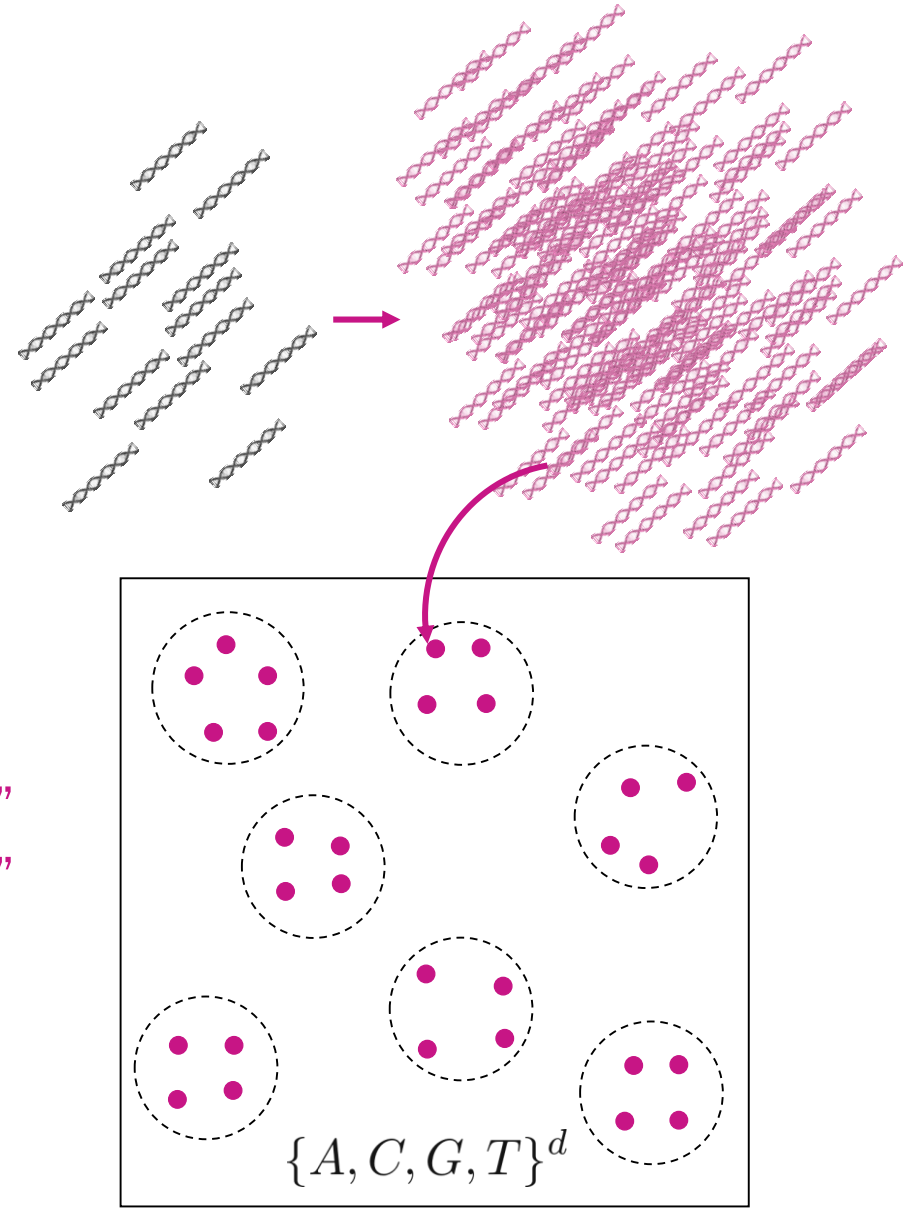
edit distance is hard

n vectors in high-dim space

clusters $k = \Omega(n)$

"Extreme Clustering"
"Micro-Clustering"

Known algorithms $O(n^2)$ time



Our Algorithm

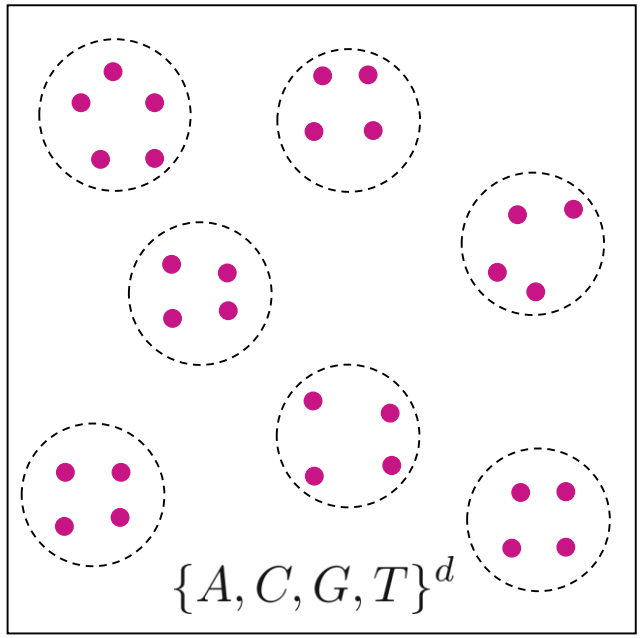
Clustering Billions of Reads for DNA Data Storage
R., Makarychev, Racz, Dumas Ang, Jevdjic, Yekhanin, Ceze, Strauss, NIPS 2017

Nearly-linear time

Scales and distributes well

New hashing and embedding

10-1000x speedup + higher accuracy



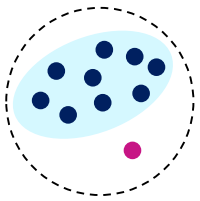
Clustering Problem Statement

Model: random centers & ~10 random copies

$k = n/10$ clusters

Task: quickly compute a highly-accurate clustering

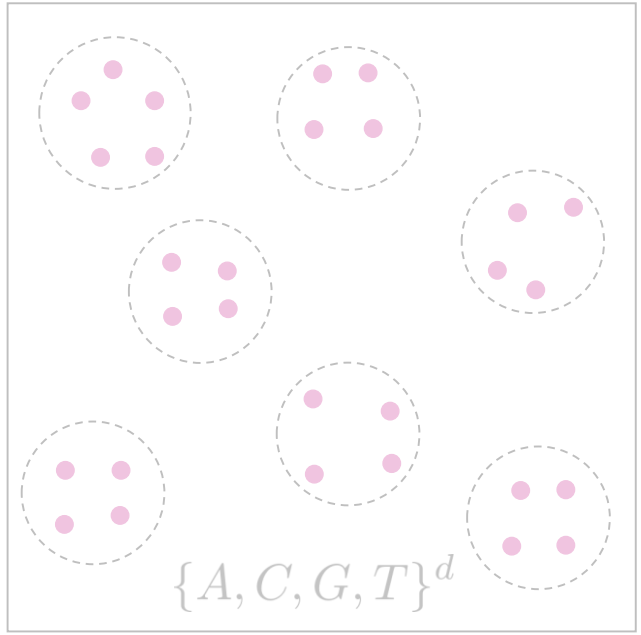
Accuracy: fraction of mostly correct clusters



$$\mathcal{A}_\gamma(\mathbf{C}, \widetilde{\mathbf{C}}) = \max_\pi \frac{1}{|\mathbf{C}|} \sum_{i=1}^{|\mathbf{C}|} \mathbf{1}_{\{\widetilde{C}_{\pi(i)} \subseteq C_i \text{ and } |\widetilde{C}_{\pi(i)}| \geq \gamma |C_i|\}}$$

