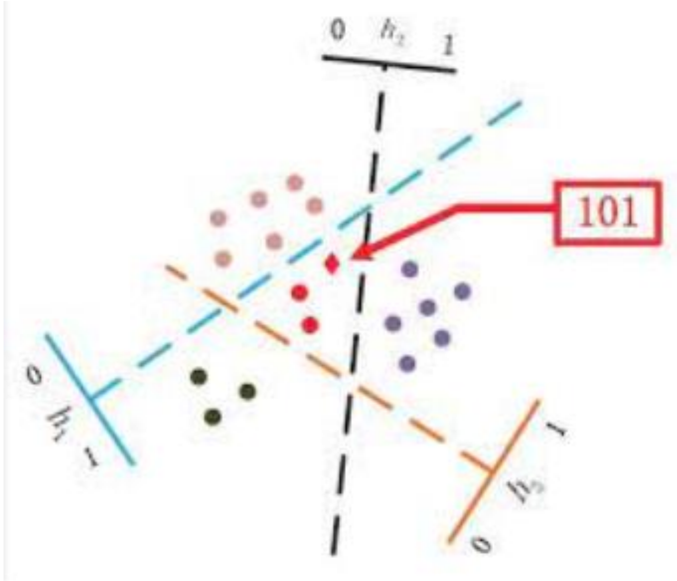
Genome Project, Computer Scientific Approach

Mohammad Saleh Bahrami

2023

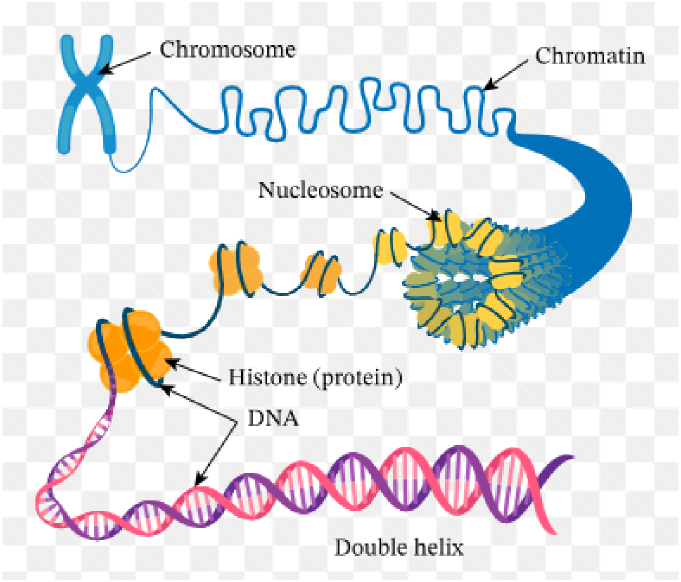
In other areas

What is Locality sensitive Hashing:

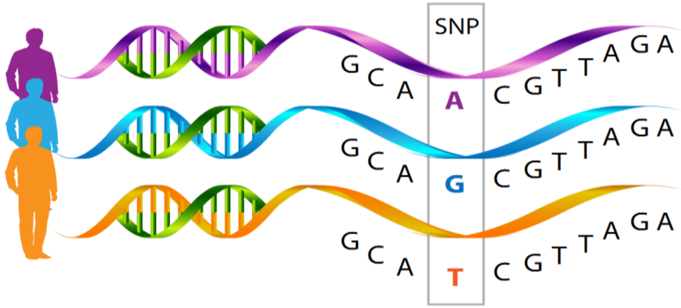


- Find Nearest Neighbor Query in High Dimension:
  - Approximately But Fast(necessity for Large Data).
  - In low dim there are many choices like : kd tree , voronoi diagram etc.
- Given Two text T_1 , T_2 with $dist(T_1, T_2) \leq \varepsilon$
- Text as a high Dim. point
- Here we prefer to design such a Hash Function like H such that:
  - $H(T_1) \approx H(T_2)$ in some sense, then hash collision find our solution (unlike before)
- Has application in wide range of areas such as:
  - Web mining, compressed sensing , etc .

Genome as Long Text



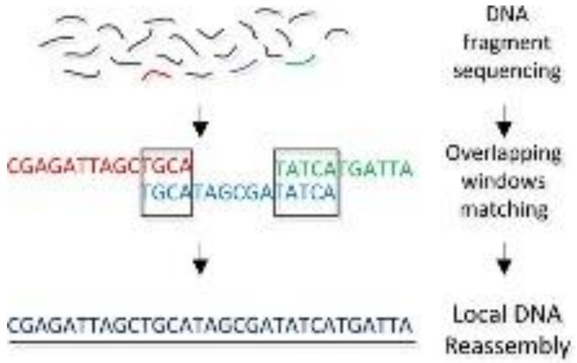
- Length = O(10^9)
- Operations :
  - search ,
  - compare two text (normal & anomaly),
  - Find repetitive patterns and interconection betweens patterns
  - in some sense learn language Models ,
  - But! even read the text is challenge! (and rewrite also)
- need Text Mining, Advanced Algorithmic Technique
- Must be in O(n) , exact as much as possible



How to read text? (sequencing)

- pieces of puzzles are given:
  - reads {short : 300 , long : 10 ^5 but noisy }
  - reconstruct whole shape
  - Overlapping and Coverage also exists

Assembly



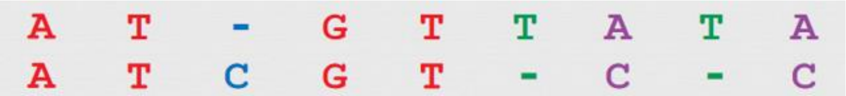
- Genome assembly is the process of reconstructing a genome from a collection of short sequencing reads.
- Called Genome project
- May be we have Reference genome , may be we don’t have …
- de novo assembly is without references reconstruction.
- An accurate reconstruction is crucial
- ***repetitive sequences*** make assembly difficult when the repeat length exceeds the read length

Interspersed Repeat
repeated sequence

repetitive sequences

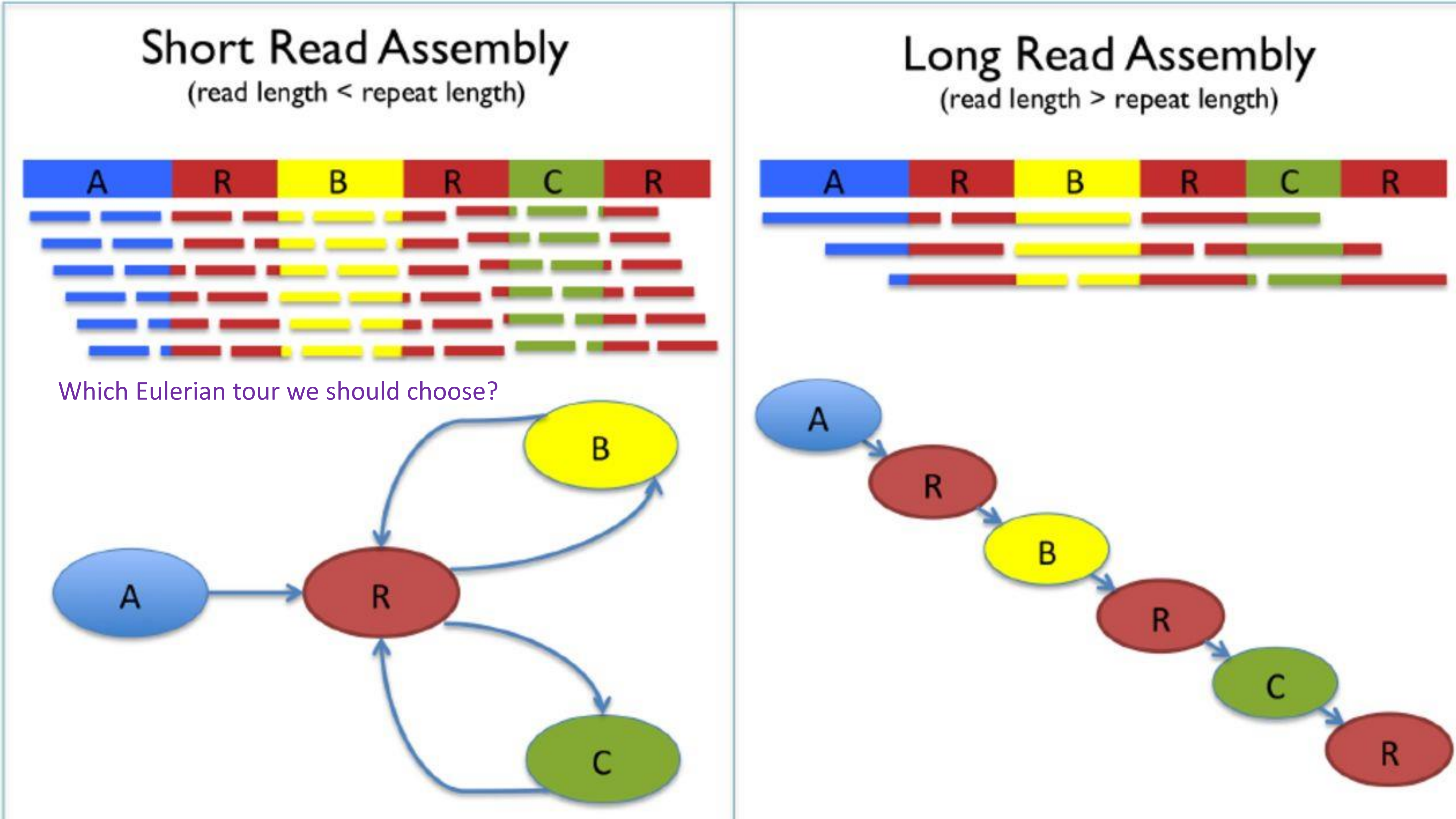
- unfortunately Its common and effect ***short reads***
- Recent advances : ***Long Reads***
  - ***pacBio SMRT sequencing:***
  - suffer from **low accuracy** (82–87% PacBio11, 78–85% MinION9)

