Sketching and locality sensitive hashing for alignment

Guillaume Marçais

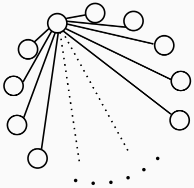
2/15/23

Why do we need sketching and Locality Sensitive Hashing for alignment?

Large scale alignment problems

Cluster N samples based on sequence similarity

- $\rightarrow N^2/2$ alignment problems
- Speed-up pairwise alignment task?
- Skip hopeless alignments?



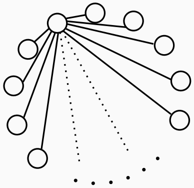
Sequence search in large database

- Avoid aligning to all sequences in database?
- Approximate nearer neighbor search
- High dimension, non-geometric space

Large scale alignment problems

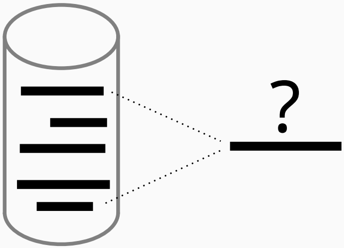
Cluster N samples based on sequence similarity

- $\rightarrow N^2/2$ alignment problems
- Speed-up pairwise alignment task?
- Skip hopeless alignments?

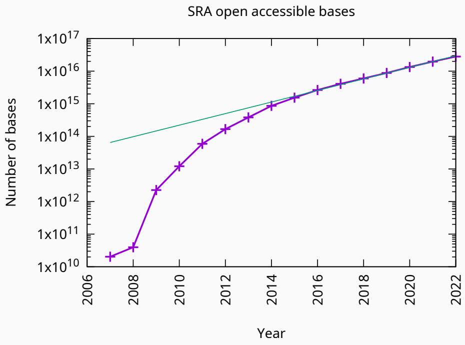


Sequence search in large database

- Avoid aligning to all sequences in database?
- Approximate nearer neighbor search
- High dimension, non-geometric space



Fast growth of sequence databases



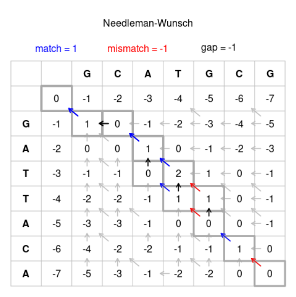
- Exponential growth in public and private databases (SRA: $1.5\times$/year)
- $\implies$ hidden exponential slow down in large scale analysis

Sequence alignment is hard

No strongly subquadratic time algorithm, most likely (Backurs, Indyk 2015)

Computing the edit distance $\mathrm{E_d}$ in time $O(n^{2-\delta}), \delta > 0$ violates the Strong Exponential Time Hypothesis (SETH).

- Usual dynamic programming: $O(n^2)$
- [1]Masek and Paterson: $O\left(\frac{n^2}{\log(n)}\right)$
- $n^{2-\delta} \ll \frac{n^2}{\log(n)} \ll n^2$
- Can't fundamentally improve



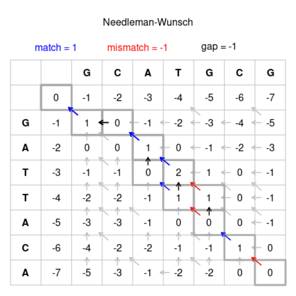
[1] A faster algorithm computing string edit distances (1980)

Sequence alignment is hard

No strongly subquadratic time algorithm, most likely (Backurs, Indyk 2015)

Computing the edit distance $\mathrm{E_d}$ in time $O(n^{2-\delta}), \delta > 0$ violates the Strong Exponential Time Hypothesis (SETH).

- Usual dynamic programming: $O(n^2)$
- [1]Masek and Paterson: $O\left(\frac{n^2}{\log(n)}\right)$
- $n^{2-\delta} \ll \frac{n^2}{\log(n)} \ll n^2$
- Can't fundamentally improve



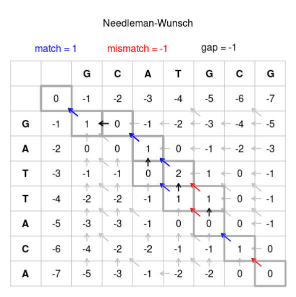
[1]A faster algorithm computing string edit distances (1980)

Sequence alignment is hard

No strongly subquadratic time algorithm, most likely (Backurs, Indyk 2015)

Computing the edit distance $\mathrm{E_d}$ in time $O(n^{2-\delta}), \delta > 0$ violates the Strong Exponential Time Hypothesis (SETH).