Induced underlying clustering

$$\mathbf{C} = \{C_1, \ldots, C_k\}$$

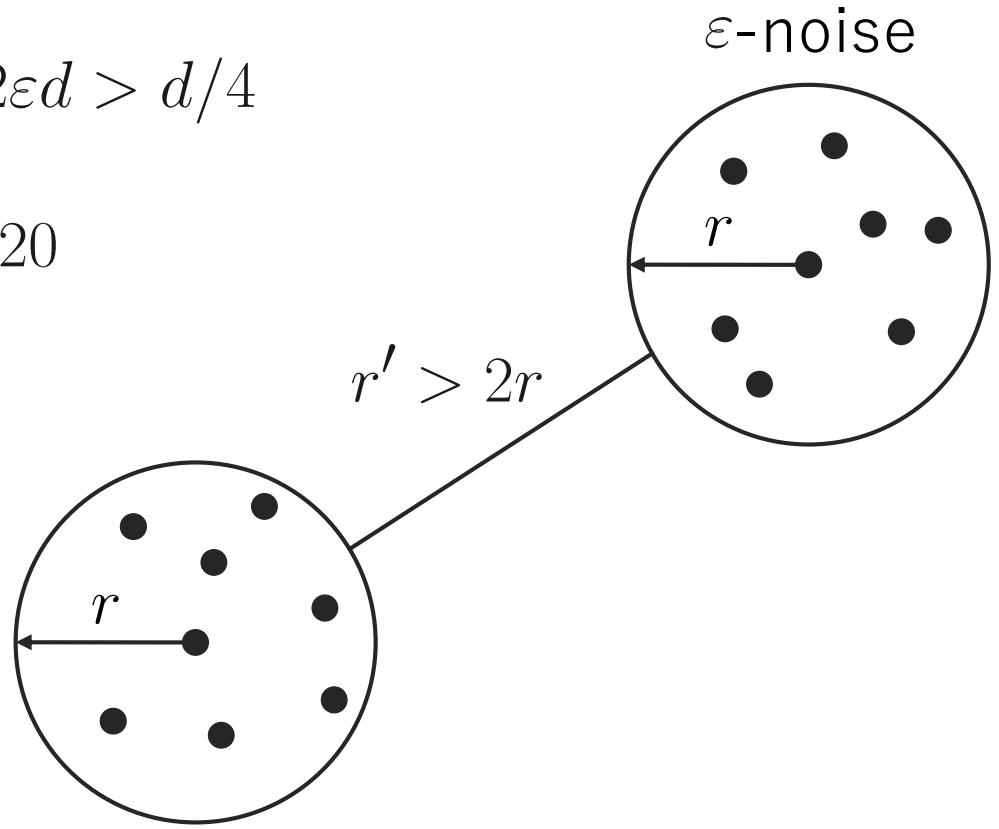


Separation

Defn: (r, r')-separated clusters

$$r' \approx d/2 - 2\varepsilon d > d/4$$

$$r = \varepsilon d \approx d/20$$

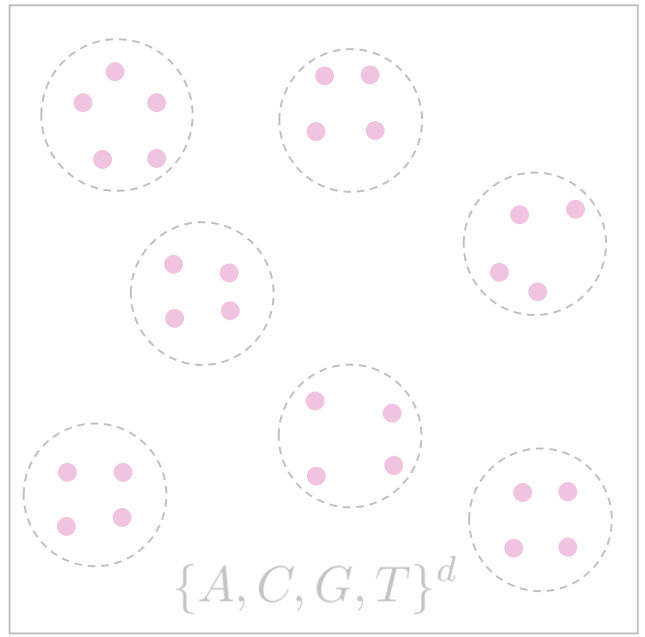


Real data

$$d = 150$$
$$r \leq 10$$
$$r' > 50$$

Induced underlying clustering

$$\mathbf{C} = \{C_1, \ldots, C_k\}$$



Good but slow algorithm

Compare all pairs of strings

Connected components of $r-$NN graph

$O(n^2)$ comparisons

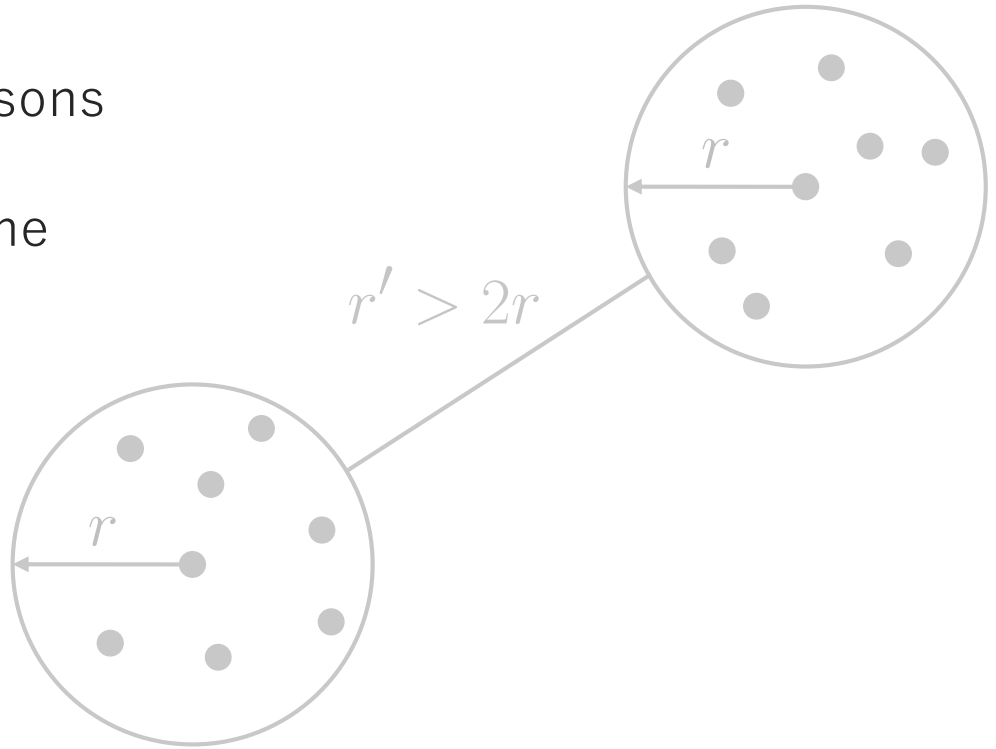
each $O(rd)$ time

Real data

$d = 150$

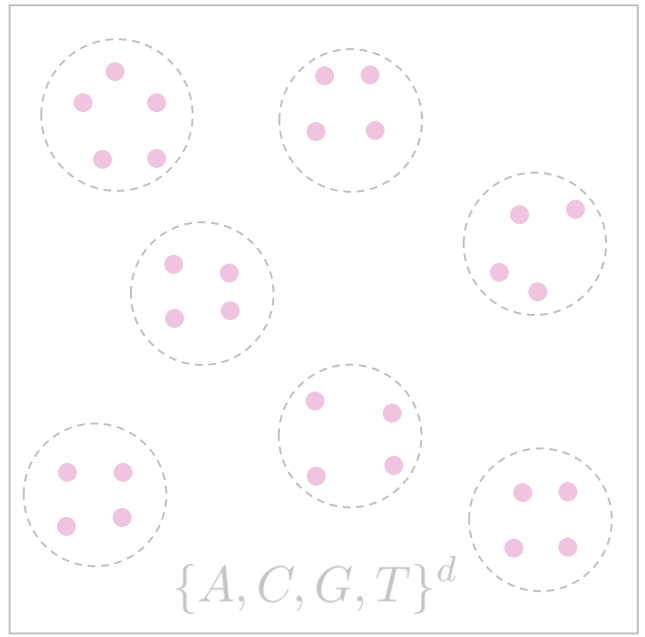
$r \leq 10$

$r' > 50$



Induced underlying clustering

$\mathbf{C} = \{C_1, \ldots, C_k\}$



Our Algorithm

(0) Initialize each string as singleton cluster

(1) Hash cluster representatives into buckets based on similarity

(2) Within a bucket

i. Compute Hamming distance of binary signatures

- Merge if very small Hamming distance
- Ignore if very large Hamming distance

ACCGTTAATCC
CCGTTAATCC
vs
0010001100110
0000001100110

ii. Compute edit distance when ambiguous; merge if close

(3) Return to step (1)

