Data Structures and Algorithms
Cardinality and MinHash Sketch

CS 225
G Carl Evans

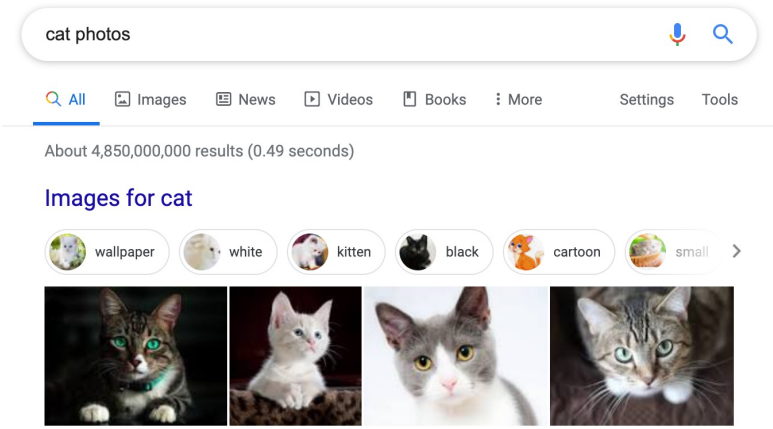
May 2, 2025

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN
Department of Computer Science

From slides by Brad Solomon

Cardinality

Sometimes its not possible or realistic to count all unique objects!



Estimate: 60 billion — 130 trillion

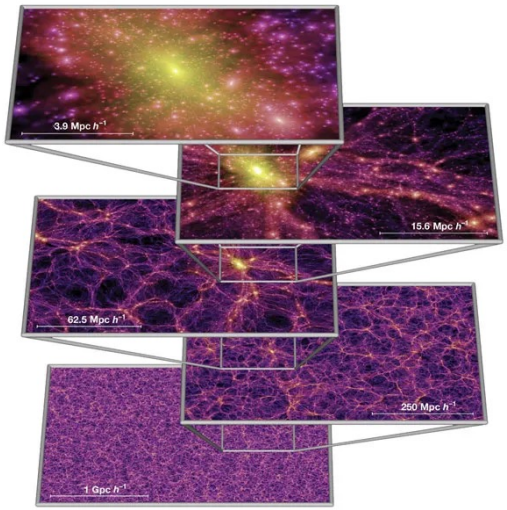


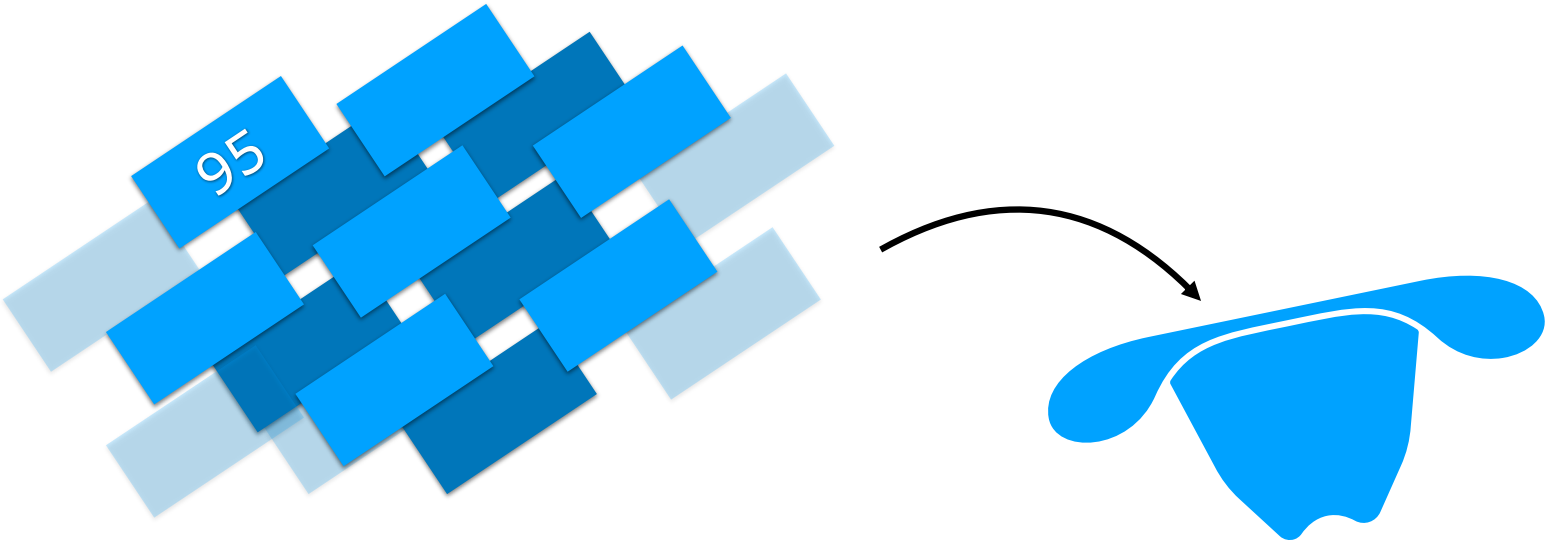
Image: https://doi.org/10.1038/nature03597

946
5581
8945
6145
8126
3887
8925
1246
8324
4549
9100
5598
8499
8970
3921
8575
4859
4960
42
6901
4336

Cardinality Estimation

Imagine I fill a hat with **a random subset** of numbered cards **from 0 to 999**

If I told you that the **minimum** value was 95, what have we learned?



Cardinality Estimation

Let min = 95. Can we estimate N, the cardinality of the set?

Conceptually: If we scatter N points randomly across the interval, we end up with $N + 1$ partitions, each about $1000/(N+1)$ long

Assuming our first 'partition' is about average:

$$95 \approx 1000/(N+1)$$
$$N+1 \approx 10.5$$
$$N \approx 9.5$$

Cardinality Estimation

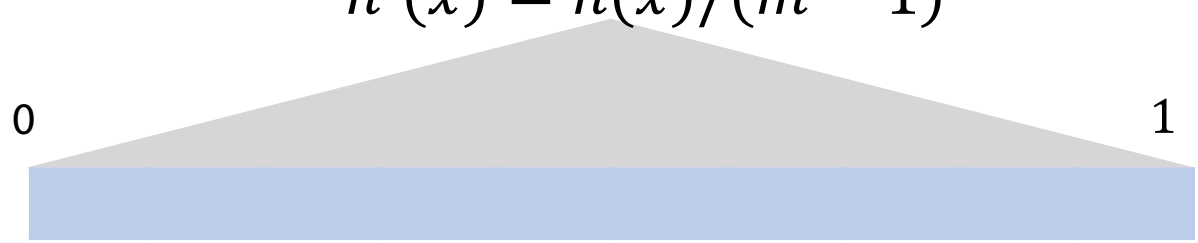
Imagine we have a SUHA hash h over a range m.

Inserting a new key is equivalent to adding a card to our hat!

Tracking only the minimum value is a **sketch** that estimates the cardinality!

To make the math work out, lets normalize our hash…

$$h'(x) = h(x)/(m-1)$$



Cardinality Sketch

Consider an $N+1$ draw: X_1 X_2 X_3 ... X_N X_{N+1}

$$M = \min_{1 \le i \le N} X_i$$

X_{N+1} **will be the new minimum with probability** M

By definition of SUHA, X_{N+1} has a $\frac{1}{N+1}$ chance of being smallest item

Thus, $\mathbf{E}[M] = \frac{1}{N+1}$

Cardinality Sketch

Claim: $\mathbf{E}[M] = \frac{1}{N+1}$

$$N \approx \frac{1}{M} - 1$$

Attempt 1	0.962	0.328	0.771	0.952	0.923
Attempt 2	0.253	0.839	0.327	0.655	0.491
Attempt 3	0.134	0.580	0.364	0.743	0.931

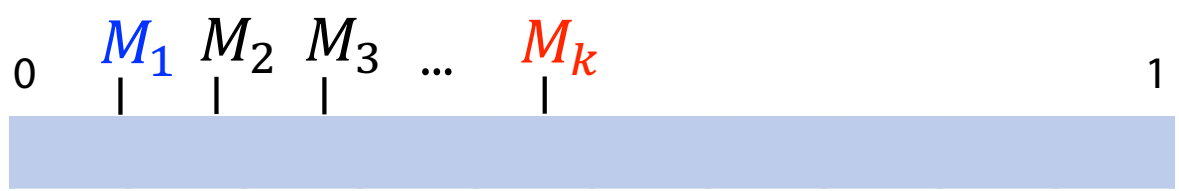
Cardinality Sketch

The minimum hash is a valid sketch of a dataset but can we do better?