exact matching doesn't work new algorithm developes such that : by **oversampling** the genome at sufficient coverage (e.g., 50× of PacBio P5C3), SMRT sequencing can be used to produce highly accurate and continuous assemblies



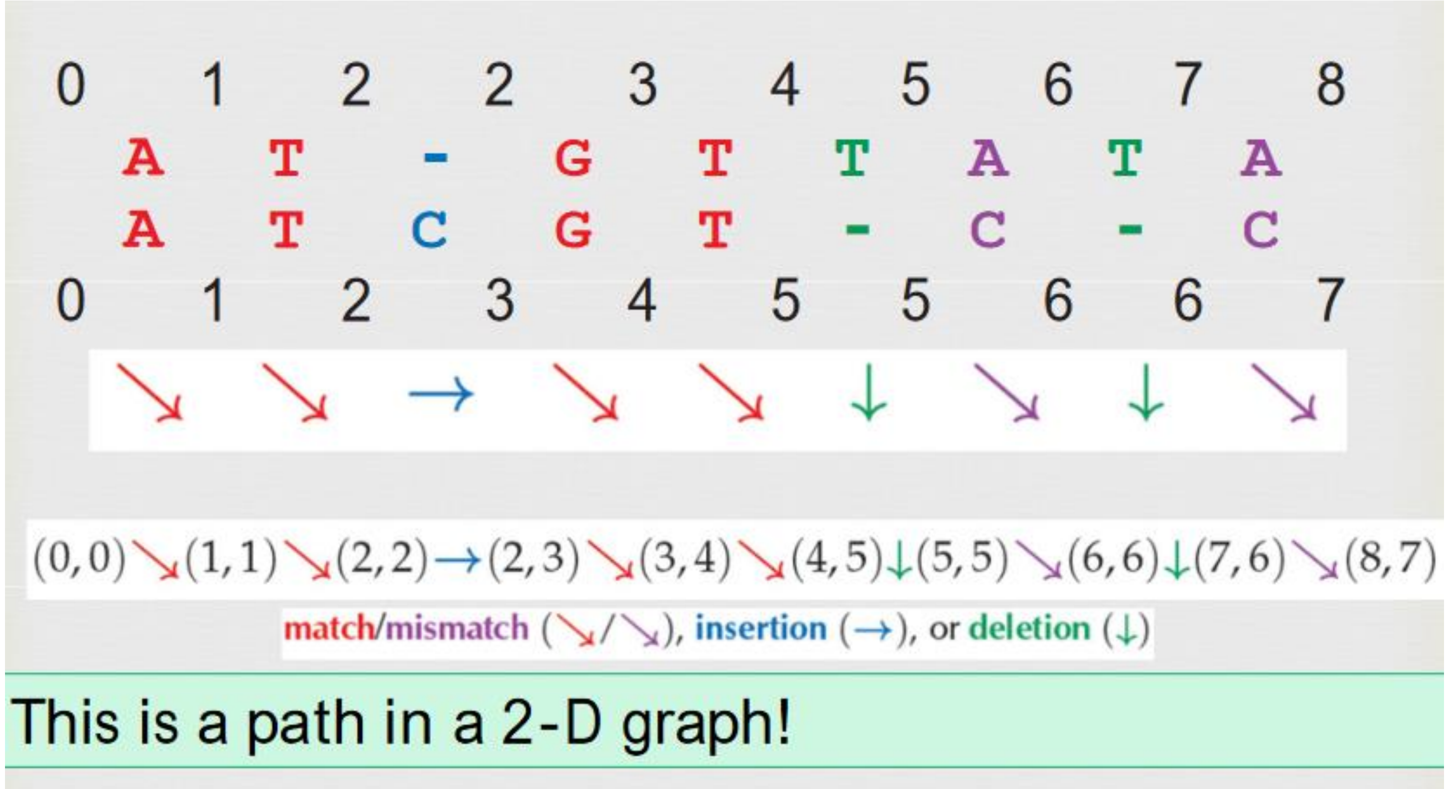
```
A  T  -  G  T  T  A  T  A
A  T  C  G  T  -  C  -  C
```

We deal with Noisy String Matching



Which Eulerian tour we should choose?

Edit distance:



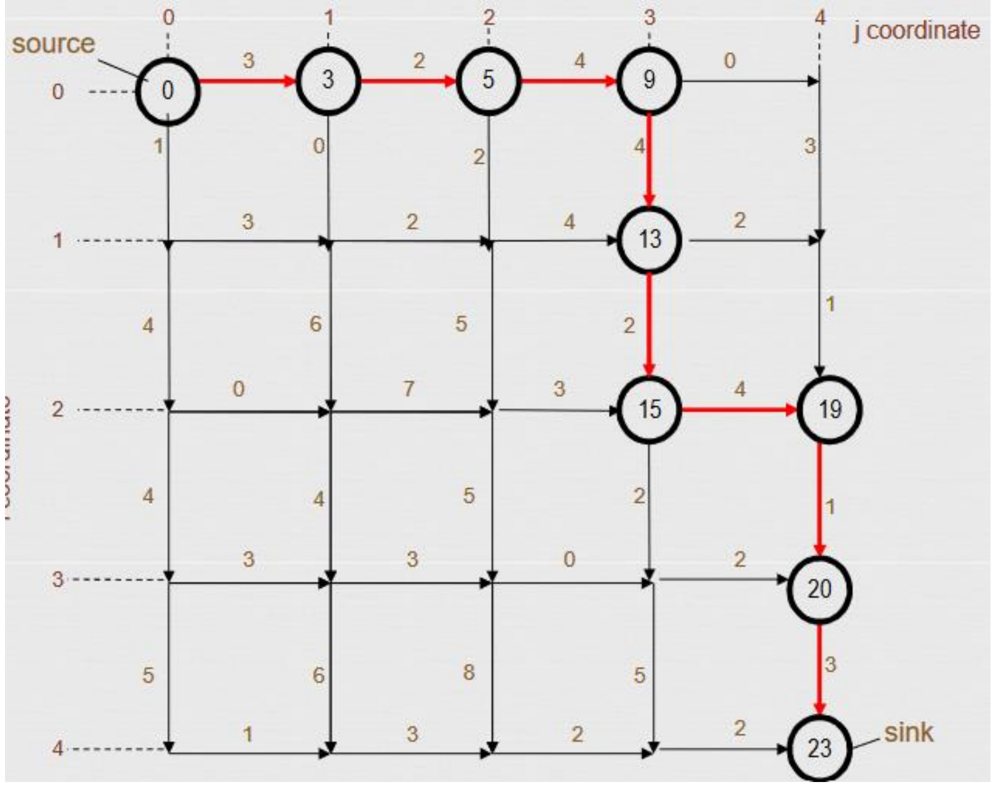
0		1		2		2		3		4		5		6		7		8
	A		T		-		G		T		T		A		T		A	
	A		T		C		G		T		-		C		-		C	
0		1		2		3		4		5		5		6		6		7
	↘		↘		→		↘		↘		↓		↘		↓		↘	