Key Ideas

Hash

close in edit distance $\Longrightarrow$ same bucket

efficient to compute

} Not quite LSH (crucially uses randomness)

Binary Signatures

edit distance $\approx$ Hamming distance

cheaper than edit dist. (within bucket)

} Not quite an embedding (crucially uses randomness)

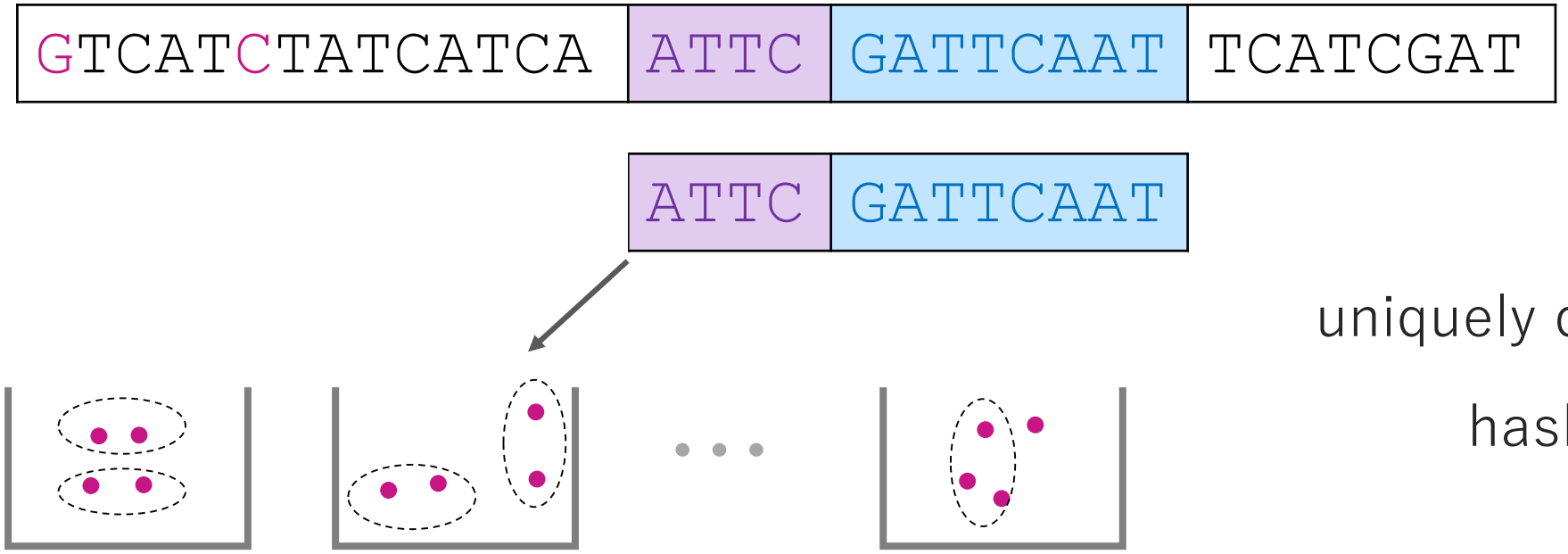
Hash

Intuition: similar strings share substring

Idea: random synchronized substrings

- **Random anchor** (same for all strings)
- **Next several characters**

MinHash-inspired
[Broder 90s]
w/ Anchors
for Edit Dist



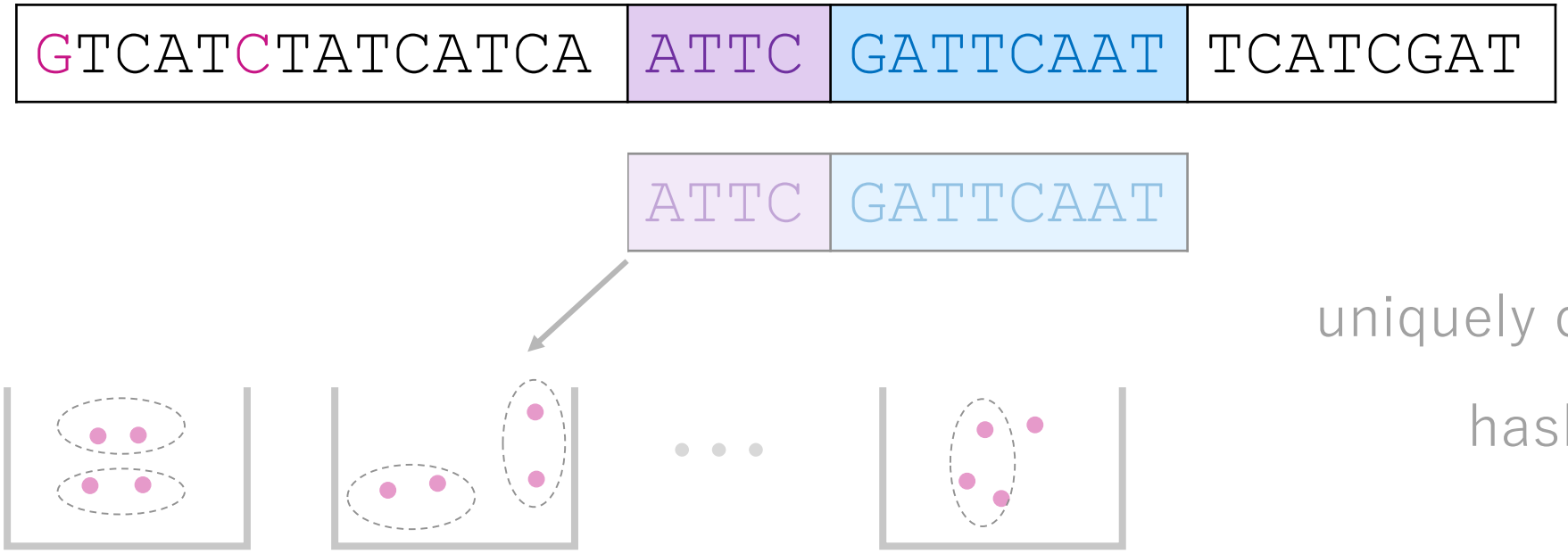
uniquely determine string

hash length $\geq \log_4(n)$

Hash (in practice)

Choose multiple anchors (fast to check presence)

4 chars + 12 chars = 16 chars (32 bits)