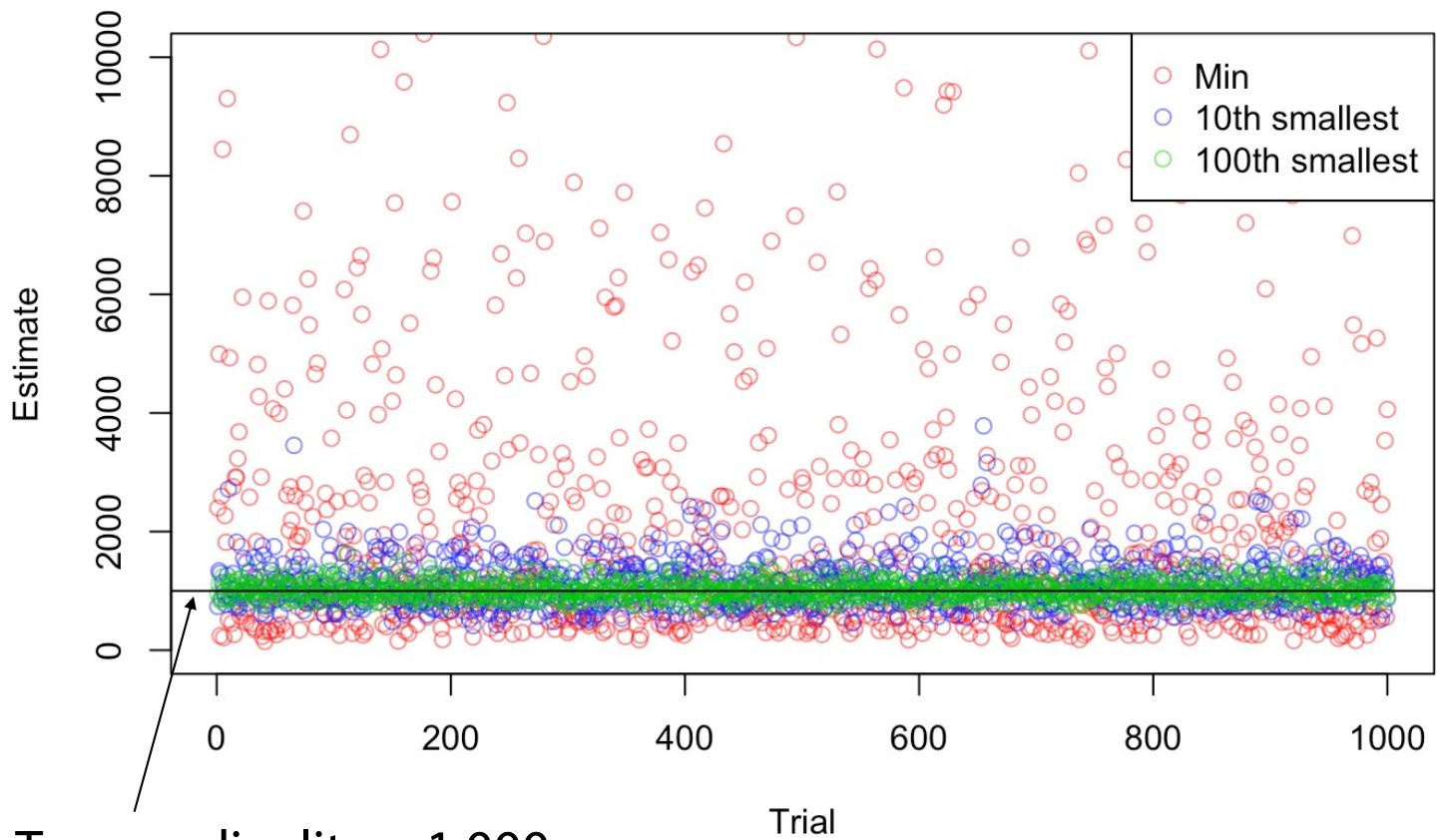
Cardinality Sketch

Claim: Taking the kth-smallest hash value is a better sketch!

Claim: $\mathbf{E}[M_k] = \frac{k}{N+1}$



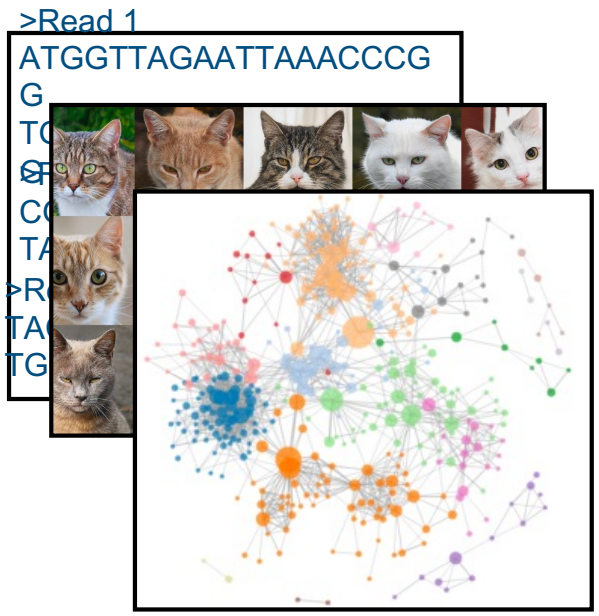
Cardinality Sketch



True cardinality = 1,000

Cardinality Sketch

Given any dataset and a SUHA hash function, we can **estimate the number of unique items** by tracking the **k-th minimum hash value**.