- Usual dynamic programming: $O(n^2)$
- [1]Masek and Paterson: $O\left(\frac{n^2}{\log(n)}\right)$
- $n^{2-\delta} \ll \frac{n^2}{\log(n)} \ll n^2$
- Can't fundamentally improve



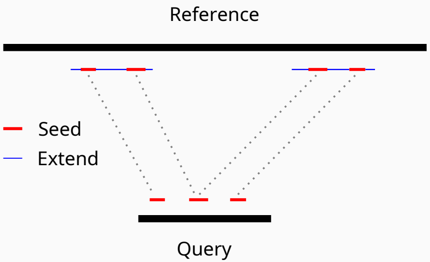
[1] A faster algorithm computing string edit distances (1980)

Seed and extend paradigm

Main paradigm:

- Find seeds (small exact matches)
- Cluster "coherent" seeds
- Extend between seeds using DP

- Used since the 90s' (Blast, MUMmer)
- Still computationally intensive for large scale
- Many ways to find seeds:
  - k-mers
  - Suffix trees/arrays, FM Index
  - LSH / sketching



Sketching / Locality Sensitive Hashing

Avoid computing edit distance directly, use proxy measures easier to compute

- LSH: hashing method to avoid fruitless comparisons
- Sketching: sparse representation allowing quick comparison

Locality Sensitive Hashing: Make collisions matters

$\mathcal{U}$: universe. T: hash table. $|T| \ll |\mathcal{U}|$. $h : \mathcal{U} \to [0, |T| - 1]$.
$\mathcal{H} = \{h : \mathcal{U} \to [0, |T| - 1]\}$

$S =$ `AACGGTG`

$h(S) = 2$

T

0	
1	
2	S
3	
4	

Universal Hashing

- Collisions as rare as possible
- $\forall x, y \in \mathcal{U}, x \neq y,$

$$\Pr_{h \in \mathcal{H}}[h(x) = h(y)] = \frac{1}{|T|}$$

Locality Sensitive Hashing

- Collision between similar elements
- $\forall x, y \in \mathcal{U}$

$$\mathrm{E_d}(x, y) \leq d_1 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \geq p_1$$

$$\mathrm{E_d}(x, y) \geq d_2 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq p_2$$

Locality Sensitive Hashing: Make collisions matters

$\mathcal{U}$: universe. T: hash table. $|T| \ll |\mathcal{U}|$. $h : \mathcal{U} \to [0, |T| - 1]$.
$\mathcal{H} = \{h : \mathcal{U} \to [0, |T| - 1]\}$

$S =$ AACGGTG

$h(S) = 2$

T

0	
1	
2	S
3	
4	

Universal Hashing

- Collisions as rare as possible
- $\forall x, y \in \mathcal{U}, x \neq y,$

$$\Pr_{h \in \mathcal{H}}[h(x) = h(y)] = \frac{1}{|T|}$$

Locality Sensitive Hashing

- Collision between similar elements
- $\forall x, y \in \mathcal{U}$

$$\mathrm{E_d}(x, y) \leq d_1 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \geq p_1$$

$$\mathrm{E_d}(x, y) \geq d_2 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq p_2$$

Locality Sensitive Hashing: Make collisions matters

$\mathcal{U}$: universe. T: hash table. $|T| \ll |\mathcal{U}|$. $h : \mathcal{U} \to [0, |T|-1]$.
$\mathcal{H} = \{h : \mathcal{U} \to [0, |T|-1]\}$

$S =$ `AACGGTG`

$h(S) = 2$

T

0	
1	
2	S
3	
4	

Universal Hashing

- Collisions as rare as possible
- $\forall x, y \in \mathcal{U}, x \neq y,$

$$\Pr_{h \in \mathcal{H}}[h(x) = h(y)] = \frac{1}{|T|}$$

Locality Sensitive Hashing

- Collision between similar elements
- $\forall x, y \in \mathcal{U}$

$$\mathrm{E_d}(x, y) \leq d_1 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \geq p_1$$
$$\mathrm{E_d}(x, y) \geq d_2 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq p_2$$