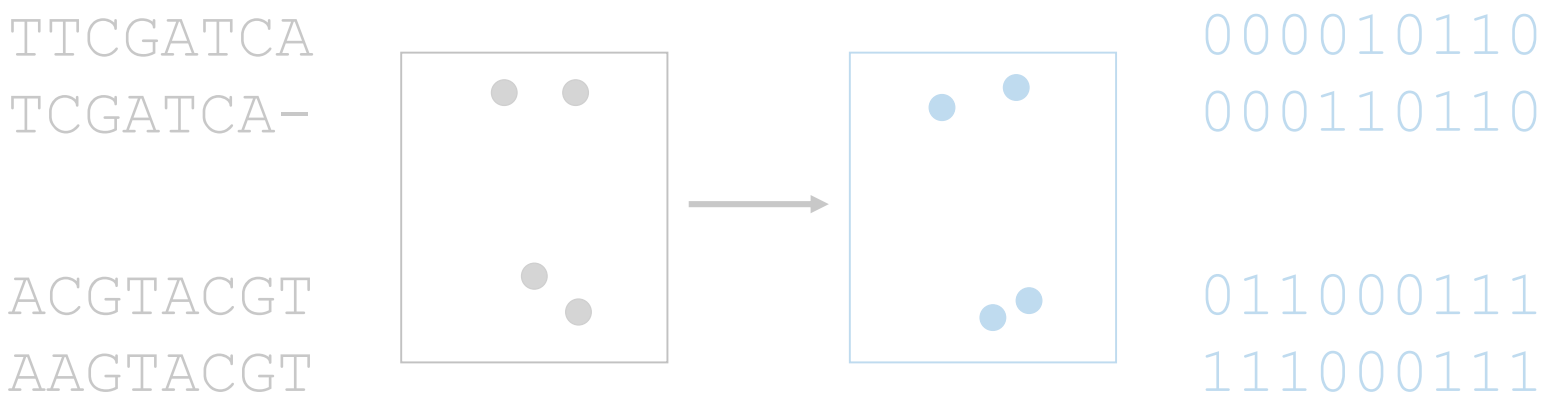


uniquely determine string

hash length $\geq \log_4(n)$

Dense Binary Coding

Metric Embedding: map to easier space & roughly preserve distances



Rule of thumb
½ ones and ½ zeros
(middle weight)

Bad News: large distortion in general for many spaces (e.g., edit to Hamming)

Good News: preserve dist. up to $O(\log d)$ for **random** strings

Binary Embedding

Hamming distance between signatures approximates edit distance

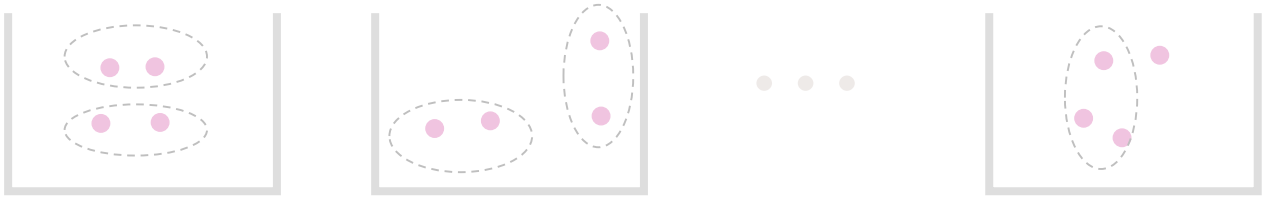
GTCATCTATCA	ATTCGATTCAAT	…

{GTC, TCA, CAT, ATC, …}

0001000010000010000001 …

GTC TCA CAT ATC

q-gram embedding [Ukkonen 80s]



Binary Embedding

Hamming distance between signatures approximates edit distance

GTCATCTATCA	ATTCGATTCAAT	…

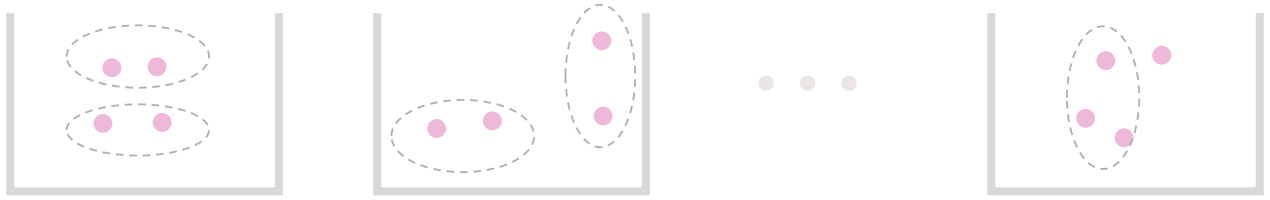
$$\sigma_q(x) \in \{0,1\}^{4^q}$$

{GTC, TCA, CAT, ATC, …}

0001000010000010000001 …

GTC TCA CAT ATC

q-gram embedding [Ukkonen 80s]



Claim 1 (close pairs stay close):

$$d_H(\sigma_q(x), \sigma_q(y)) \leq 2q \cdot d_E(x, y)$$

Claim 2 (random centers are far):

for random x, y , if $q > 2 \log d$

$$d_H(\sigma_q(x), \sigma_q(y)) \geq 2d - O(1)$$

with high probability

Binary Embedding

Hamming distance between signatures approximates edit distance

$\sigma_q(x) \in \{0,1\}^{4^q}$

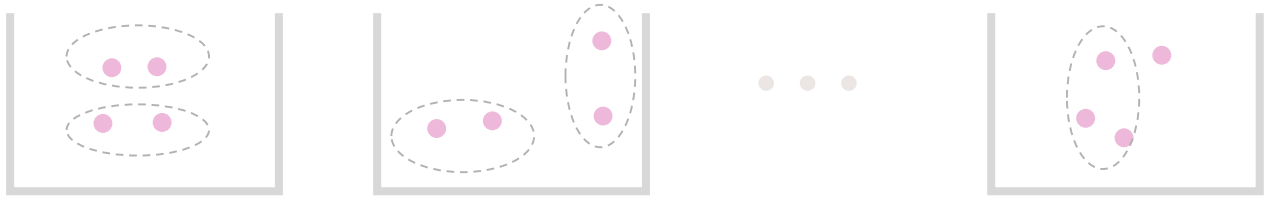
GTCATCTATCA	ATTCGATTCAAT	…

{GTC, TCA, CAT, ATC, …}

0001000010000010000001 …

GTC TCA CAT ATC

q-gram embedding [Ukkonen 80s]



Theory → Practice

Idea: use blocks

$q = 3 \implies$ 64-dim. bit-vector

use 32 chars / block

(roughly ½ ones and ½ zeros)

Convergence Theorem

Theorem: Compute 99.9% accurate clustering in time $n^2 \cdot (1/d)^{O(1/\varepsilon)}$

Proof:

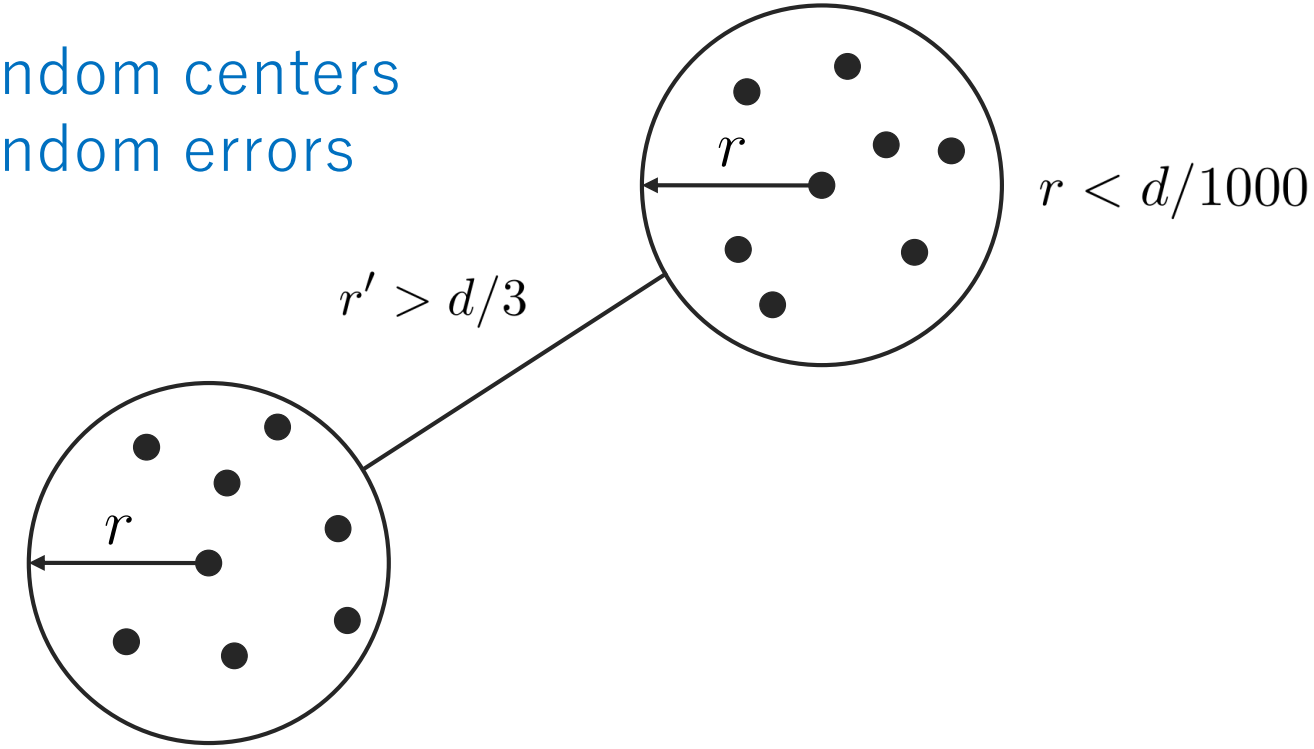
Number of iterations roughly $\frac{n}{d^{1/\varepsilon}}$