$(0,0)\searrow(1,1)\searrow(2,2)\rightarrow(2,3)\searrow(3,4)\searrow(4,5)\downarrow(5,5)\searrow(6,6)\downarrow(7,6)\searrow(8,7)$

match/mismatch (↘/↘), insertion (→), or deletion (↓)

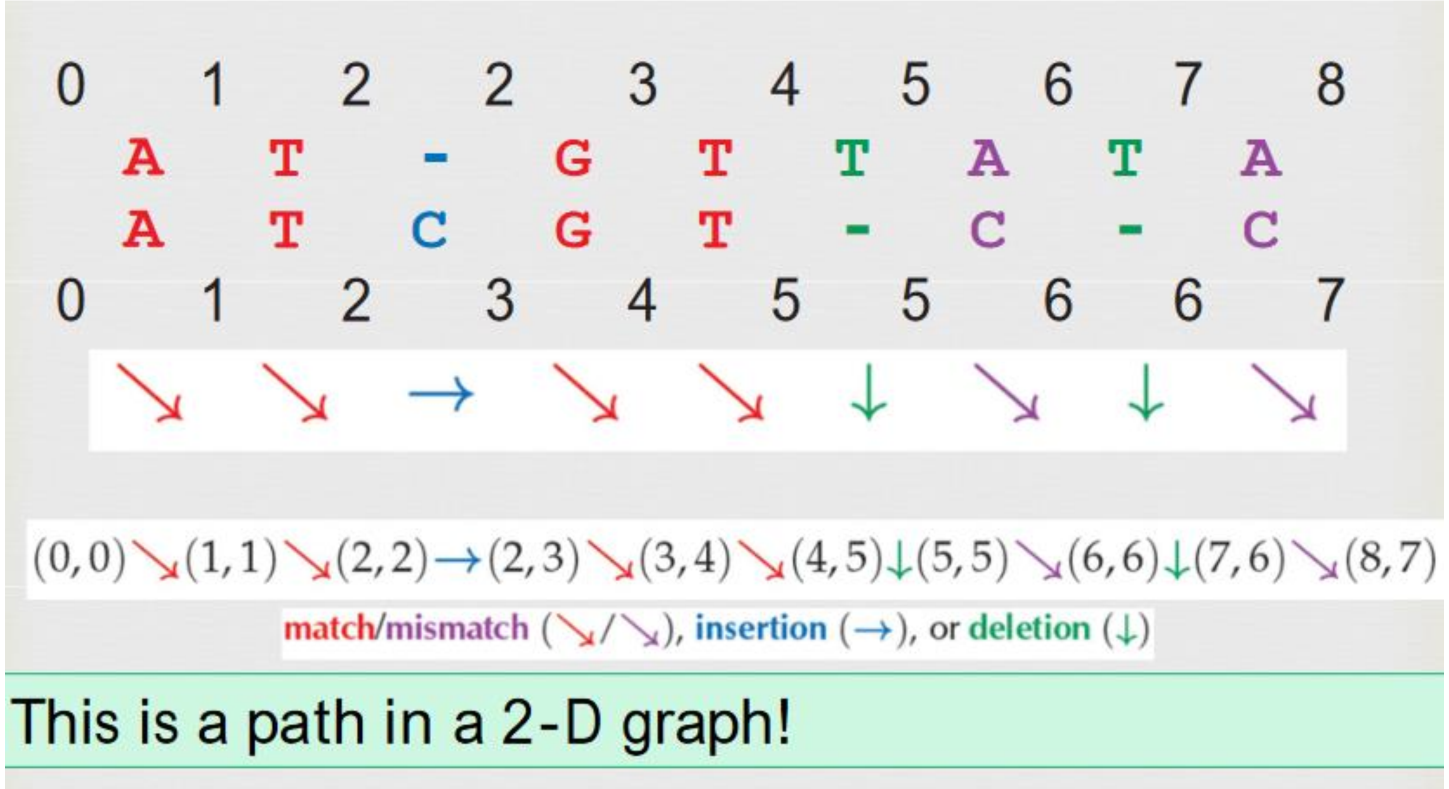
This is a path in a 2-D graph!

DP & Edit distance:



Not just Grids it work in every low dim DAG

DP & DAG & hamming distance



0 1 2 2 3 4 5 6 7 8

A T - G T T A T A

A T C G T - C - C

0 1 2 3 4 5 5 6 6 7

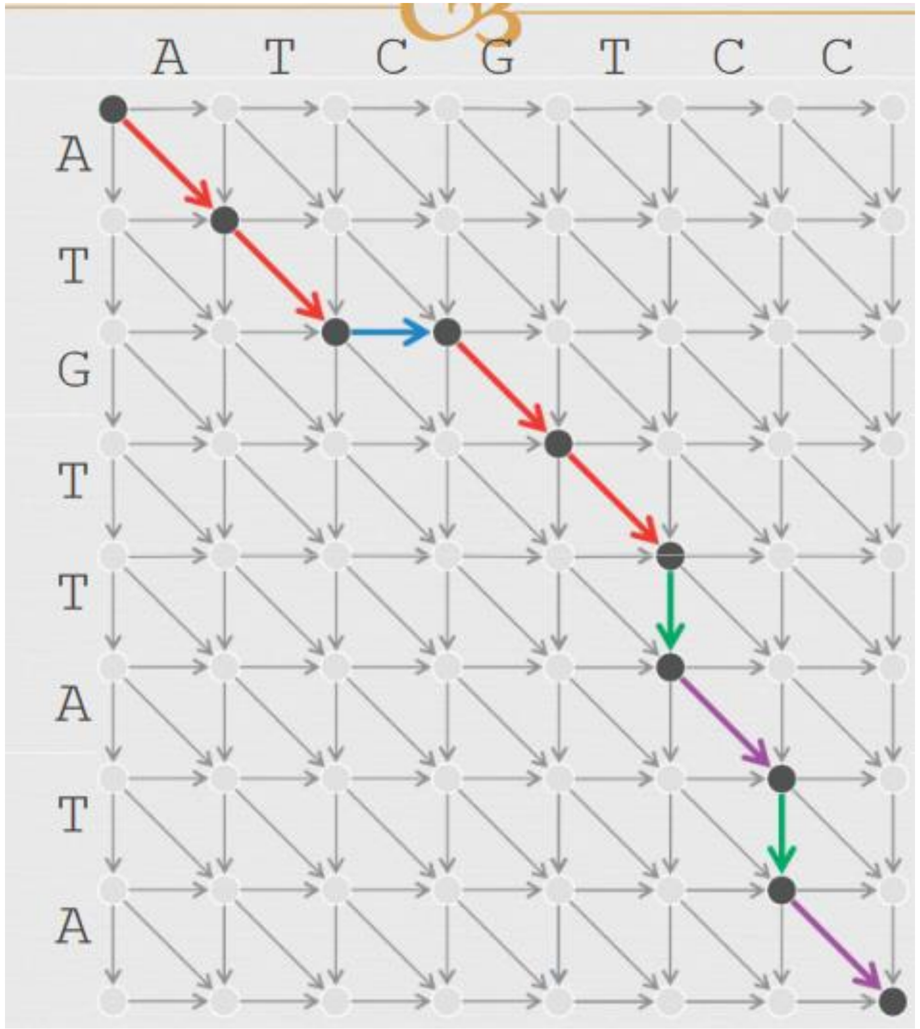
↘ ↘ → ↘ ↘ ↓ ↘ ↓ ↘

(0,0)↘(1,1)↘(2,2)→(2,3)↘(3,4)↘(4,5)↓(5,5)↘(6,6)↓(7,6)↘(8,7)

match/mismatch (↘/↘), insertion (→), or deletion (↓)

This is a path in a 2-D graph!