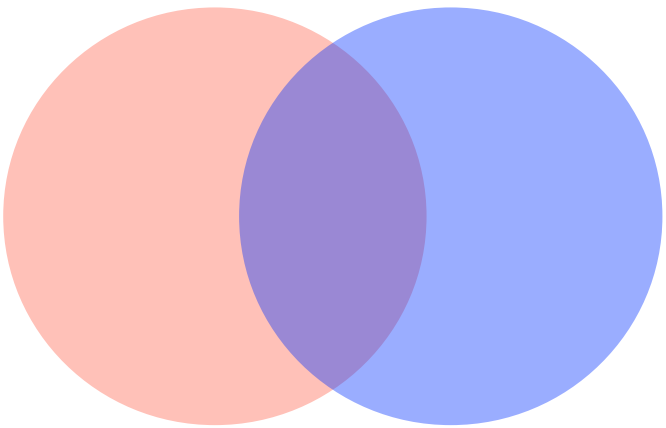
Hash

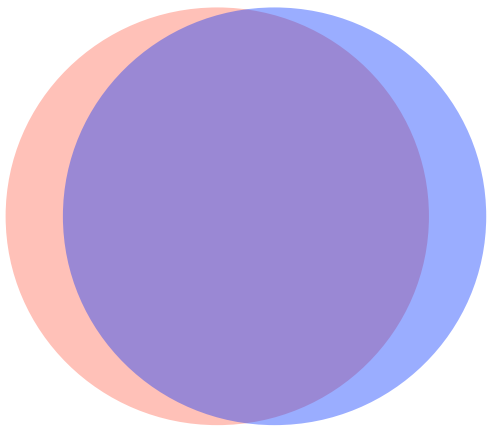
0.253	0.839	0.327	0.655	0.491

To use the k-th min, we have to track k minima. **Can we use ALL minima?**

Set Similarity

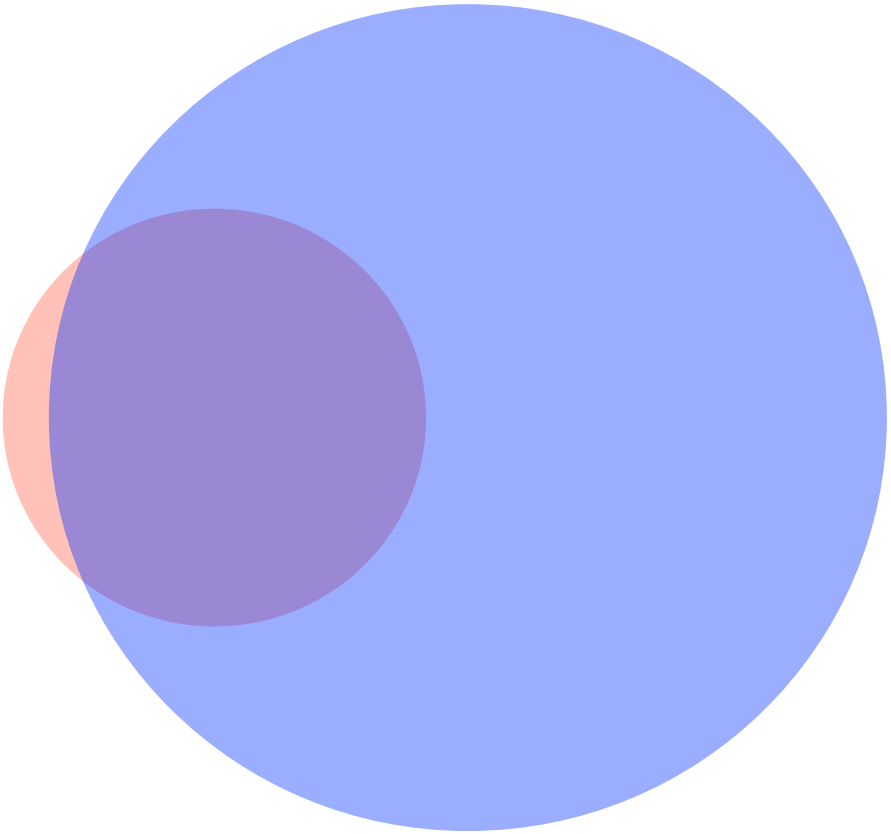
How can we describe how ***similar*** two sets are?

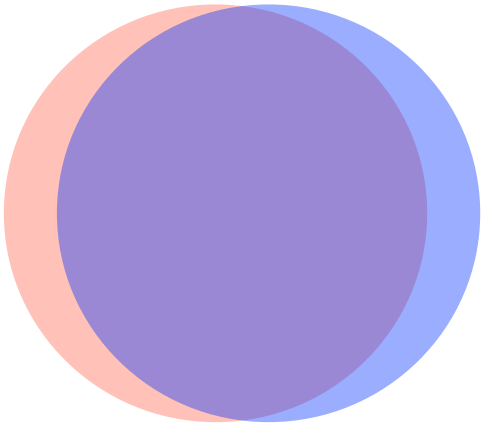




Set Similarity

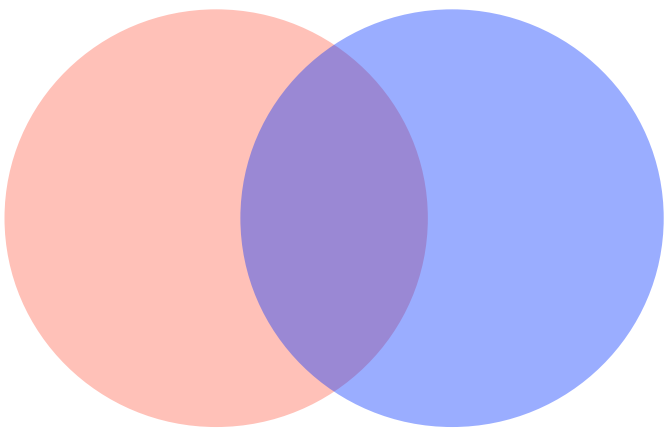
How can we describe how ***similar*** two sets are?





Set Similarity

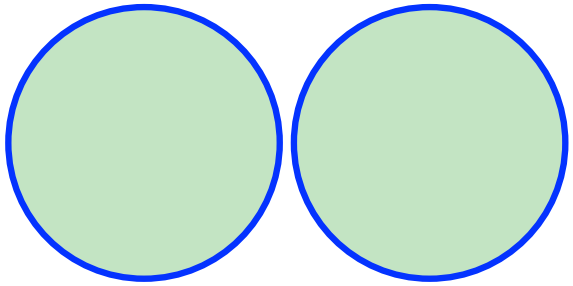
To measure **similarity** of A & B, we need both a measure of how similar the sets are but also the total size of both sets.



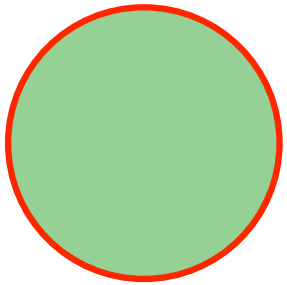
$$J = \frac{|A \cap B|}{|A \cup B|}$$

J is the ***Jaccard coefficient***

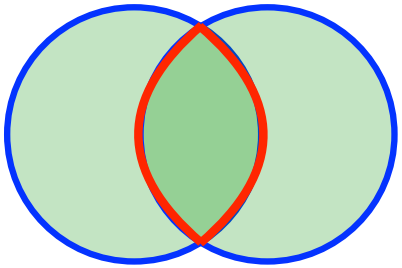
Set Similarity



$$\frac{|A \cap B|}{|A \cup B|} = 0$$



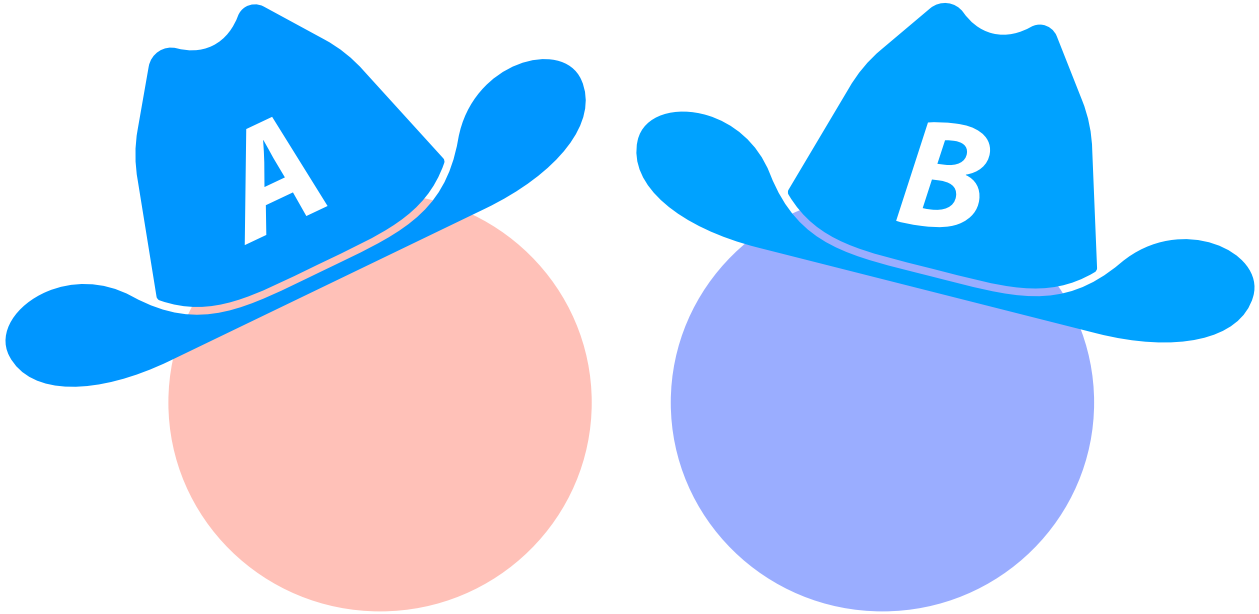
$$\frac{|A \cap B|}{|A \cup B|} = 1$$



$$0 < \frac{|A \cap B|}{|A \cup B|} < 1$$

Similarity Sketches

But what do we do when we only have a sketch?