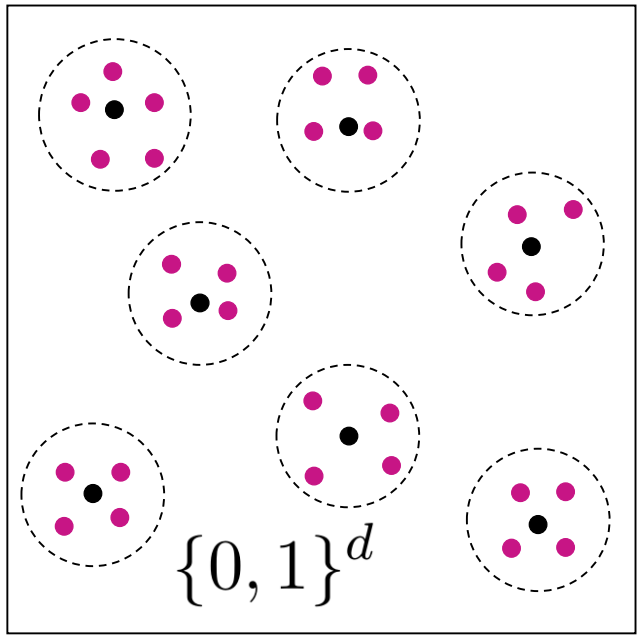
Each with $O(n)$ comparisons that take time $O(rd) = O(d^2)$

Question: can we get down to time $n^{1+\varepsilon} \cdot \mathrm{poly}(d)$???

Clustering in Hamming Distance

random centers
random errors



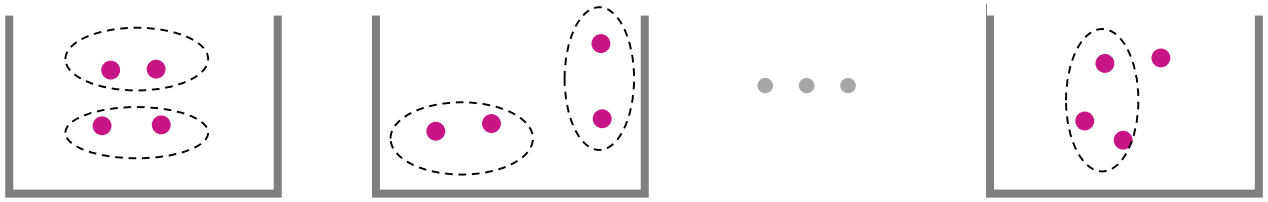


Question: Compute 99.9% accurate clustering in time $n \cdot \mathrm{poly}(d, \log n)$

Distributed Version

(0) Initialize each string as singleton cluster

(1) Hash cluster representatives into buckets based on similarity



(2) Within a bucket

i. Compute Hamming distance of binary signatures

- Merge if very small Hamming distance
- Ignore if very large Hamming distance

ii. Compute edit distance when ambiguous; merge if close

(3) Return to step (1)

Distributed Version

(0) Initialize each string as singleton cluster

(1) Hash cluster representatives into buckets based on similarity

Shuffle Current Clusters

Several local iterations for every global round

Balance comm. time and local comp. time

(2) Within a bucket

i. Compute Hamming distance of binary signatures

- Merge if very small Hamming distance
- Ignore if very large Hamming distance

ii. Compute edit distance when ambiguous; merge if close

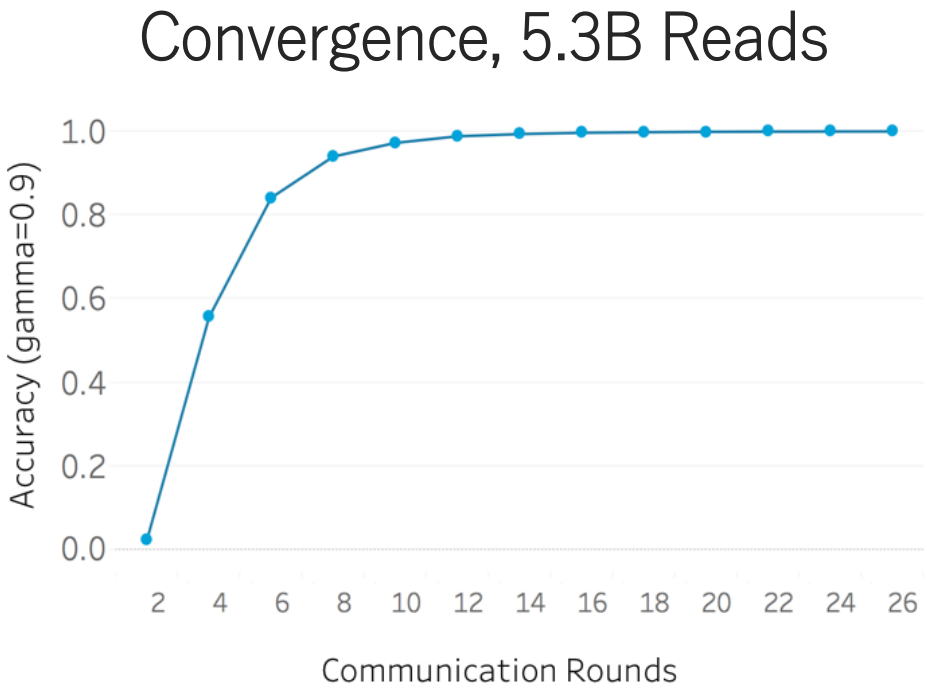
(3) Return to step (1)

Implementation & Experiments

MPI w/ RDMA → Batch shuffle (all-to-all, using non-blocking gets)