DP & DAG & hamming distance



DP & DAG & edit distance

- early assemblies of noisy, long reads have been successful, but have
- suffered from a substantial computational cost
- assembly of D. melanogaster from SMRT reads
- 600,000 CPU hours
- where is bottleneck ?
- 95% of the total runtime
- all-pairs overlapping will remain a substantial
- bottleneck in overlap-layout-consensus assembly

Locality sensitive Hashing:

- n pages which pairs are more similar:
  - Naïve : O(n^2)
  - O(n),Big-Foot of CS
- probabilistic algorithm for efficiently detecting overlaps

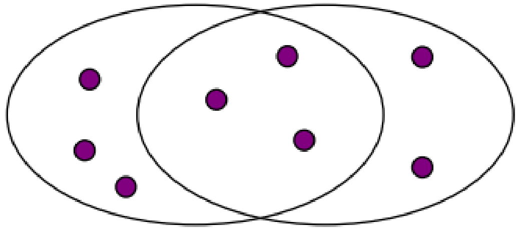
between noisy, long reads. MHAP uses a dimensionality reduction

technique named Min-Hash originally developed to determine the similarity of web pages

99.99% accurate when compared with available reference genomes.

Jaccard distance/similarity

- The **Jaccard similarity** of two **sets** is the size of their intersection divided by the size of their union: $sim(C_1, C_2) = |C_1 \cap C_2| / |C_1 \cup C_2|$
- **Jaccard distance:** $d(C_1, C_2) = 1 - |C_1 \cap C_2| / |C_1 \cup C_2|$



3 in intersection
8 in union
Jaccard similarity= 3/8
Jaccard distance = 5/8

3 Essential Steps for Similar

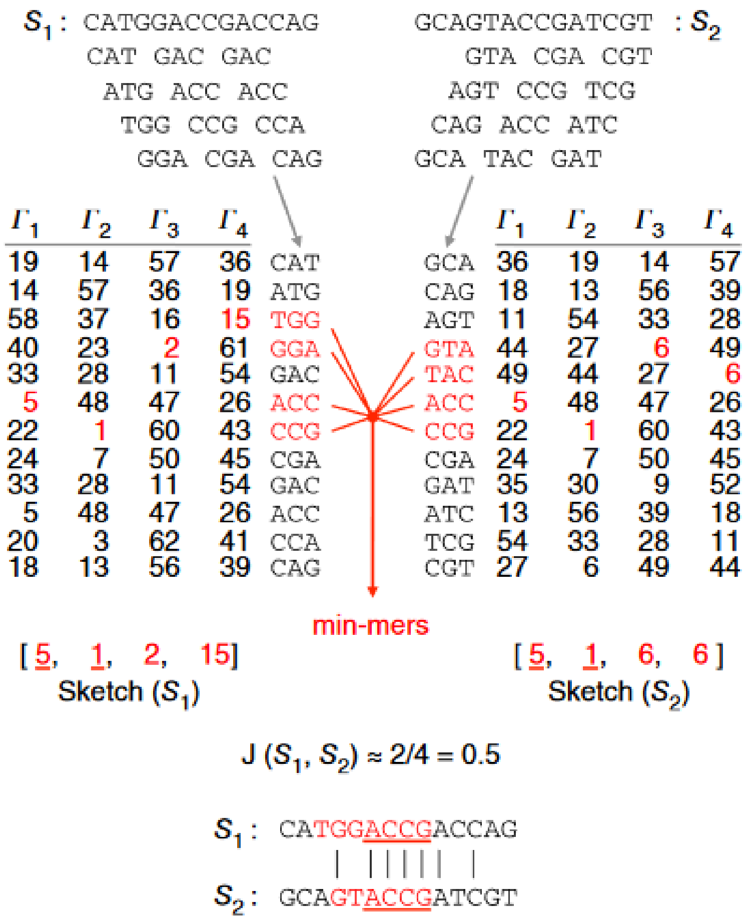
1. ***Shingling:*** Convert documents to sets

2. ***Min-Hashing:*** Convert large sets to short signatures, while preserving similarity

3. ***Locality-Sensitive Hashing:*** Focus on pairs of signatures likely to be from similar documents

    - **Candidate pairs!**

Min-Hash , Jaccard similarity:



S_1: CATGGACCGACCAG

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

GCAGTACCGATCGT : S_2

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

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

	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

min-mers

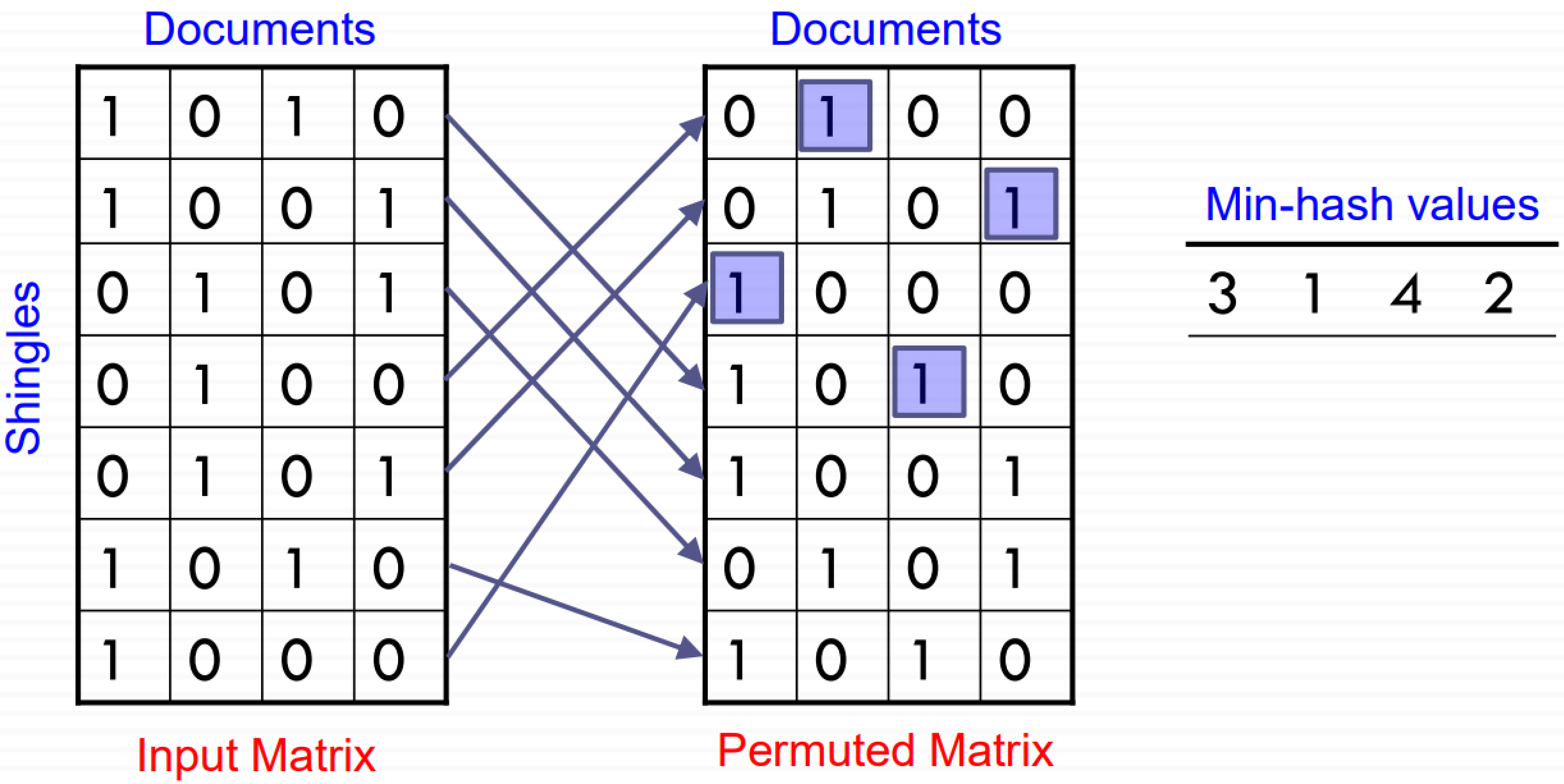
[5, 1, 2, 15]
Sketch (S_1)

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

J (S_1, S_2) ≈ 2/4 = 0.5

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

Which k is the most suitable ?



Documents

1	0	1	0
1	0	0	1
0	1	0	1
0	1	0	0
0	1	0	1
1	0	1	0
1	0	0	0

Input Matrix

Documents

0	**1**	0	0
0	1	0	**1**
1	0	0	0
1	0	**1**	0
1	0	0	1
0	1	0	1
1	0	1	0

Permuted Matrix

Min-hash values

3	1	4	2

- **Choose a random permutation π**
- **Claim:** $\Pr[h_\pi(C_i) = h_\pi(C_j)] = sim(C_i, C_j)$
- **Proof:**
  - Consider 3 types of rows:

    type X: C_i and C_j both have 1s

    type Y: only one of C_i and C_j has 1

    type Z: C_i and C_j both have 0s
  - After random permutation π, what if the first X-type row is before the first Y-type row?

    $$h_\pi(C_i) = h_\pi(C_j)$$

	C_i		C_j	
X	1		1	
Y	1		0	
Z	0		0	
Z	0		0	
Z	0		0	
X	1		1	
Y	1		0	

Input Matrix

- What is the probability that the first not-Z row is of type X?

$$\frac{|X|}{|X|+|Y|}$$

- $\mathrm{Pr}[h_\pi(\mathrm{C}_\mathrm{i}) = h_\pi(\mathrm{C}_\mathrm{j})] = \frac{|X|}{|X|+|Y|}$

- $\mathrm{sim}(\mathrm{C}_\mathrm{i}, \mathrm{C}_\mathrm{j}) = \frac{|C_i \cap C_\mathrm{j}|}{|C_i \cup C_\mathrm{j}|} = \frac{|X|}{|X|+|Y|} = \mathrm{Pr}[h_\pi(\mathrm{C}_\mathrm{i}) = h_\pi(\mathrm{C}_\mathrm{j})]$

- **Conclusion:** $\mathrm{Pr}[h_\pi(\mathrm{C}_\mathrm{i}) = h_\pi(\mathrm{C}_\mathrm{j})] = sim(\mathrm{C}_\mathrm{i}, \mathrm{C}_\mathrm{j})$

Permutation π

2	4	3
3	2	4
7	1	7
6	3	2
1	6	6
5	7	1
4	5	5

Input matrix (Shingles x Documents)

1	0	1	0
1	0	0	1
0	1	0	1
0	1	0	1
0	1	0	1
1	0	1	0
1	0	1	0

Signature matrix *M*

2	1	2	1
2	1	4	1
1	2	1	2

Similarities:

	1-3	2-4	1-2	3-4
Col/Col	0.75	0.75	0	0
Sig/Sig	0.67	1.00	0	0

Don’t worry about uncertainty …

- **Suppose we need to find near-duplicate documents among $N = 1$ million documents**

- Naïvely, we would have to compute **pairwise Jaccard similarities** for **every pair of docs**
  - $\boldsymbol{N(N-1)/2 \approx 5*10^{11}}$ comparisons
  - At 10^5 secs/day and 10^6 comparisons/sec, it would take **5 days**

- For $N = 10$ million, it takes more than a year…

Documents

Shingles

1	1	1	0
1	1	0	1
0	1	0	1
0	0	0	1
1	0	0	1
1	1	1	0
1	0	1	0

2	1	4	1
1	2	1	2
2	1	2	1

LSH: First Cut

2	1	4	1
1	2	1	2
2	1	2	1

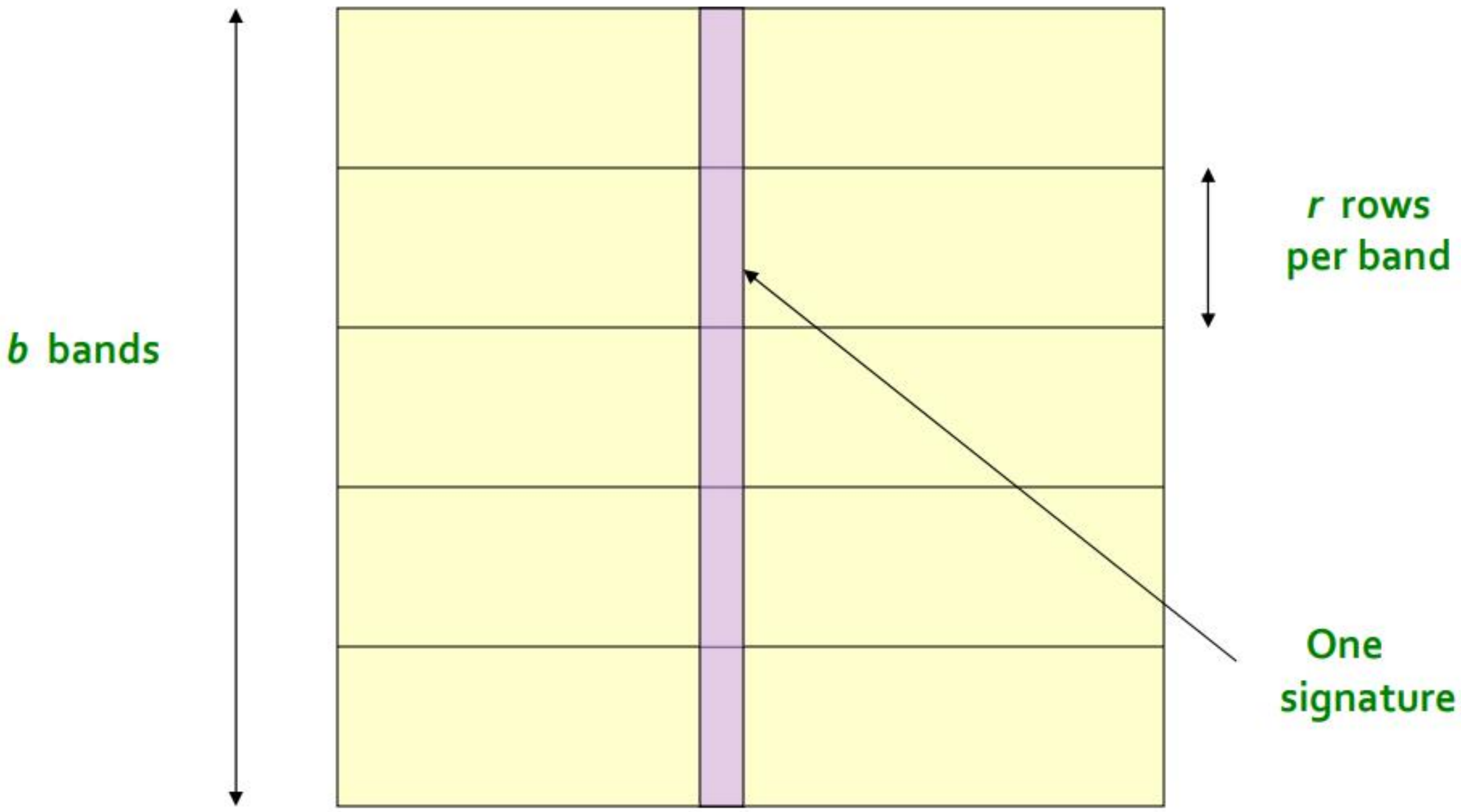
- **Goal:** Find documents with Jaccard similarity at least ***s*** (for some similarity threshold, e.g., ***s***=0.8)

- **LSH – General idea:** Use a function ***f(x,y)*** that tells whether ***x*** and ***y*** is a ***candidate pair****:* a pair of elements whose similarity must be evaluated

- **For Min-Hash matrices:**
  - Hash columns of signature matrix ***M*** to many buckets
  - Each pair of documents that hashes into the same bucket is a **candidate pair**

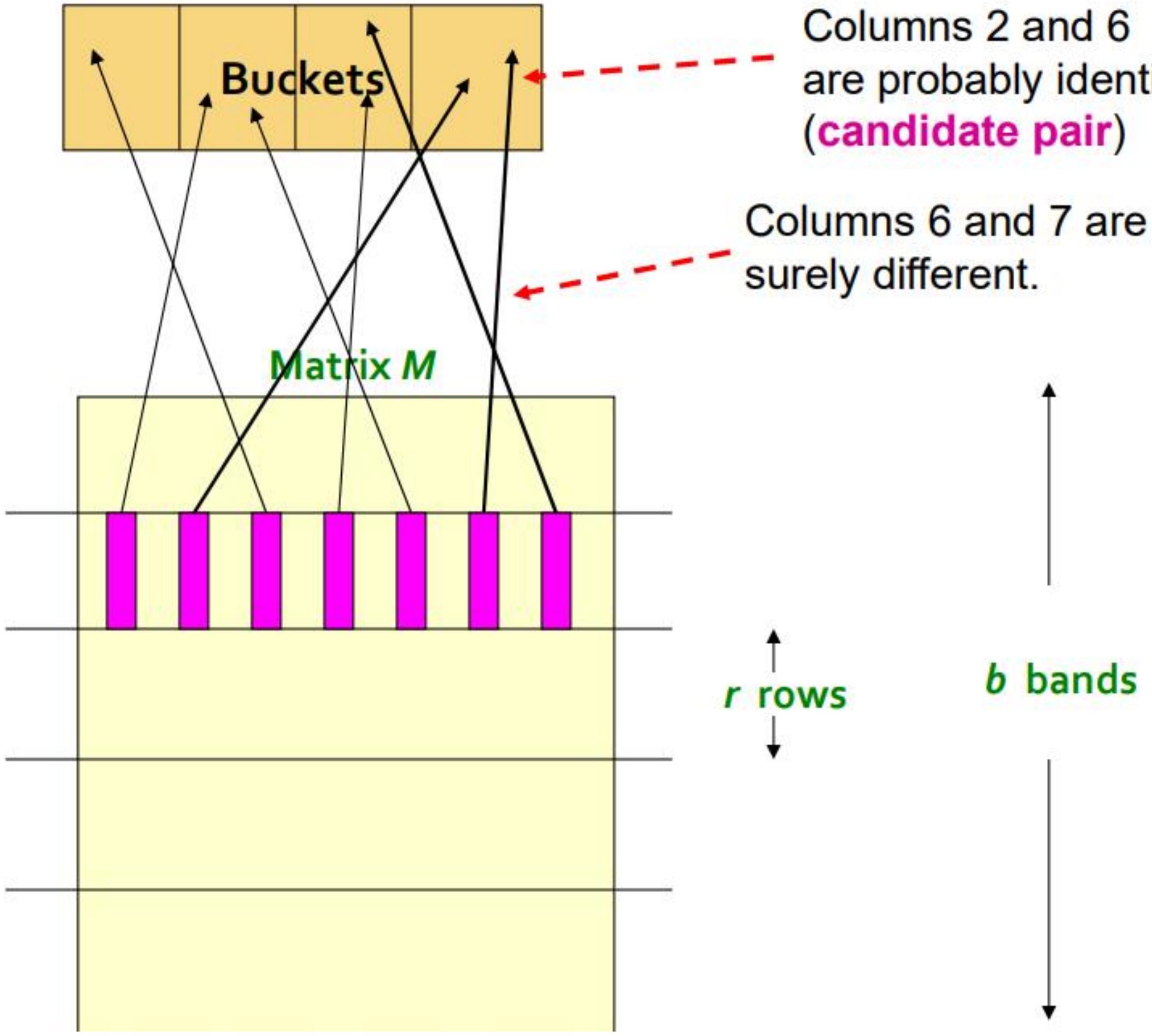
Partition *M* into *b* Bands

2	1	4	1
1	2	1	2
2	1	2	1



Signature matrix *M*

Hashing Bands

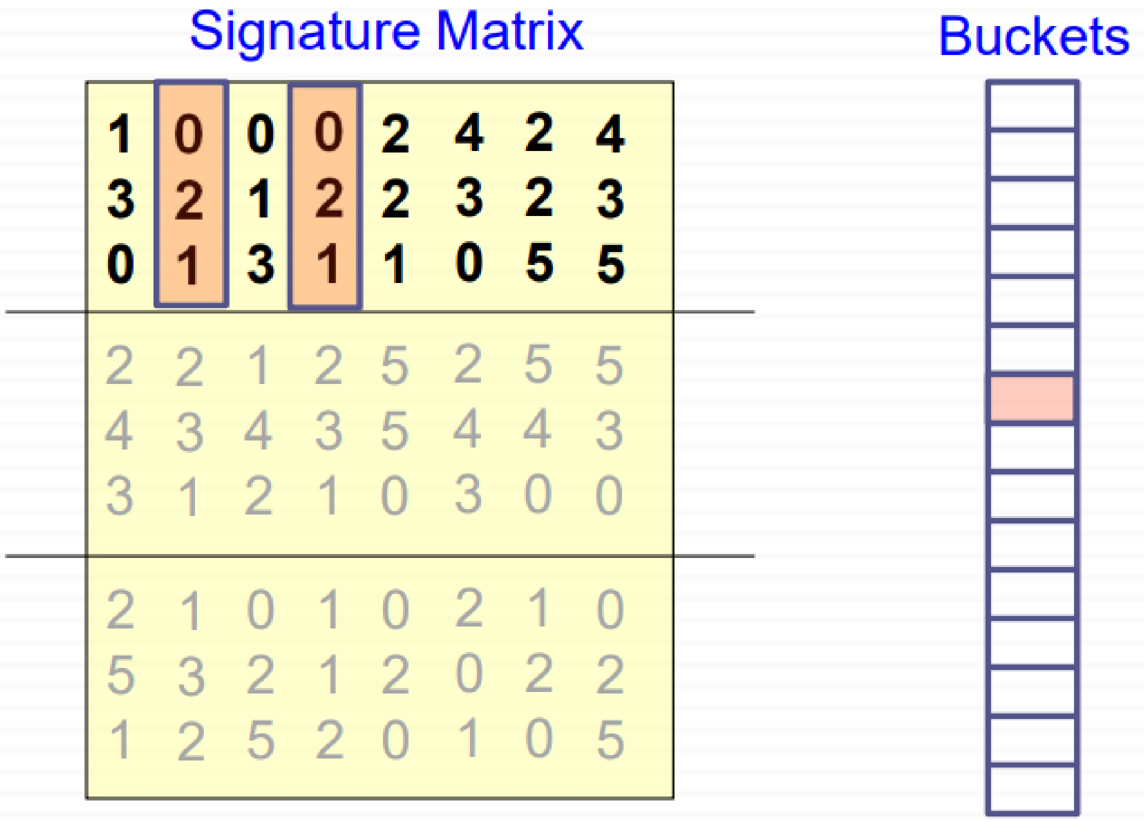


Columns 2 and 6
are probably identical
(**candidate pair**)

Columns 6 and 7 are
surely different.

we can go deeper,Minia

Banding Example



1	0	0	0	2	4	2	4
3	2	1	2	2	3	2	3
0	1	3	1	1	0	5	5
2	2	1	2	5	2	5	5
4	3	4	3	5	4	4	3
3	1	2	1	0	3	0	0
2	1	0	1	0	2	1	0
5	3	2	1	2	0	2	2
1	2	5	2	0	1	0	5

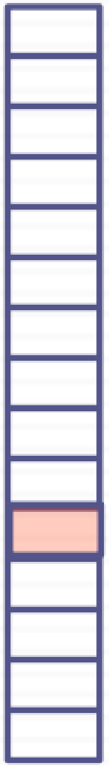
Candidate pairs: {(2,4);

Banding Example

Signature Matrix

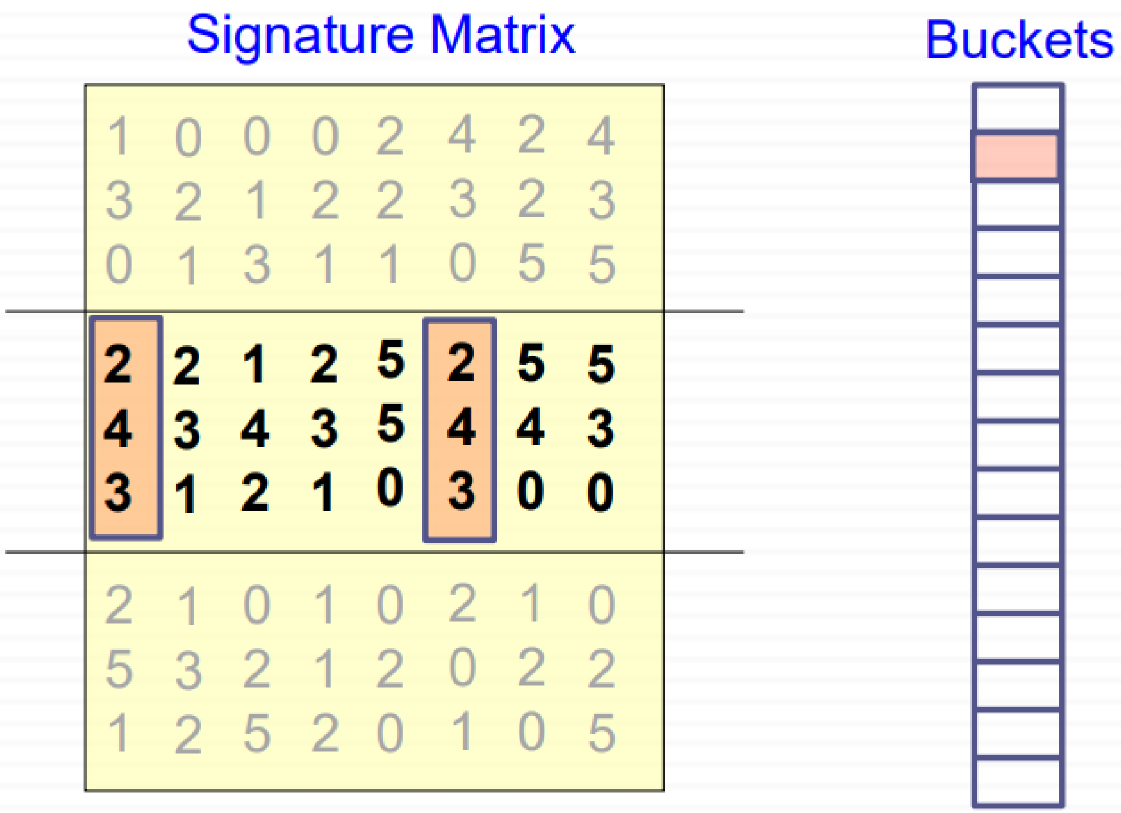
1	0	0	0	2	4	2	4
3	2	1	2	2	3	2	3
0	1	3	1	1	0	5	5
2	**2**	**1**	**2**	**5**	**2**	**5**	**5**
4	**3**	**4**	**3**	**5**	**4**	**4**	**3**
3	**1**	**2**	**1**	**0**	**3**	**0**	**0**
2	1	0	1	0	2	1	0
5	3	2	1	2	0	2	2
1	2	5	2	0	1	0	5

Buckets



Candidate pairs: {(2,4);

Banding Example



1	0	0	0	2	4	2	4
3	2	1	2	2	3	2	3
0	1	3	1	1	0	5	5
2	**2**	**1**	**2**	**5**	**2**	**5**	**5**
4	**3**	**4**	**3**	**5**	**4**	**4**	**3**
3	**1**	**2**	**1**	**0**	**3**	**0**	**0**
2	1	0	1	0	2	1	0
5	3	2	1	2	0	2	2
1	2	5	2	0	1	0	5

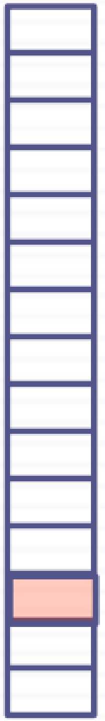
Candidate pairs: {(2,4); (1,6)

Banding Example

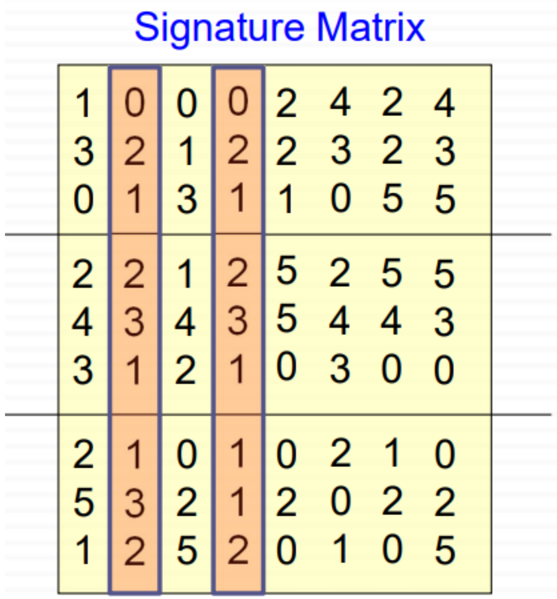
Signature Matrix

1	0	0	0	2	4	2	4
3	2	1	2	2	3	2	3
0	1	3	1	1	0	5	5
2	2	1	2	5	2	5	5
4	3	4	3	5	4	4	3
3	1	2	1	0	3	0	0
2	**1**	**0**	**1**	**0**	**2**	**1**	**0**
5	**3**	**2**	**1**	**2**	**0**	**2**	**2**
1	**2**	**5**	**2**	**0**	**1**	**0**	**5**

Buckets



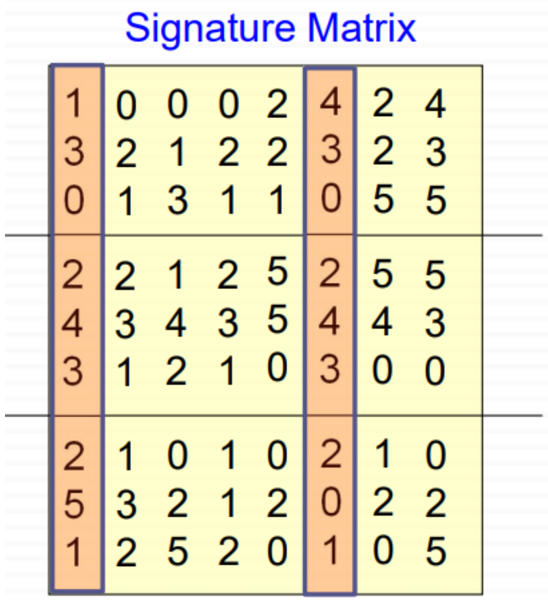
Candidate pairs: {(2,4); (1,6) (3,8)}



Signature Matrix

1	**0**	0	**0**	2	4	2	4
3	**2**	1	**2**	2	3	2	3
0	**1**	3	**1**	1	0	5	5
2	**2**	1	**2**	5	2	5	5
4	**3**	4	**3**	5	4	4	3
3	**1**	2	**1**	0	3	0	0
2	**1**	0	**1**	0	2	1	0
5	**3**	2	**1**	2	0	2	2
1	**2**	5	**2**	0	1	0	5

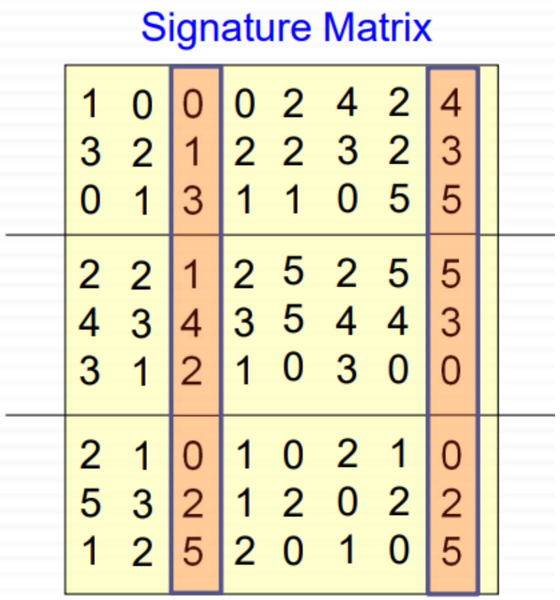
True positive



Signature Matrix

1	0	0	0	2	**4**	2	4
3	2	1	2	2	**3**	2	3
0	1	3	1	1	**0**	5	5
2	2	1	2	5	**2**	5	5
4	3	4	3	5	**4**	4	3
3	1	2	1	0	**3**	0	0
2	1	0	1	0	**2**	1	0
5	3	2	1	2	**0**	2	2
1	2	5	2	0	**1**	0	5