Similarity Sketches

Imagine we have two datasets represented by their kth minimum values

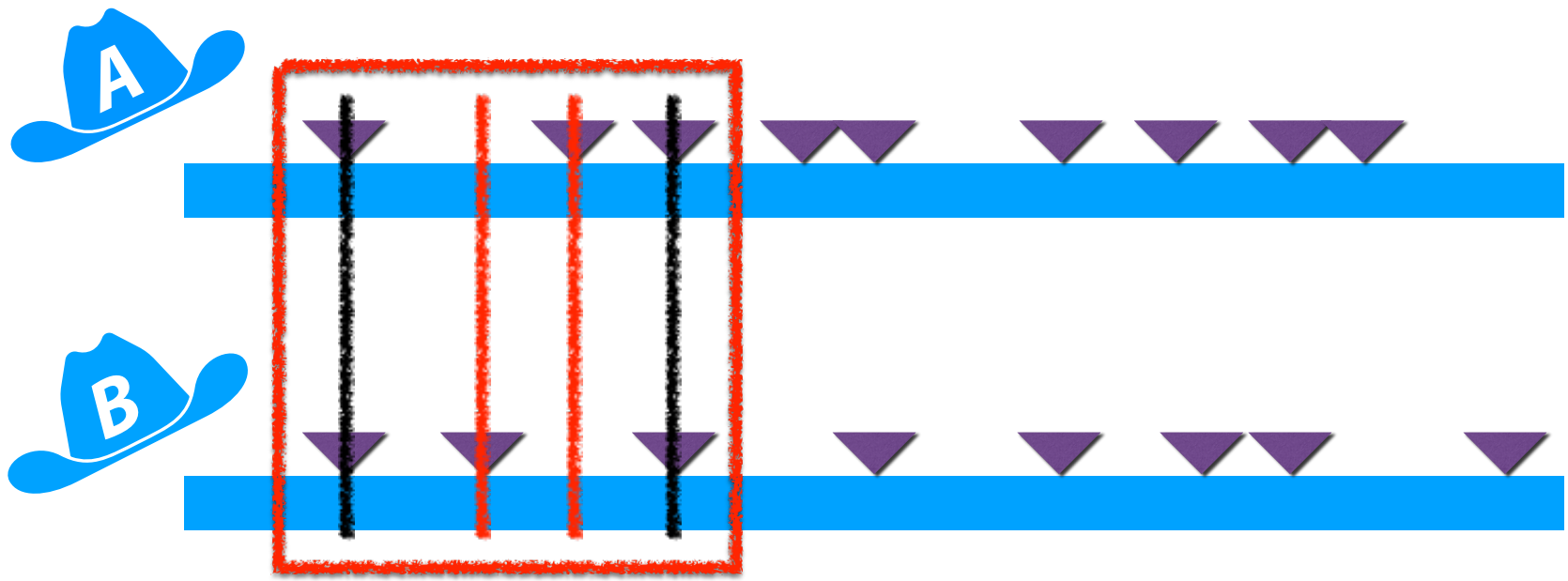


Image inspired by: Ondov B, Starrett G, Sappington A, Kostic A, Koren S, Buck CB, Phillippy AM. **Mash Screen: high-throughput sequence containment estimation for genome discovery**. Genome Biol 20, 232 (2019)

Similarity Sketches

Claim: Under SUHA, set similarity can be estimated by sketch similarity!

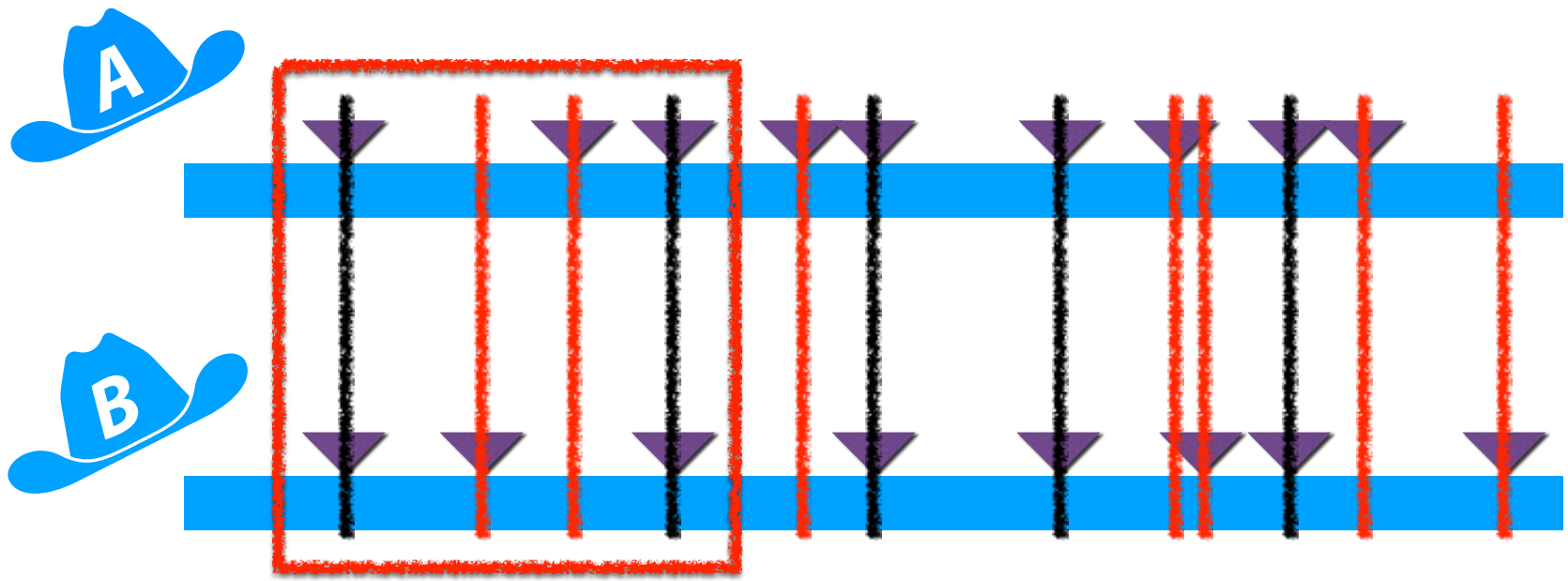
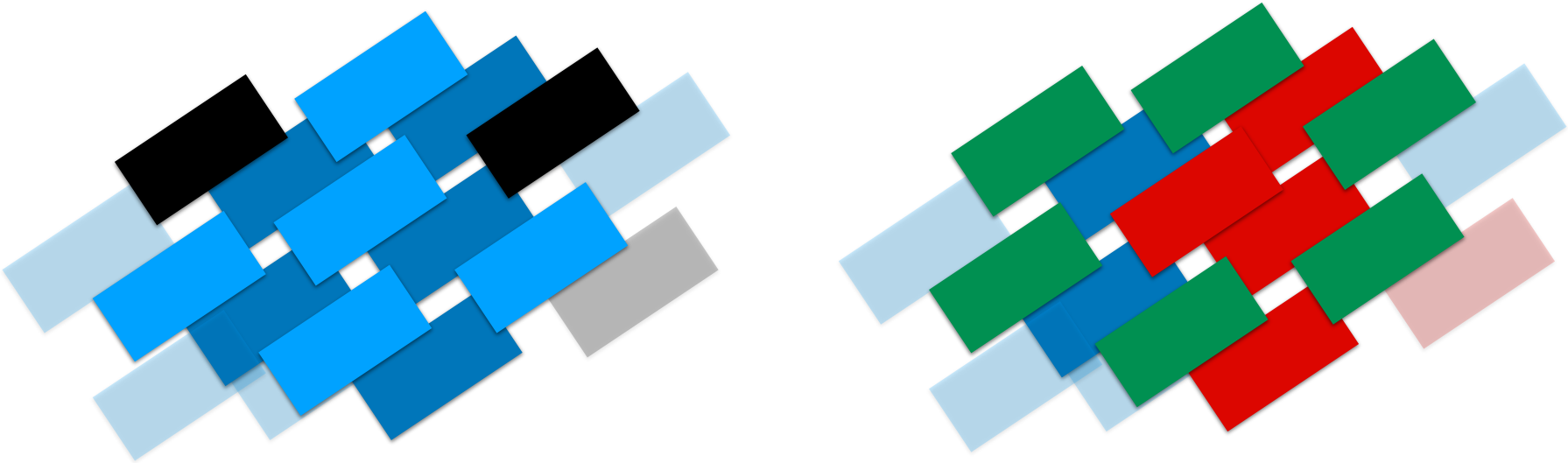


Image inspired by: Ondov B, Starrett G, Sappington A, Kostic A, Koren S, Buck CB, Phillippy AM. **Mash Screen: high-throughput sequence containment estimation for genome discovery**. Genome Biol 20, 232 (2019)

MinHash Sketch

The **k-th minimum value sketch** is built by tracking k minima but only uses one value (the k-th minima) to get **cardinality!**

We can extend this approach into a full **MinHash sketch** that can also estimate **set similarities.**



MinHash Construction

S = { 16, 8, 4, 13, 15}

h(x) = x % 7

k = 3

Algorithm is trivial:

1. Hash each item
2. Keep the k-minimum values in memory
   (Ignore collisions / duplicates)

0	
1	
2	

MinHash Jaccard Estimation

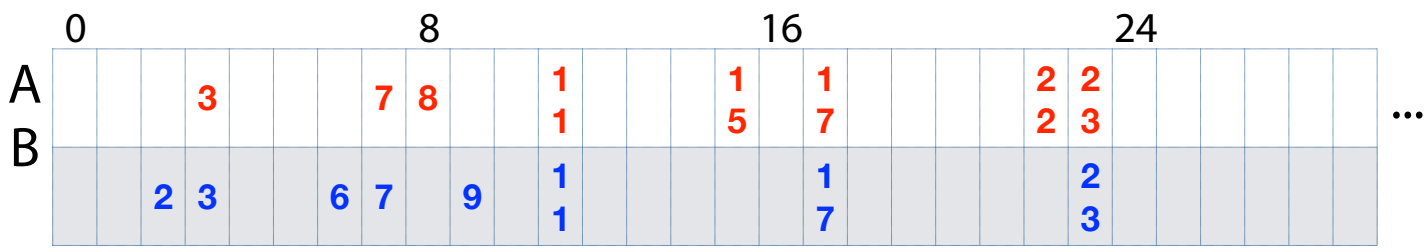
Given sets A and B sampled uniformly from [0, 100], store the bottom-8 **MinHash:**

Sketch A

3	15
7	17
8	22
11	23

Sketch B

2	9
3	11
6	17
7	23



MinHash Jaccard Estimation

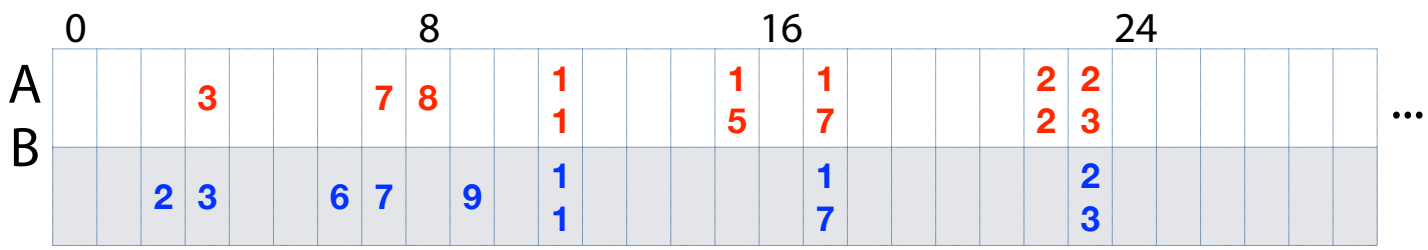
We want to estimate the Jaccard Coefficient: $\frac{|A \cap B|}{|A \cup B|}$

Sketch A

3	15
7	17
8	22
11	23

Sketch B

2	9
3	11
6	17
7	23



MinHash Jaccard Estimation

What do we know about $A \cup B$?

Sketch A

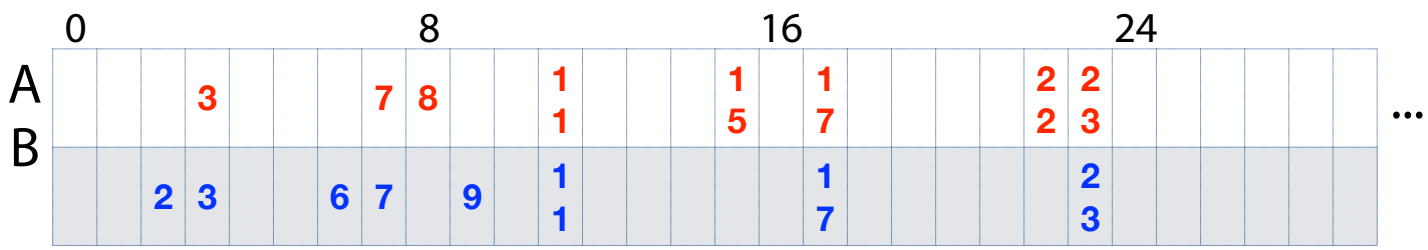
3	15
7	17
8	22
11	23

Sketch B

2	9
3	11
6	17
7	23

$A \cup B$

...



MinHash Jaccard Estimation

We dont *know* $A \cup B$, but we can make a sketch!

Sketch A

3	15
7	17
8	22
11	23

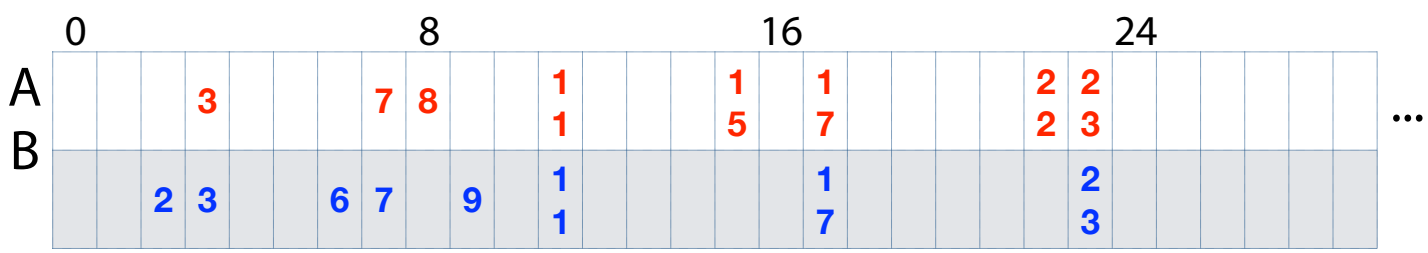
$\cup$

Sketch B

2	9
3	11
6	17
7	23

$=$

Sketch $A \cup B$



MinHash Jaccard Estimation

Estimate $|A \cup B|$ (the cardinality of the union) from sketch:

Sketch $A \cup B$

2	8
3	9
6	11
7	15

Our sets sampled from [0, 100].

MinHash Jaccard Estimation

Can we build a 8-Minhash of $A \cap B$?

Sketch A

3	15
7	17
8	22
11	23

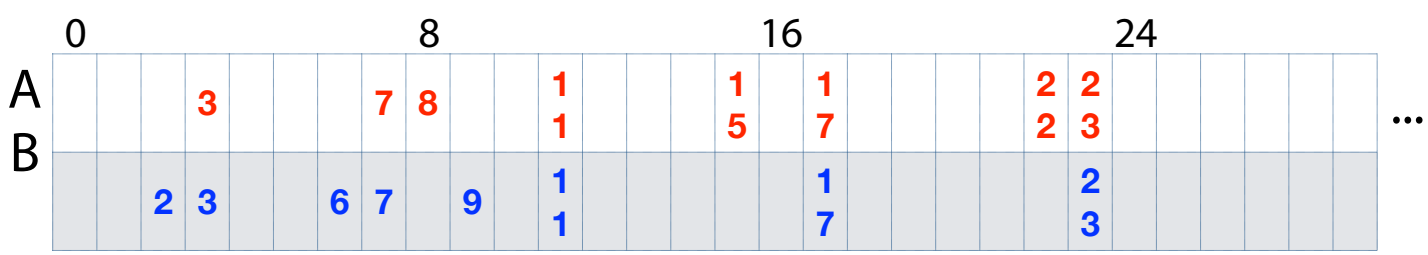
$\cap$

Sketch B

2	9
3	11
6	17
7	23

$=$

Sketch $A \cap B$



MinHash Jaccard Estimation

Unlikely to be able to get a full sketch of the intersection!

Sketch A

3	15
7	17
8	22
11	23

$\cap$

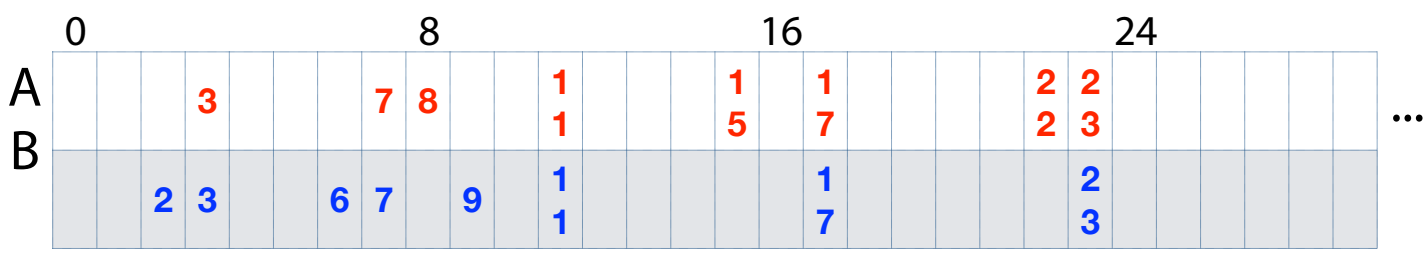
Sketch B

2	9
3	11
6	17
7	23

$=$

Sketch $A \cap B$

3	23
7	
11	
17	



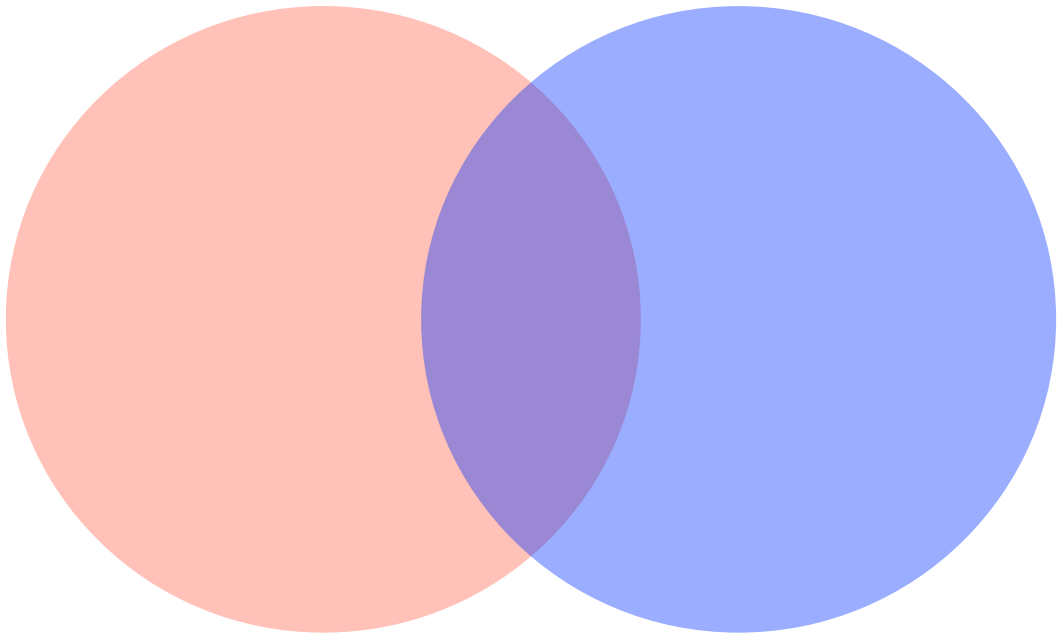
MinHash Jaccard Estimation

Using MinHash sketches, we can estimate $|A|$, $|B|$, and $|A \cup B|$

Is this enough to estimate the Jaccard?

Inclusion-Exclusion Principle

$$|A \cap B| =$$



MinHash Indirect Jaccard Estimation

$$\frac{|A| \cap |B|}{|A| \cup |B|} = \frac{|A| + |B| - |A \cup B|}{|A \cup B|}$$

$k = 8$ MinHash sketches

Our sets sampled from [0, 100]

Sketch A

3	15
7	17
8	22
11	23

Sketch B

2	9
3	11
6	17
7	23

Sketch of $|A \cup B|$

2	8
3	9
6	11
7	15

$$= \frac{(800/23 - 1) + (800/23 - 1) - (800/15 - 1)}{800/15 - 1}$$

$$= \frac{34.782 + 34.782 - 53.333 - 1}{53.333 - 1} \approx 0.29$$

MinHash Direct Jaccard Estimate

We can also estimate cardinality directly using our sketches!

Sketch A

3	15
7	17
8	22
11	23

Sketch B

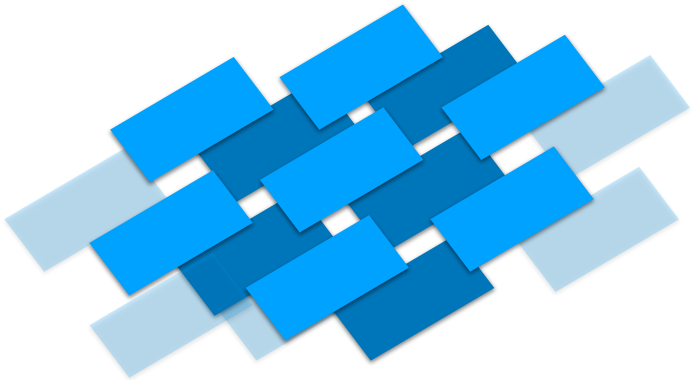
2	9
3	11
6	17
7	23

Intersection

Union

MinHash Sketch

We can convert any hashable dataset into a **MinHash sketch**



{1 , 3 , 5}

We lose our original dataset, but we can still estimate two things:

1.

2.

Alternative MinHash Sketch Approaches

Rather than use one single hashes and take bottom-k, we can also use k hashes — **if you have access to that many independent hashes**!

1) Sequence decomposed into **kmers**

```
S1 : CATGGACCGACCAG      GCAGTACCGATCGT : S2
     CAT GAC GAC            GTA CGA CGT
      ATG ACC ACC          AGT CCG TCG
       TGG CCG CCA        CAG ACC ATC
        GGA CGA CAG      GCA TAC GAT
```

Assembling large genomes with single-molecule sequencing and locality-sensitive hashing
Berlin et al (2015) *Nature Biotechnology*

1) Sequence decomposed into **kmers**

2) Multiple hash functions (**Γ**) map kmers to values.

S_1 : CATGGACCGACCAG

CAT GAC GAC
ATG ACC ACC
TGG CCG CCA
GGA CGA CAG

Γ_1	Γ_2	Γ_3	Γ_4	
19	14	57	36	CAT
14	57	36	19	ATG
58	37	16	15	TGG
40	23	2	61	GGA
33	28	11	54	GAC
5	48	47	26	ACC
22	1	60	43	CCG
24	7	50	45	CGA
33	28	11	54	GAC
5	48	47	26	ACC
20	3	62	41	CCA
18	13	56	39	CAG

GCAGTACCGATCGT : S_2

GTA CGA CGT
AGT CCG TCG
CAG ACC ATC
GCA TAC GAT

	Γ_1	Γ_2	Γ_3	Γ_4
GCA	36	19	14	57
CAG	18	13	56	39
AGT	11	54	33	28
GTA	44	27	6	49
TAC	49	44	27	6
ACC	5	48	47	26
CCG	22	1	60	43
CGA	24	7	50	45
GAT	35	30	9	52
ATC	13	56	39	18
TCG	54	33	28	11
CGT	27	6	49	44

Assembling large genomes with single-molecule sequencing and locality-sensitive hashing
Berlin et al (2015) *Nature Biotechnology*

1) Sequence decomposed into **kmers**

2) Multiple hash functions (**Γ**) map kmers to values.

3) The smallest values for each hash function is chosen

S_1 : CATGGACCGACCAG

CAT GAC GAC
ATG ACC ACC
TGG CCG CCA
GGA CGA CAG

Γ_1	Γ_2	Γ_3	Γ_4	
19	14	57	36	CAT
14	57	36	19	ATG
58	37	16	15	TGG
40	23	2	61	GGA
33	28	11	54	GAC
5	48	47	26	ACC
22	1	60	43	CCG
24	7	50	45	CGA
33	28	11	54	GAC
5	48	47	26	ACC
20	3	62	41	CCA
18	13	56	39	CAG

[5, 1, 2, 15]
Sketch (S_1)

GCAGTACCGATCGT : S_2

GTA CGA CGT
AGT CCG TCG
CAG ACC ATC
GCA TAC GAT

	Γ_1	Γ_2	Γ_3	Γ_4
GCA	36	19	14	57
CAG	18	13	56	39
AGT	11	54	33	28
GTA	44	27	6	49
TAC	49	44	27	6
ACC	5	48	47	26
CCG	22	1	60	43
CGA	24	7	50	45
GAT	35	30	9	52
ATC	13	56	39	18
TCG	54	33	28	11
CGT	27	6	49	44

[5, 1, 6, 6]
Sketch (S_2)

Assembling large genomes with single-molecule sequencing and locality-sensitive hashing
Berlin et al (2015) *Nature Biotechnology*

1) Sequence decomposed into **kmers**

2) Multiple hash functions (**Γ**) map kmers to values.

3) The smallest values for each hash function is chosen

4) The Jaccard similarity can be estimated by the overlap in the **Min**imum **Hash**es (**MinHash)**

```
S1 : CATGGACCGACCAG
     CAT GAC GAC
      ATG ACC ACC
       TGG CCG CCA
        GGA CGA CAG
```

Γ_1	Γ_2	Γ_3	Γ_4	
19	14	57	36	CAT
14	57	36	19	ATG
58	37	16	15	TGG
40	23	2	61	GGA
33	28	11	54	GAC
5	48	47	26	ACC
22	1	60	43	CCG
24	7	50	45	CGA
33	28	11	54	GAC
5	48	47	26	ACC
20	3	62	41	CCA
18	13	56	39	CAG

[5, 1, 2, 15]
Sketch (S_1)

```
GCAGTACCGATCGT : S2
   GTA CGA CGT
  AGT CCG TCG
 CAG ACC ATC
GCA TAC GAT
```

	Γ_1	Γ_2	Γ_3	Γ_4
GCA	36	19	14	57
CAG	18	13	56	39
AGT	11	54	33	28
GTA	44	27	6	49
TAC	49	44	27	6
ACC	5	48	47	26
CCG	22	1	60	43
CGA	24	7	50	45
GAT	35	30	9	52
ATC	13	56	39	18
TCG	54	33	28	11
CGT	27	6	49	44

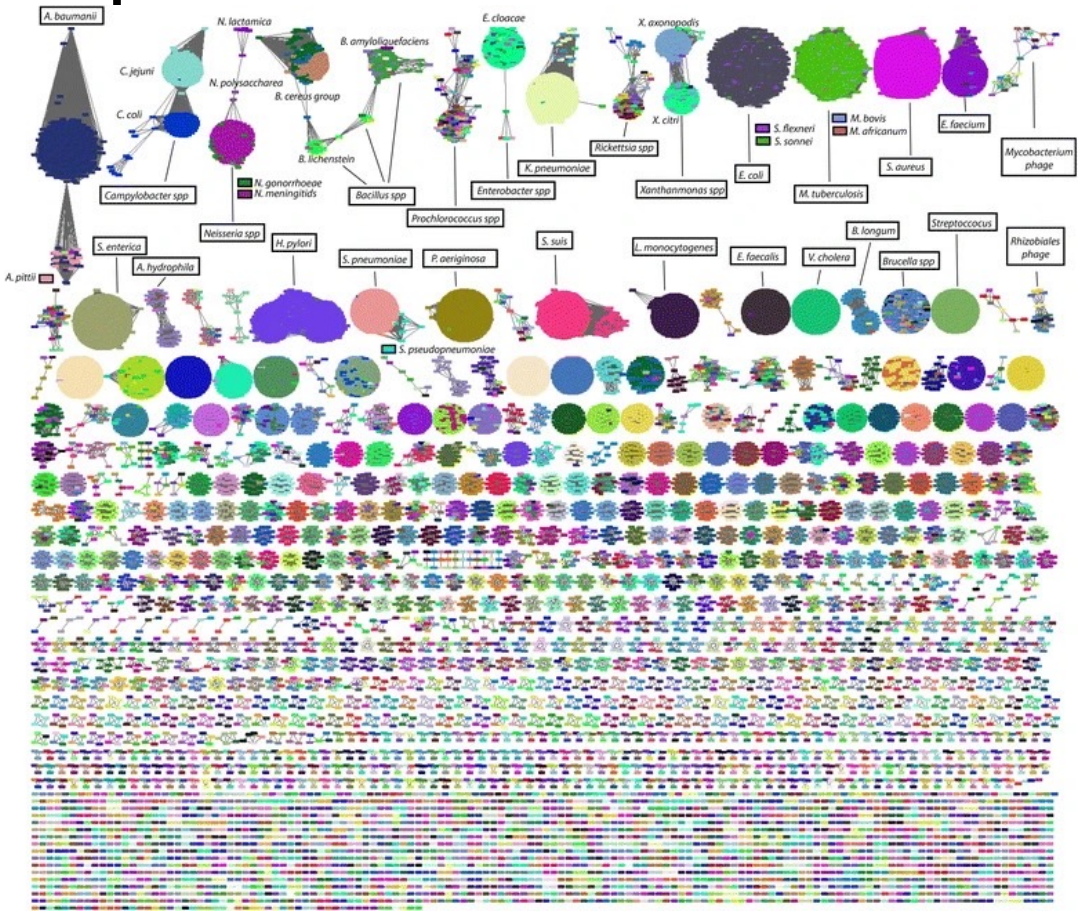
[5, 1, 6, 6]
Sketch (S_2)

$J(S_1, S_2) \approx 2/4 = 0.5$

```
S1 : CATGGACCGACCAG
          | |||||  |
S2 : GCAGTACCGATCGT
```

Assembling large genomes with single-molecule sequencing and locality-sensitive hashing
Berlin et al (2015) *Nature Biotechnology*

MinHash in practice



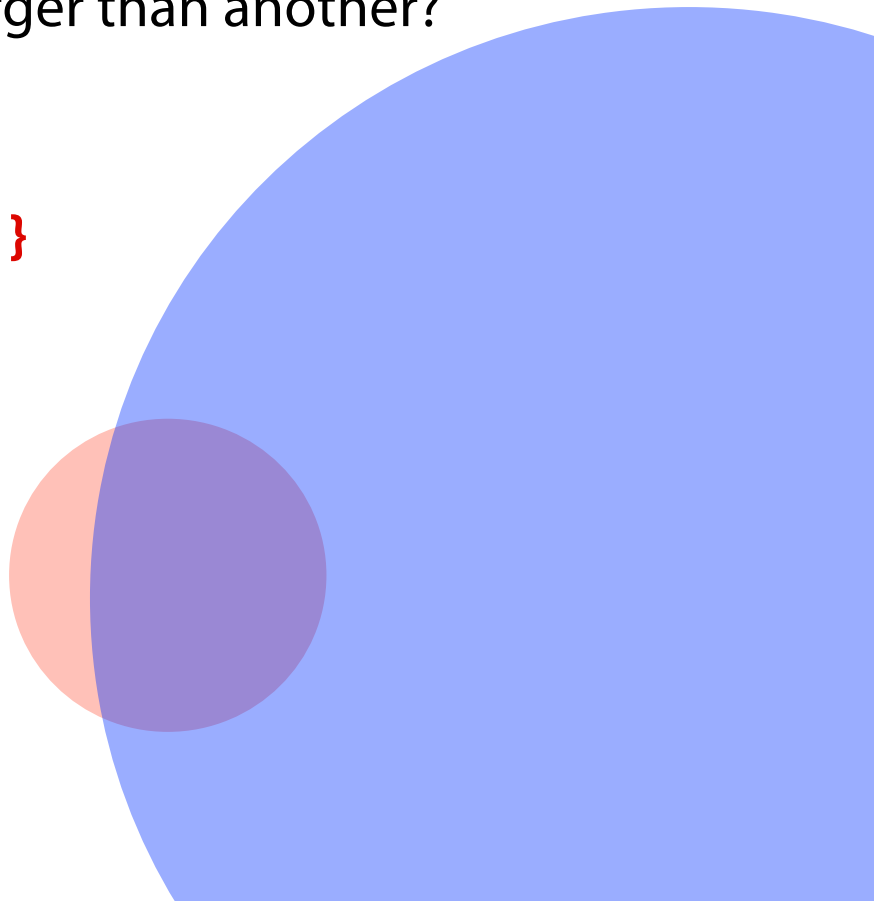
Mash: fast genome and metagenome distance estimation using MinHash
Ondov et al (2016) *Genome Biology*

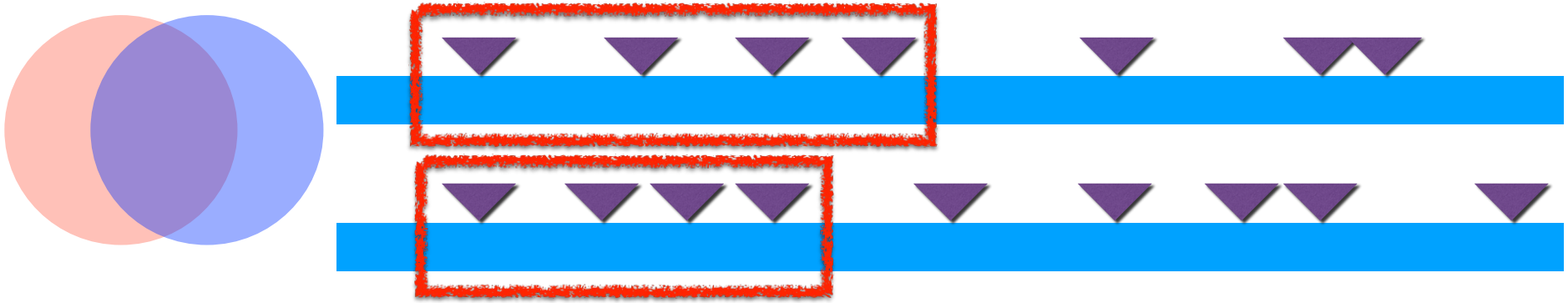
Alternative MinHash Sketch Approaches

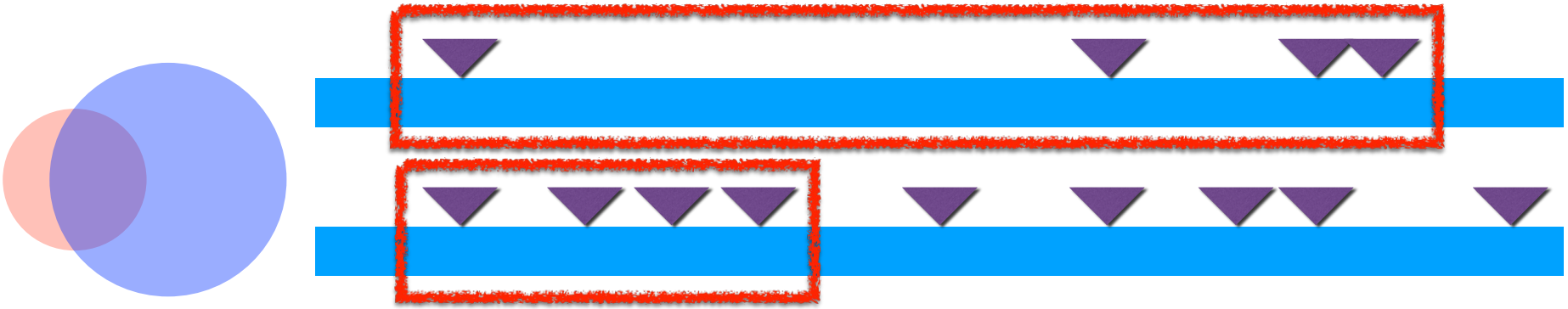
What if I have a dataset which is **much** larger than another?

S_1 = { 1, 3, 40, 59, 82, 101 }

S_2 = { 1, 2, 3, 4, 5, 6, 7, … 59, 82, 101, … }

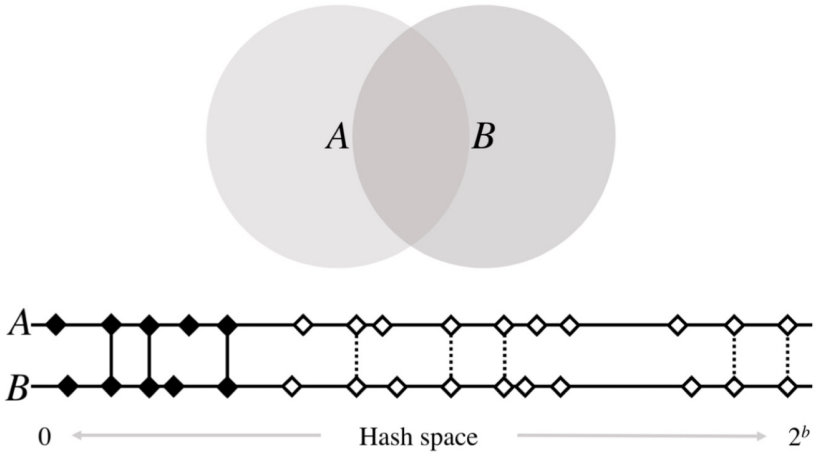


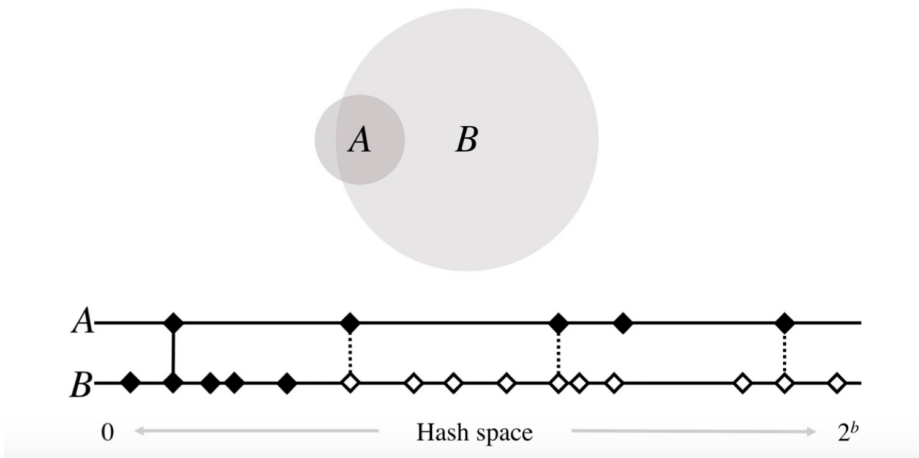




Alternative MinHash sketches

Bottom-k minhash has low accuracy if the cardinality of sets are skewed

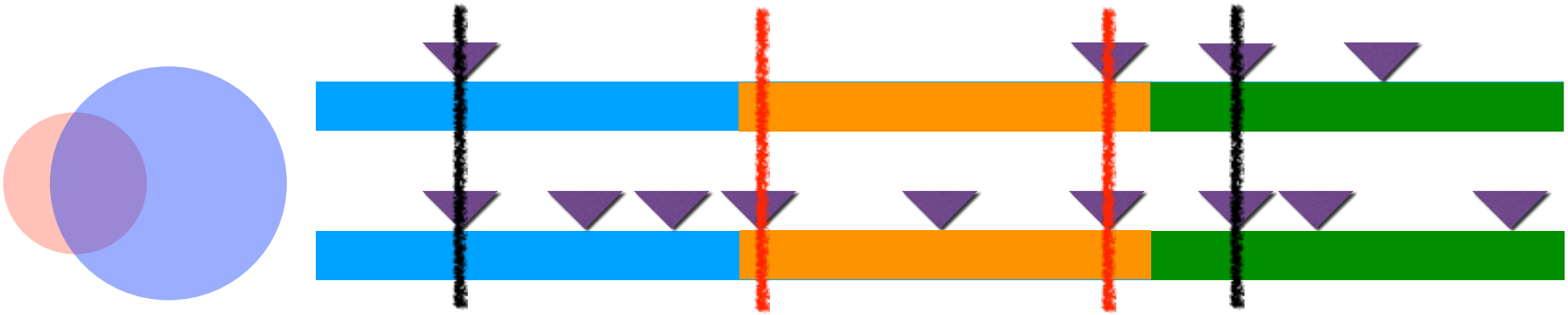




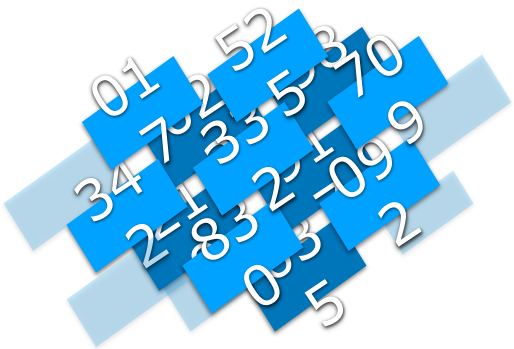
Ondov, Brian D., Gabriel J. Starrett, Anna Sappington, Aleksandra Kostic, Sergey Koren, Christopher B. Buck, and Adam M. Phillippy. **Mash Screen: High-throughput sequence containment estimation for genome discovery**. Genome biology 20.1 (2019): 1-13.

Alternative MinHash Sketch Approaches

If there is a large cardinality difference, **use k-partitions!**



K-Partition Minhash



Hash

1010110101

0001111010

1101101011

1011010110

0101100000

0010001101

Partition

00

01111010

10001101

01

01100000

10

10110101

11010110

11

01101011

Probabilistic Data Structures

Probabilistic data structures trade accuracy for efficiency

Most can maintain surprisingly good accuracy

“Cheat” Big O limitations on conventional data analysis