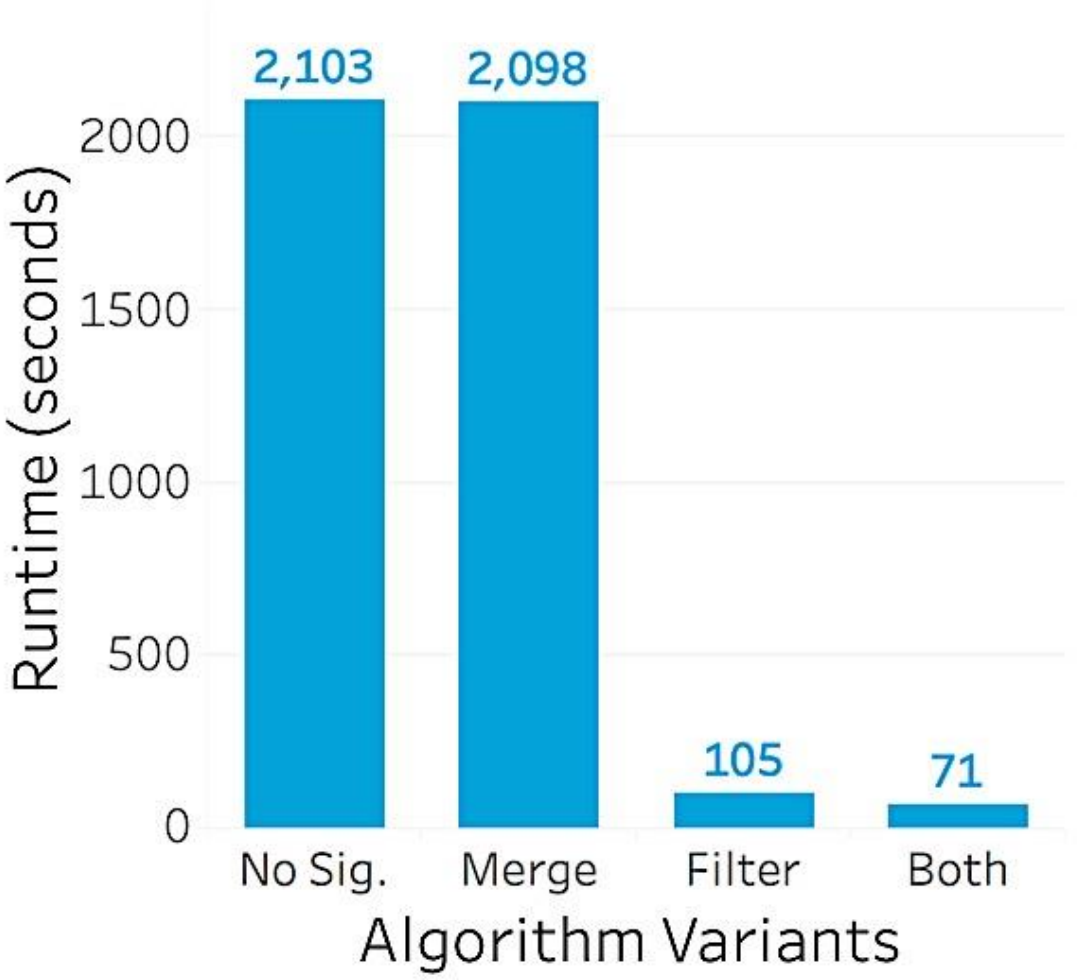
~26 shuffles, each ~30 local iterations

45 mins for 500GB (24 machines, 384 cores)
- ~23 min. communication
- ~19 min. local clustering



Binary Signature Performance

(3.1M Real Reads; Single Thread)



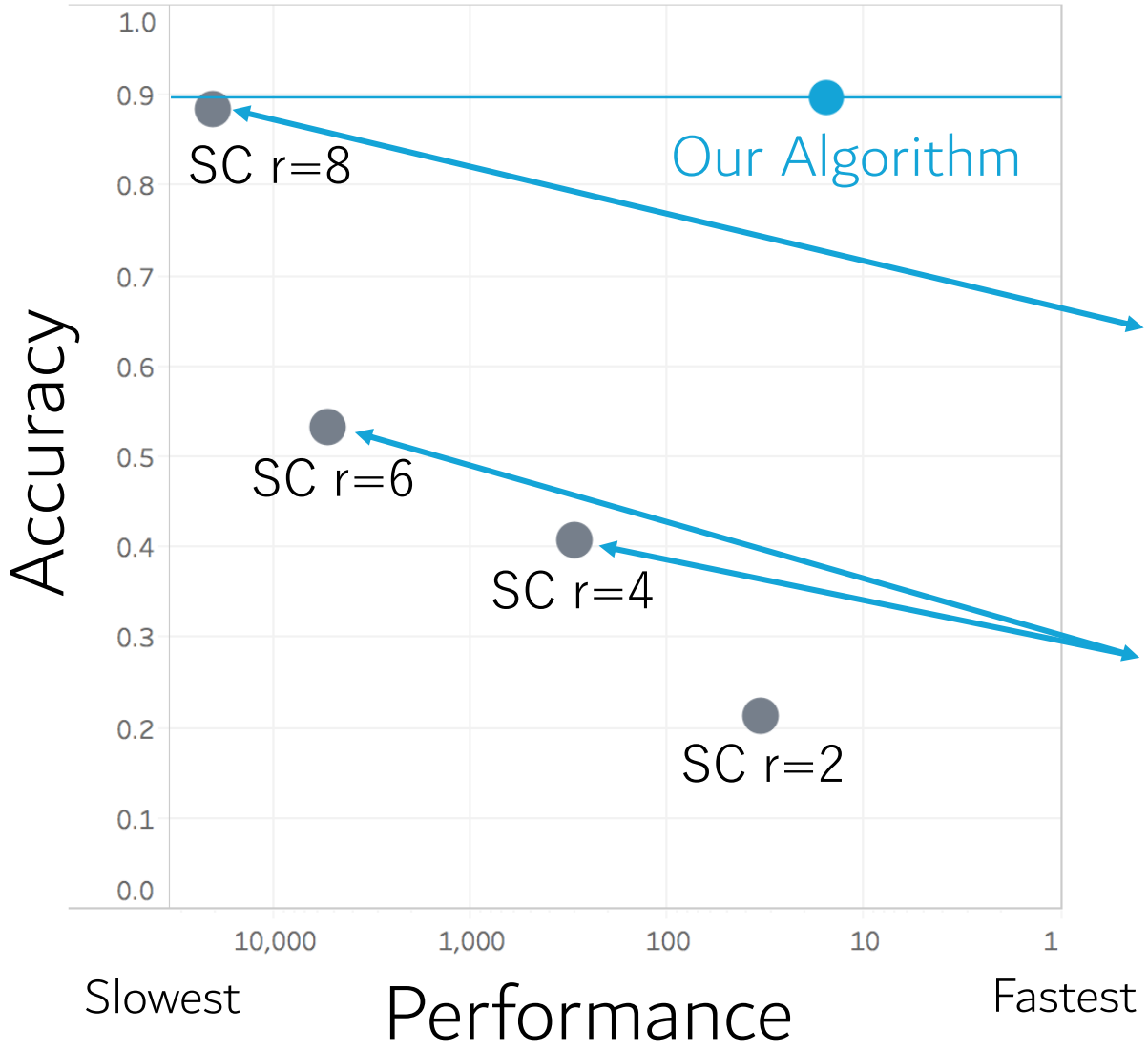
Outperforms Previous Methods

Comparison with StarCode*

*varying distance param. r = 2, 4, 6, 8

[Starcode: sequence clustering based on all-pairs search, Zorita, Cuscó, Filion, *Bioinformatics,* 2015]