Locality Sensitive Hashing Definition

The family $\mathcal{H}$ is "(d_1, d_2, p_1, p_2)-sensitive" for distance D if there exists $d_1 < d_2$, $p_1 > p_2$ such that for all $x, y \in \mathcal{U}$

$$\mathrm{D}(x,y) \leq d_1 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \geq p_1$$

$$\mathrm{D}(x,y) \geq d_2 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq p_2$$

- **Low distance** $\iff$ **High collisions**
- **High distance** $\iff$ **Low collisions**
- **In between** d_1, d_2: **No guarantee**

Locality sensitive hash family

Family $\mathcal{H}$ of hash functions where similar elements are more likely to have the same value than distant elements.

Locality Sensitive Hashing Definition

The family $\mathcal{H}$ is "(d_1, d_2, p_1, p_2)-sensitive" for distance D if there exists $d_1 < d_2$, $p_1 > p_2$ such that for all $x, y \in \mathcal{U}$

$$\mathrm{D}(x,y) \leq d_1 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \geq p_1$$

$$\mathrm{D}(x,y) \geq d_2 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq p_2$$

- **Probability over choice of $h \in \mathcal{H}$, not over the elements x, y**

Locality sensitive hash family

Family $\mathcal{H}$ of hash functions where similar elements are more likely to have the same value than distant elements.

Locality Sensitive Hashing Definition

The family $\mathcal{H}$ is "(d_1, d_2, p_1, p_2)-sensitive" for distance D if there exists $d_1 < d_2$, $p_1 > p_2$ such that for all $x, y \in \mathcal{U}$

$$\mathrm{D}(x,y) \leq d_1 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \geq p_1$$

$$\mathrm{D}(x,y) \geq d_2 \implies \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq p_2$$

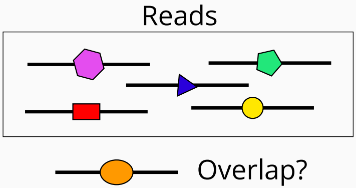
- $d_1 < d_2$: "gapped" LSH
- $d_1 = d_2$, "ungapped" LSH
- Gap not desirable but not always avoidable.

Locality sensitive hash family

Family $\mathcal{H}$ of hash functions where similar elements are more likely to have the same value than distant elements.

Overlap computation

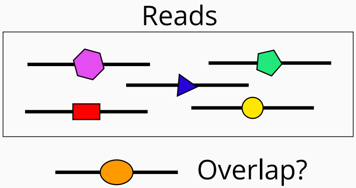
- Compute overlaps between reads (MHAP[2])
- Instance of "Nearest Neighbor Problem" for edit distance
- Use multiple hash tables
- Orange ellipse in same location as yellow circle



[2] Assembling large genomes with single-molecule sequencing and locality-sensitive hashing

Overlap computation

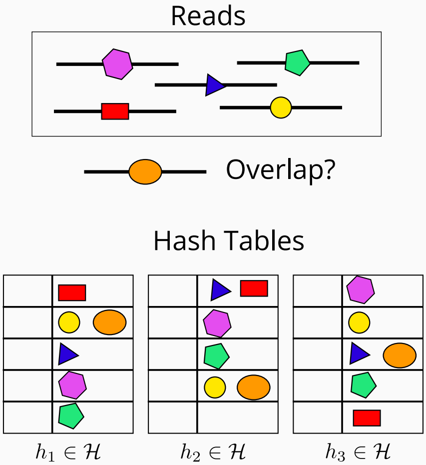
- Compute overlaps between reads (MHAP[2])
- Instance of "Nearest Neighbor Problem" for edit distance
- Use multiple hash tables
- Orange ellipse in same location as yellow circle



[2] Assembling large genomes with single-molecule sequencing and locality-sensitive hashing

Overlap computation

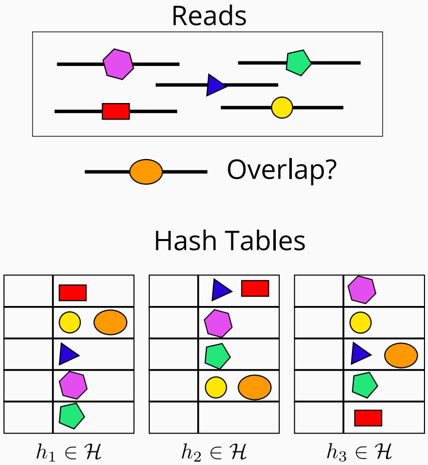
- Compute overlaps between reads (MHAP[2])
- Instance of "Nearest Neighbor Problem" for edit distance
- Use multiple hash tables
- Orange ellipse in same location as yellow circle