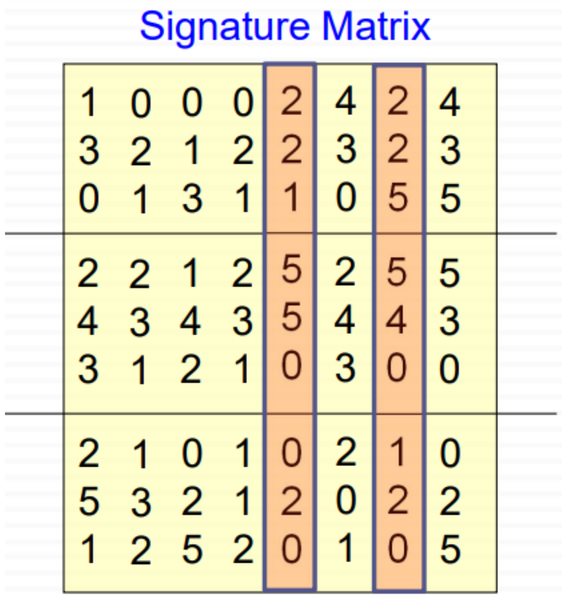
True positive



Signature Matrix

1	0	**0**	0	2	4	2	**4**
3	2	**1**	2	2	3	2	**3**
0	1	**3**	1	1	0	5	**5**
2	2	**1**	2	5	2	5	**5**
4	3	**4**	3	5	4	4	**3**
3	1	**2**	1	0	3	0	**0**
2	1	**0**	1	0	2	1	**0**
5	3	**2**	1	2	0	2	**2**
1	2	**5**	2	0	1	0	**5**

False positive?



Signature Matrix

1	0	0	0	**2**	4	**2**	4
3	2	1	2	**2**	3	**2**	3
0	1	3	1	**1**	0	**5**	5
2	2	1	2	**5**	2	**5**	5
4	3	4	3	**5**	4	**4**	3
3	1	2	1	**0**	3	**0**	0
2	1	0	1	**0**	2	**1**	0
5	3	2	1	**2**	0	**2**	2
1	2	5	2	**0**	1	**0**	5

False negative?

b bands, *r* rows/band

- Columns C_1 and C_2 have similarity ***t***
- Pick any band (***r*** rows)
  - Prob. that all rows in band equal

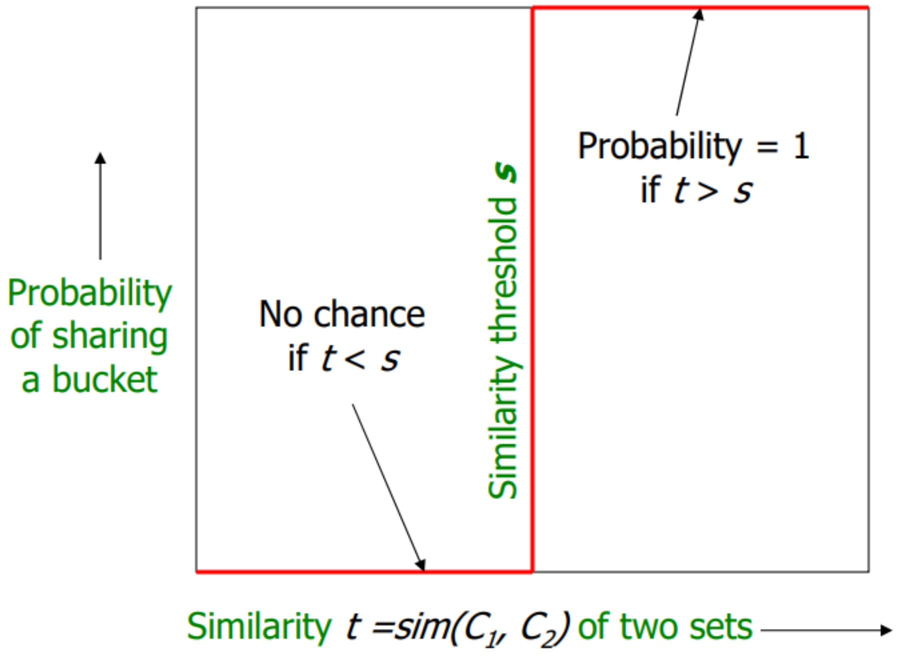
    $$t^r$$
  - Prob. that some row in band unequal

    $$1 - t^r$$
- Prob. that no band identical

  $$(1 - t^r)^b$$
- Prob. that at least 1 band identical

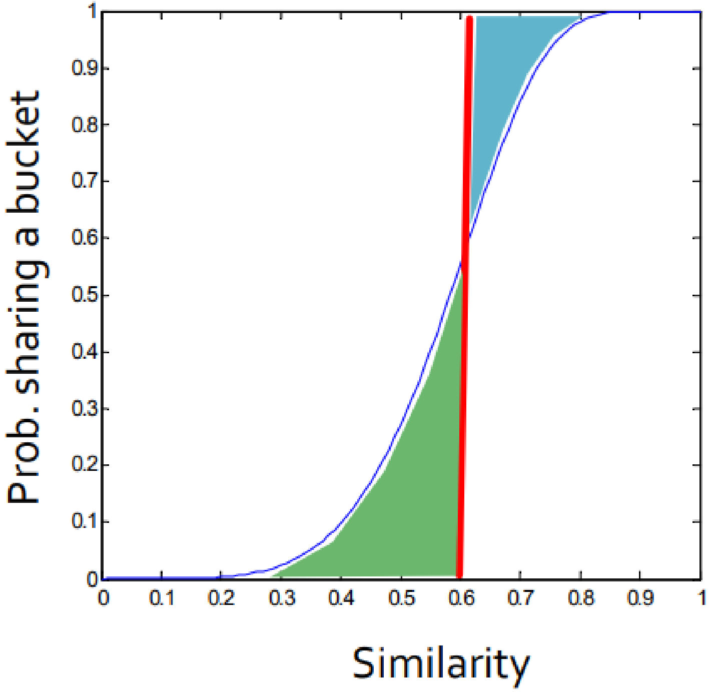
  $$1 - (1 - t^r)^b$$

What we wish to have:



Picking *r* and *b*: The S-curve

- **Picking *r* and *b* to get the best S-curve**
  - 50 hash-functions (r=5, b=10)

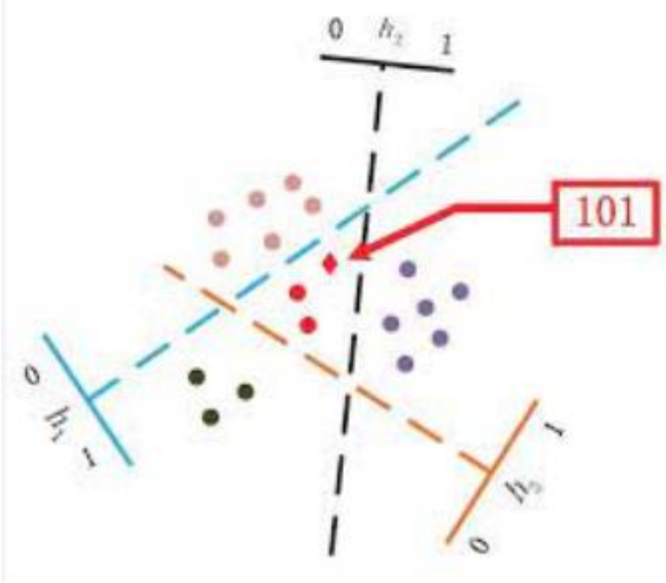


Green area: False Positive rate

Blue area: False Negative rate

Can we use LSH for other distance measures?

- Jaccard distance
- Cosine distance
- Euclidean distance , l_2 , l_1 etc .
- We could go beyond … .



Epilogue

- Biology face with large mysterious text with specific structure .so computer scientific techniques could give us many good tools to deal with this complexities.
- Beside LSH , many other algorithmic techniques (such as Bloom Filter)

Has been developed to study genomic structure as efficient as we can.

Thanks for your Attention ☺

References :

- Assembling large genomes with single-molecule sequencing and locality-sensitive hashing
- leskovec et al.:Massive-Data-Mining/CS246W/stanford
- Dr.Koohi lec notes bio Informatic algorithms course CE – 1402.01
- Dr.Gholampour lec notes Data Mining course EE – 1402.01