Outperforms Previous Methods



Comparison with StarCode*

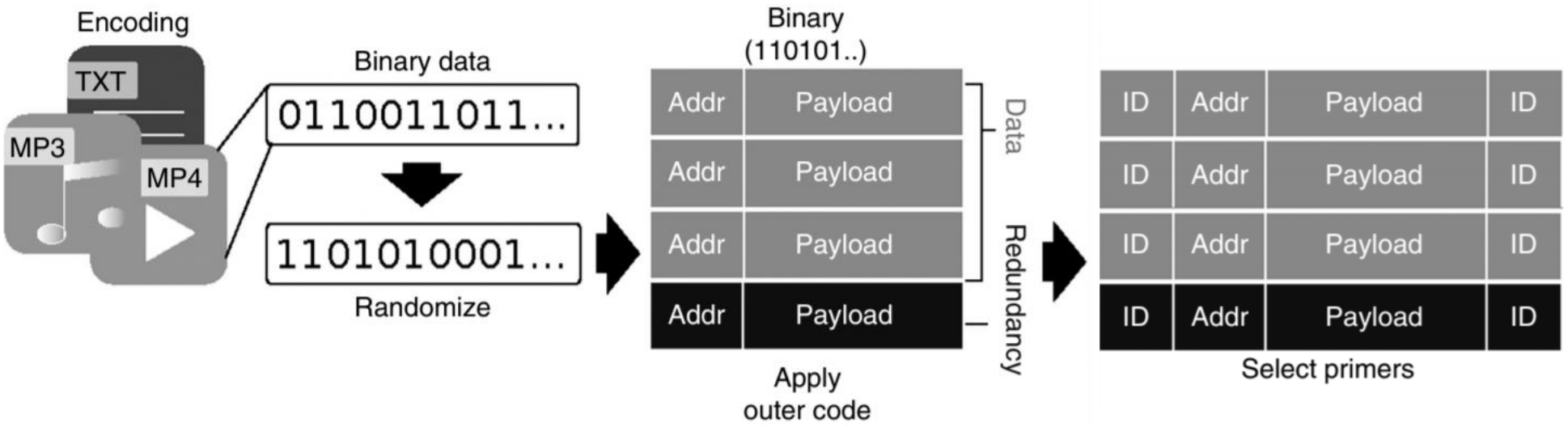
*varying distance param. r = 2, 4, 6, 8

1000x Speed-up
& Slightly Better Accuracy

10-100x Speed-up
& Much Higher Accuracy

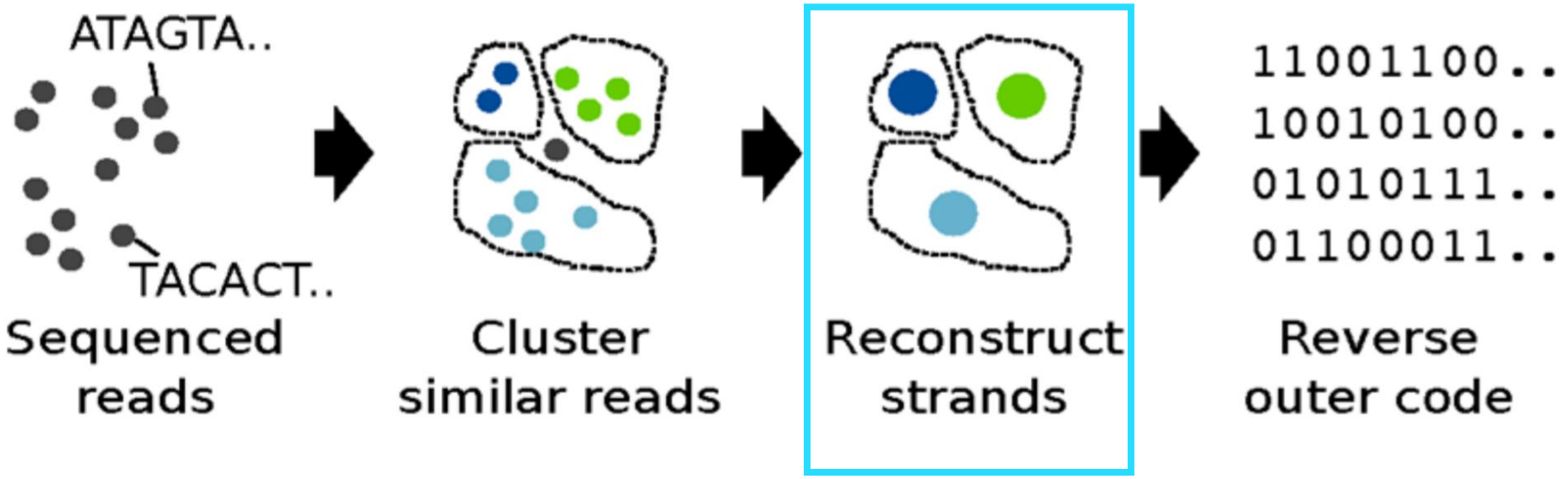
[Starcode: sequence clustering based on all-pairs search, Zorita, Cuscó, Filion, *Bioinformatics,* 2015]

Storing Data



Organick et. al., *Nature Biotech,* 2018

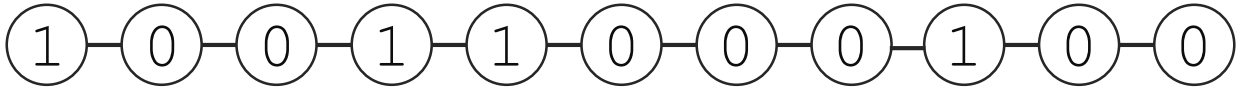
Retrieving Data



Organick et. al., *Nature Biotech,* 2018

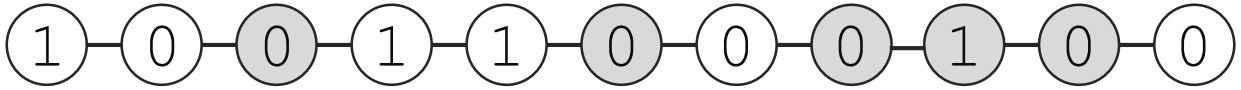
Trace Reconstruction

unknown worst-case string n bits

X 

Trace Reconstruction

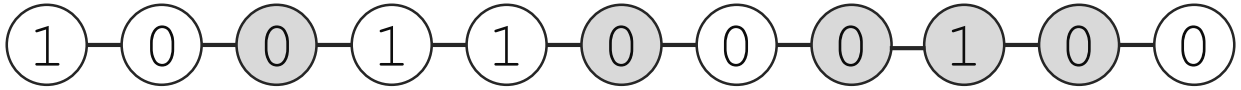
unknown worst-case string n bits

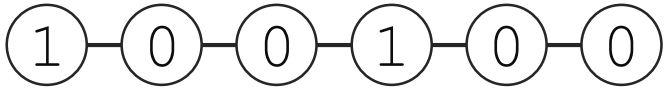
X 

Deletion channel, probability 0.5

Trace Reconstruction

unknown worst-case string n bits

X 

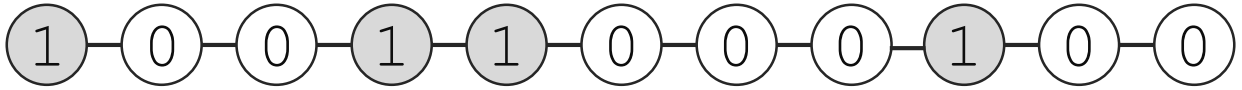


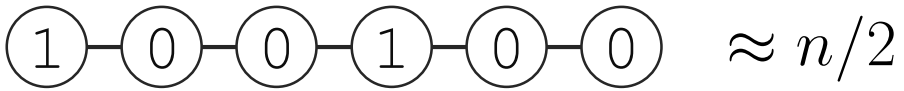
$\approx n/2$

Deletion channel, probability 0.5

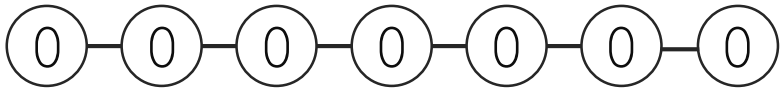
Trace Reconstruction

unknown worst-case string n bits

X 



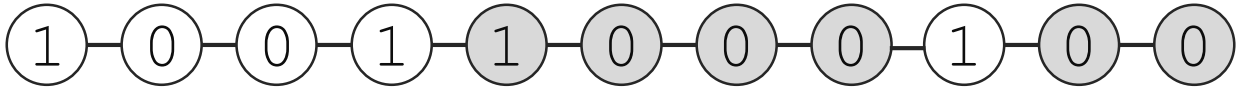
$\approx n/2$

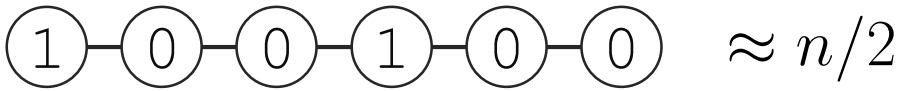


Deletion channel, probability 0.5

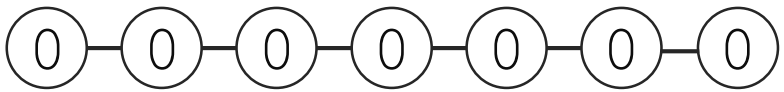
Trace Reconstruction

unknown worst-case string n bits

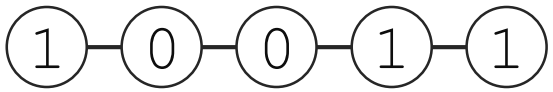
X 



$\approx n/2$



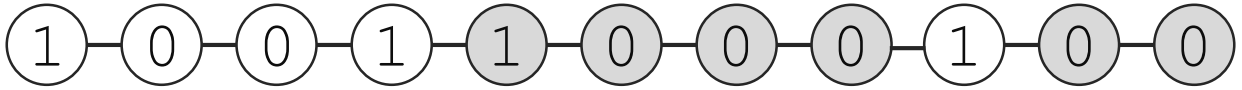
...