[2] Assembling large genomes with single-molecule sequencing and locality-sensitive hashing

Overlap computation

- Compute overlaps between reads (MHAP[2])
- Instance of "Nearest Neighbor Problem" for edit distance
- Use multiple hash tables
- Orange ellipse in same location as yellow circle



[2] Assembling large genomes with single-molecule sequencing and locality-sensitive hashing

How to design an LSH for edit distance?

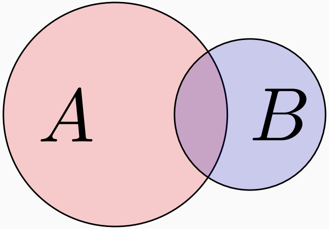
- **minHash: LSH for k-mer Jaccard distance**
- OMH: Ordered Min Hash

How to design an LSH for edit distance?

- **minHash: LSH for k-mer Jaccard distance**
- OMH: Ordered Min Hash

Jaccard distance

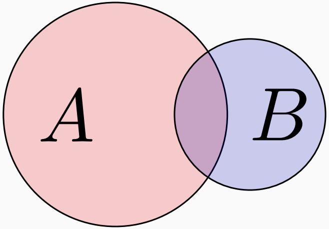
Jaccard distance between sets A, B:



$$\mathrm{J_d}(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

Jaccard distance

Jaccard distance between sets A, B:



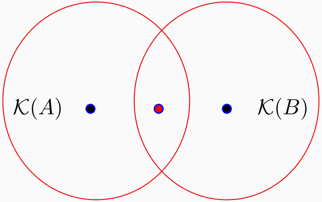
$$\mathrm{J_d}(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

Jaccard between sequences x, y:
Jaccard distance of their k-mer sets

$$\mathrm{J_d}(x, y) = \mathrm{J_d}(\mathcal{K}(x), \mathcal{K}(y))$$

- Low $\mathrm{E_d}(x, y) \implies$ Low $\mathrm{J_d}(x, y)$
- High $\mathrm{E_d}(x, y) \not\Longrightarrow$ High $\mathrm{J_d}(x, y)$
- Can have false positive, few false negative

MinHash: an LSH for the Jaccard distance



- Permutation of k-mers: $\pi : 4^k \to 4^k$ one-to-one

  $$\mathcal{H} = \{h_\pi(S) = \operatorname*{argmin}_{m \in \mathcal{K}(\mathcal{S})} \pi(m) \mid \pi \text{ permutation of } k\text{-mers}\}$$

- Fix π, every k-mer of $A \cup B$ equally likely to be the minimum for π

  $$\Pr_{h \in \mathcal{H}}[h(A) = h(B)] = \frac{|A \cap B|}{|A \cup B|}$$

- Unbiased estimator, ungapped LSH

minHash sketch: dimensionality reduction

- Choose L hash functions from $\mathcal{H}$: $h_i, 1 \leq i \leq L$
- Sketch of S: vector $\mathrm{Sk}(S) = (h_i(S))_{1 \leq i \leq L}$
- Big compression: Mash[3] $L = 1000, k = 21$, $7000 \times$ compression
- Very fast pairwise comparison (Hamming distance between sketches)

$$\mathrm{Sk}(A) = \begin{pmatrix} \texttt{CGAG} \\ \texttt{TTAC} \\ \texttt{CATC} \\ \texttt{CCAT} \\ \texttt{CATG} \\ \texttt{ACAA} \end{pmatrix}, \mathrm{Sk}(B) = \begin{pmatrix} \texttt{GTTT} \\ \texttt{TTAC} \\ \texttt{GTAG} \\ \texttt{ATTT} \\ \texttt{ACCC} \\ \texttt{ACAA} \end{pmatrix} \quad \rightarrow \quad \mathrm{J_d}(\mathcal{K}(A), \mathcal{K}(B)) \approx 1 - \frac{2}{6}$$

[3] Mash: fast genome and metagenome distance estimation using MinHash

OMH: LSH for the edit distance

- minHash: LSH for k-mer Jaccard distance
- **OMH: Ordered Min Hash**

Jaccard ignores k-mer repetition

$$x = \overbrace{\texttt{AAAAAAAAAAAAAAA}}^{n-k}\overbrace{\texttt{CCCCC}}^{k}$$

$$y = \underbrace{\texttt{AAAAA}}_{k}\underbrace{\texttt{CCCCCCCCCCCCCCC}}_{n-k}$$

Jaccard ignores k-mer repetition

$$x = \overbrace{\texttt{AAAAAAAAAAAAAAA}}^{n-k}\overbrace{\texttt{CCCCC}}^{k} \quad \rightarrow \{\texttt{AAAAA}, \texttt{AAAAC}, \texttt{AAACC}, \texttt{AACCC}, \texttt{ACCCC}, \texttt{CCCCC}\}$$

$$y = \underbrace{\texttt{AAAAA}}_{k}\underbrace{\texttt{CCCCCCCCCCCCCCC}}_{n-k} \quad \rightarrow \{\texttt{AAAAA}, \texttt{AAAAC}, \texttt{AAACC}, \texttt{AACCC}, \texttt{ACCCC}, \texttt{CCCCC}\}$$

Jaccard ignores k-mer repetition

$$x = \overbrace{\texttt{AAAAAAAAAAAAAAA}}^{n-k}\overbrace{\texttt{CCCCC}}^{k} \quad \to \{\texttt{AAAAA}, \texttt{AAAAC}, \texttt{AAACC}, \texttt{AACCC}, \texttt{ACCCC}, \texttt{CCCCC}\}$$

$$y = \underbrace{\texttt{AAAAA}}_{k}\underbrace{\texttt{CCCCCCCCCCCCCCC}}_{n-k} \quad \to \{\texttt{AAAAA}, \texttt{AAAAC}, \texttt{AAACC}, \texttt{AACCC}, \texttt{ACCCC}, \texttt{CCCCC}\}$$

Jaccard distance $\mathrm{J_d}(x, y) = 0$ Edit distance $\mathrm{E_d}(x, y) \geq 1 - \frac{2k}{n}$

Identical k-mer content and high edit distance

Weighted Jaccard: Jaccard on multi-set

- $\chi_A : \mathcal{U} \to \{0, 1\}$,
  $\chi_A(x) = 1 \iff x \in A$

$$\mathrm{J}(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\sum_{x \in \mathcal{U}} \min(\chi_A(x), \chi_B(x))}{\sum_{x \in \mathcal{U}} \max(\chi_A(x), \chi_B(x))}$$

- $\chi_A^w : \mathcal{U} \to \mathbb{N}$,
  $\chi_A^w(x) = \#$ of instances of x in A

Weighted Jaccard: Jaccard on multi-set

- $\chi_A : \mathcal{U} \to \{0, 1\}$,
  $\chi_A(x) = 1 \iff x \in A$

$$\mathrm{J}(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\sum_{x \in \mathcal{U}} \min(\chi_A(x), \chi_B(x))}{\sum_{x \in \mathcal{U}} \max(\chi_A(x), \chi_B(x))}$$

- $\chi_A^w : \mathcal{U} \to \mathbb{N}$,
  $\chi_A^w(x) = \#$ of instances of x in A

$$\mathrm{J}^{\mathrm{w}}(A, B) = \frac{\sum_{x \in \mathcal{U}} \min(\chi_A^w(x), \chi_B^w(x))}{\sum_{x \in \mathcal{U}} \max(\chi_A^w(x), \chi_B^w(x))}$$

Weighted Jaccard handles repetitions

$$x = \overbrace{\texttt{AAAAAAAAAAAAAAAA}}^{n-k}\overbrace{\texttt{CCCCC}}^{k} \rightarrow \left\{\begin{matrix}(\texttt{AAAAA},1),(\texttt{AAAAA},2),\ldots,(\texttt{AAAAA},11)\\(\texttt{AAAAC},1),(\texttt{AAACC},1),(\texttt{AACCC},1),(\texttt{ACCCC},1),(\texttt{CCCCC},1)\end{matrix}\right\}$$

$$y = \underbrace{\texttt{AAAAA}}_{k}\underbrace{\texttt{CCCCCCCCCCCCCCCC}}_{n-k} \rightarrow \left\{\begin{matrix}(\texttt{AAAAA},1),(\texttt{AAAAC},1),(\texttt{AAACC},1),(\texttt{AACCC},1),(\texttt{ACCCC},1),\\(\texttt{CCCCC},1),(\texttt{CCCCC},2),\ldots,(\texttt{CCCCC},11)\end{matrix}\right\}$$

Weighted Jaccard $\mathrm{J}_{\mathrm{d}}^{\mathrm{w}}(x, y) = 1 - \frac{k+2}{n}$ Edit distance $\mathrm{E}_{\mathrm{d}}(x, y) \geq 1 - \frac{2k}{n}$

Weighted Jaccard = Jaccard for multi-sets

Jaccard and weighted Jaccard ignore relative order

$x = \texttt{CCCCACCAACACAAAACCC}$

$y = \texttt{AAAACACAACCCCACCAAA}$

Jaccard and weighted Jaccard ignore relative order

$x = \texttt{CCCCACCAACACAAAACCC}$ $\rightarrow \left\{ \begin{matrix} \texttt{AAAA}, \texttt{AAAC}, \texttt{AACA}, \texttt{AACC}, \texttt{ACAA}, \texttt{ACAC}, \texttt{ACCA}, \texttt{ACCC} \\ \texttt{CAAA}, \texttt{CAAC}, \texttt{CACA}, \texttt{CACC}, \texttt{CCAA}, \texttt{CCAC}, \texttt{CCCA}, \texttt{CCCC} \end{matrix} \right\}$

$y = \texttt{AAAACACAACCCCACCAAA}$ $\rightarrow \left\{ \begin{matrix} \texttt{AAAA}, \texttt{AAAC}, \texttt{AACA}, \texttt{AACC}, \texttt{ACAA}, \texttt{ACAC}, \texttt{ACCA}, \texttt{ACCC} \\ \texttt{CAAA}, \texttt{CAAC}, \texttt{CACA}, \texttt{CACC}, \texttt{CCAA}, \texttt{CCAC}, \texttt{CCCA}, \texttt{CCCC} \end{matrix} \right\}$

x, y: de Bruijn sequences,
contain all 16 possible 4-mers once
$(\sigma!)^{\sigma^{k-1}}$ de Bruijn sequences of length $\sigma^k + \sigma - 1$

Jaccard and weighted Jaccard ignore relative order

$x = \texttt{CCCCACCAACACAAAACCC} \quad \rightarrow \left\{\begin{matrix}\texttt{AAAA,AAAC,AACA,AACC,ACAA,ACAC,ACCA,ACCC} \\ \texttt{CAAA,CAAC,CACA,CACC,CCAA,CCAC,CCCA,CCCC}\end{matrix}\right\}$

$y = \texttt{AAAACACAACCCCACCAAA} \quad \rightarrow \left\{\begin{matrix}\texttt{AAAA,AAAC,AACA,AACC,ACAA,ACAC,ACCA,ACCC} \\ \texttt{CAAA,CAAC,CACA,CACC,CCAA,CCAC,CCCA,CCCC}\end{matrix}\right\}$

x, y: de Bruijn sequences,
contain all 16 possible 4-mers once
$(\sigma!)^{\sigma^{k-1}}$ de Bruijn sequences of length $\sigma^k + \sigma - 1$

$$\mathrm{J_d}(x,y) = \mathrm{J^w_d}(x,y) = 0 \qquad \mathrm{E_d}(x,y) = 0.63$$

Jaccard is different from edit distance

Unlike edit distance, k-mer Jaccard is insensitive to:

1. k-mer repetitions
2. relative positions of k-mers

- k-mer Jaccard is not an LSH for the edit distance
- Still provides big computation saving: asymmetric error model

Jaccard is different from edit distance

Unlike edit distance, k-mer Jaccard is insensitive to:

1. k-mer repetitions
2. relative positions of k-mers

- k-mer Jaccard is not an LSH for the edit distance
- Still provides big computation saving: asymmetric error model

OMH: Order Min Hash

- minHash is an LSH for Jaccard
- OMH is a refinement of minHash
- OMH is sensitive to
  - repeated k-mers
  - relative order of k-mers

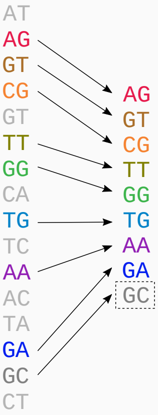
minHash & OMH sketches

$S =$ AGTTGAGCGGAAGGTG, $k = 2$

minHash & OMH sketches

$S = \texttt{AGTTGAGCGGAAGGTG},\ k = 2$

π: permutation of Σ^k



minHash & OMH sketches

$S =$ `AGTTGAGCGGAAGGTG`, $k = 2$, $L = 3$

π: permutation of Σ^k

1	2	3
AG	GG	CG
GT	GA	GA
CG	CG	TG
TT	AG	AG
GG	GC	GC
TG	GT	GG
AA	AA	TT
GA	TT	AA
GC	TG	GT

minHash & OMH sketches

$S = \texttt{AGTTGAGCGGAAGGTG},\ k = 2,\ L = 3$

π: permutation of Σ^k

1	2	3
AG	GG	CG
GT	GA	GA
CG	CG	TG
TT	AG	AG
GG	GC	GC
TG	GT	GG
AA	AA	TT
GA	TT	AA
GC	TG	GT

minHash & OMH sketches

$S = \texttt{AGTTGAGCGGAAGGTG},\ k = 2,\ L = 3$

π: permutation of Σ^k

1	2	3
AG	GG	CG
GT	GA	GA
CG	CG	TG
TT	AG	AG
GG	GC	GC
TG	GT	GG
AA	AA	TT
GA	TT	AA
GC	TG	GT

Order: permutation of $\Sigma^k \times \{1, \ldots, n\}$

GA, 4
TG, 3
AG, 5
GT, 1
GT, 13
AA, 10
AG, 11
TT, 2
AG, 0
CG, 7
GG, 12
GC, 6
TG, 14
GG, 8
GA, 9

minHash & OMH sketches

$S = \texttt{AGTTGAGCGGAAGGTG},\ k = 2,\ L = 3,\ \ell = 2$

π: permutation of Σ^k

1	2	3
AG	GG	CG
GT	GA	GA
CG	CG	TG
TT	AG	AG
GG	GC	GC
TG	GT	GG
AA	AA	TT
GA	TT	AA
GC	TG	GT

1	2	3	4	5	6
GA, 4	CG, 7	GT, 13	AG, 0	AA, 10	GA, 9
TG, 3	TG, 14	GA, 4	TT, 2	GT, 13	GG, 8
AG, 5	AG, 0	GA, 9	AG, 11	GA, 9	GC, 6
GT, 1	GA, 9	TG, 3	AG, 5	GT, 1	TG, 14
GT, 13	AG, 5	AG, 5	AA, 10	AG, 5	GT, 13
AA, 10	AG, 11	CG, 7	GT, 13	TT, 2	TT, 2
AG, 11	GA, 4	TT, 2	CG, 7	GA, 4	AA, 10
TT, 2	GT, 13	AA, 10	GG, 8	CG, 7	AG, 0
AG, 0	TT, 2	GG, 12	GA, 4	AG, 0	CG, 7
CG, 7	TG, 3	GG, 8	GA, 9	TG, 3	GG, 12
GG, 12	GG, 8	TG, 14	TG, 14	GG, 8	AG, 11
GC, 6	AA, 10	GT, 1	TG, 3	GG, 12	TG, 3
TG, 14	GG, 12	AG, 11	GC, 6	GC, 6	GT, 1
GG, 8	GT, 1	GC, 6	GT, 1	AG, 11	GA, 4
GA, 9	GC, 6	AG, 0	GG, 12	TG, 14	AG, 5

minHash & OMH sketches

$S =$ AGTTGAGCGGAAGGTG, $k = 2$, $L = 3$, $\ell = 2$

π: permutation of Σ^k

1	2	3
AG	GG	CG
GT	GA	GA
CG	CG	TG
TT	AG	AG
GG	GC	GC
TG	GT	GG
AA	AA	TT
GA	TT	AA
GC	TG	GT

1	2	3	4	5	6
GA, 4	CG, 7	GT, 13	AG, 0	AA, 10	GA, 9
TG, 3	TG, 14	GA, 4	TT, 2	GT, 13	GG, 8
AG, 5	AG, 0	GA, 9	AG, 11	GA, 9	GC, 6
GT, 1	GA, 9	TG, 3	AG, 5	GT, 1	TG, 14
GT, 13	AG, 5	AG, 5	AA, 10	AG, 5	GT, 13
AA, 10	AG, 11	CG, 7	GT, 13	TT, 2	TT, 2
AG, 11	GA, 4	TT, 2	CG, 7	GA, 4	AA, 10
TT, 2	GT, 13	AA, 10	GG, 8	CG, 7	AG, 0
AG, 0	TT, 2	GG, 12	GA, 4	AG, 0	CG, 7
CG, 7	TG, 3	GG, 8	GA, 9	TG, 3	GG, 12
GG, 12	GG, 8	TG, 14	TG, 14	GG, 8	AG, 11
GC, 6	AA, 10	GT, 1	TG, 3	GG, 12	TG, 3
TG, 14	GG, 12	AG, 11	GC, 6	GC, 6	GT, 1
GG, 8	GT, 1	GC, 6	GT, 1	AG, 11	GA, 4
GA, 9	GC, 6	AG, 0	GG, 12	TG, 14	AG, 5

minHash & OMH sketches

$S = \texttt{AGTTGAGCGGAAGGTG}$, $k = 2$, $L = 3$, $\ell = 2$

π: permutation of Σ^k

1	2	3
AG	GG	CG
GT	GA	GA
CG	CG	TG
TT	AG	AG
GG	GC	GC
TG	GT	GG
AA	AA	TT
GA	TT	AA
GC	TG	GT

1	2	3	4	5	6
GA, 4	CG, 7	GT, 13	AG, 0	AA, 10	GA, 9
TG, 3	TG, 14	GA, 4	TT, 2	GT, 13	GG, 8
AG, 5	AG, 0	GA, 9	AG, 11	GA, 9	GC, 6
GT, 1	GA, 9	TG, 3	AG, 5	GT, 1	TG, 14
GT, 13	AG, 5	AG, 5	AA, 10	AG, 5	GT, 13
AA, 10	AG, 11	CG, 7	GT, 13	TT, 2	TT, 2
AG, 11	GA, 4	TT, 2	CG, 7	GA, 4	AA, 10
TT, 2	GT, 13	AA, 10	GG, 8	CG, 7	AG, 0
AG, 0	TT, 2	GG, 12	GA, 4	AG, 0	CG, 7
CG, 7	TG, 3	GG, 8	GA, 9	TG, 3	GG, 12
GG, 12	GG, 8	TG, 14	TG, 14	GG, 8	AG, 11
GC, 6	AA, 10	GT, 1	TG, 3	GG, 12	TG, 3
TG, 14	GG, 12	AG, 11	GC, 6	GC, 6	GT, 1
GG, 8	GT, 1	GC, 6	GT, 1	AG, 11	GA, 4
GA, 9	GC, 6	AG, 0	GG, 12	TG, 14	AG, 5

GC GA | AG GG | AG TG

minHash & OMH sketches

$$S = \texttt{AGTTGAGCGGAAGGTG},\ k = 2,\ L = 3,\ \ell = 2$$

Jaccard:

$$\mathrm{Sk}(S) = \begin{pmatrix} \texttt{GC} \\ \texttt{TG} \\ \texttt{GT} \end{pmatrix}$$

OMH:

$$\mathrm{Sk}(S) = \begin{pmatrix} \texttt{GC} & \texttt{CA} \\ \texttt{AG} & \texttt{GG} \\ \texttt{AG} & \texttt{TG} \end{pmatrix}$$

OMH is a LSH for edit distance

Theorem: OMH is a LSH for edit distance

There exists (d_1, d_2, p_1, p_2) such that OMH is sensitive for the edit distance.

- p_1: related to probability of hash collisions of weighted Jaccard
- p_2: related to length of increasing sequence given weighted Jaccard

Practical considerations with Jaccard sketches

Jaccard:

- Can use canonical k-mers
- Difficult to find independent hashes: use bottom sketches ($L \ll n$)

OMH:

- ℓ times as large (cost to encode order)
- $\ell = 1$: LSH / unbiased estimator of weighted Jaccard
- Can't use canonical k-mers: double sketch

OMH has a large gap

- $|S| = 100, k = 5$
- Current proof has a large gap
- What is smallest gap possible?
- OMH/minHash similar to embedding in Hamming space: gap probably unavoidable