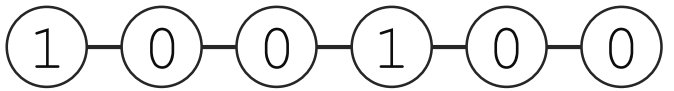


Deletion channel, probability 0.5

Trace Reconstruction

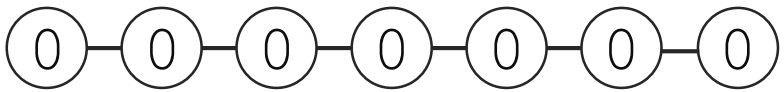
unknown worst-case string n bits

X 



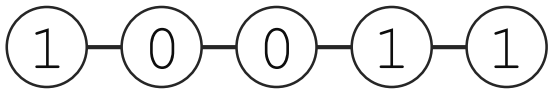
$\approx n/2$

Deletion channel, probability 0.5



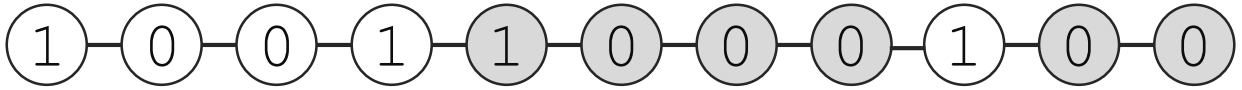
Goal: Recover X w.h.p. using min # traces

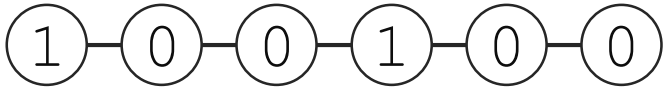
...



Trace Reconstruction

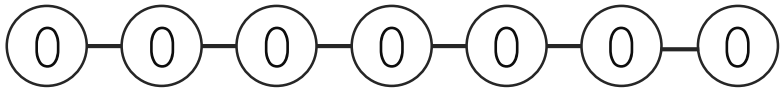
unknown worst-case string n bits

X 



$\approx n/2$

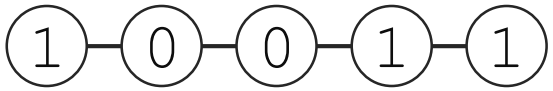
Deletion channel, probability 0.5



Goal: Recover X w.h.p. using min # traces

…

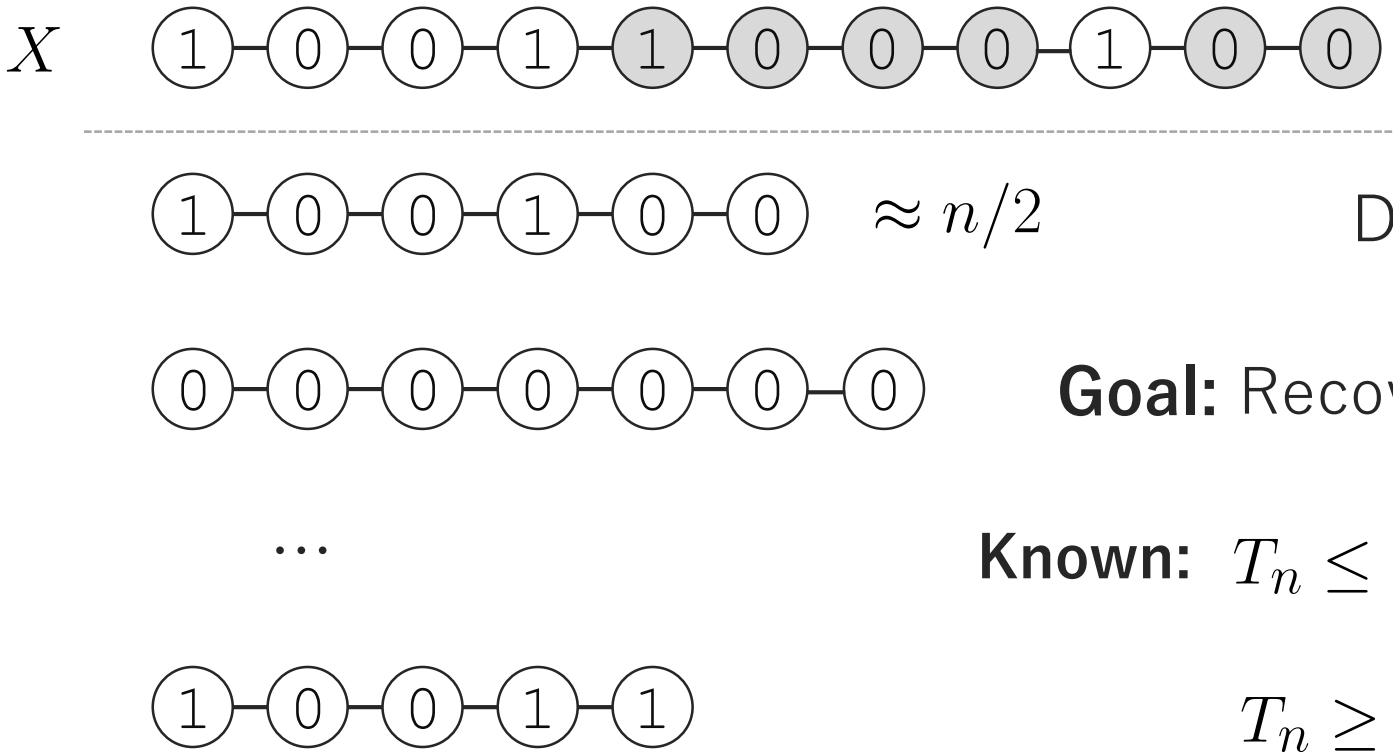
Known: $T_n \leq \exp\left(n^{1/3}\right)$ [Nazarov-Peres '16; De, O'Donnell, Servedio '16]



$T_n \geq \widetilde{\Omega}\left(n^{5/4}\right)$ [Holden-Lyons '18]

Trace Reconstruction

unknown worst-case string n bits



Deletion channel, probability 0.5

Goal: Recover X w.h.p. using min # traces

Known: $T_n \leq \exp\left(n^{1/3}\right)$ [Nazarov-Peres '16; De, O'Donnell, Servedio '16]

$T_n \geq \widetilde{\Omega}\left(n^{5/4}\right)$ [Holden-Lyons '18]

[Batu, Kannan, Khanna, McGregor '04]
[Holenstein, Mitzenmacher, Panigrahy, Wieder '08]

Open TR Questions

Q1: Recover approximately using 10 traces

- 95% of cluster centers?
- some errors okay
- nontrivial 1,2,3, … traces

Goal: Recover X w.h.p. using min # traces

Q2: Improve these bounds!!

- current algorithms: single bits
- known exponential barrier
- need new ideas . . .

Known: $T_n \leq \exp\left(n^{1/3}\right)$ [Nazarov-Peres '16; De, O'Donnell, Servedio '16]

$T_n \geq \widetilde{\Omega}\left(n^{5/4}\right)$ [Holden-Lyons '18]

Future Directions: Molecular Informatics

1. How and what can we encode in molecules?

2. What types of operations can molecules execute?

3. What are the representational abstractions, mathematical or computational **primitives** that can describe these operations?

4. What does ‘computation’ mean in a molecular context?

5. What functions can be decided via molecular means and what equivalence might they have to traditional computing methods?

6. Can we design approaches to **compute directly on and with molecular data?**

Thanks!

Contact

Cyrus Rashtchian

www.cyrusrashtchian.com

UC San Diego

MISL

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING