UNIVERSITÄT DES SAARLANDES

ZBI ZENTRUM FÜR BIOINFORMATIK

Min-Hashing and Applications

Algorithms for Sequence Analysis

Sven Rahmann
partially based on slides by Ali Ghaffaari

Summer 2021

Overview

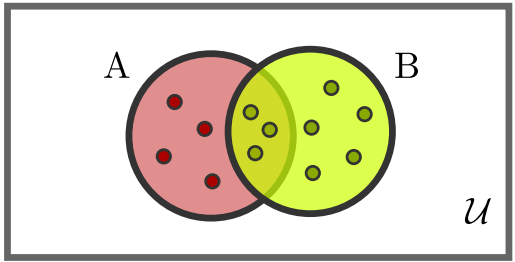
Previous lecture

- Hashing, collisions
- (h, b) Cuckoo hashing
- Locality sensitive hashing
- Min-hashing: Locality sensitive for Jaccard similarity of k-mer sets

Today's lecture

- Details on min-hashing of DNA k-mers
- Applications

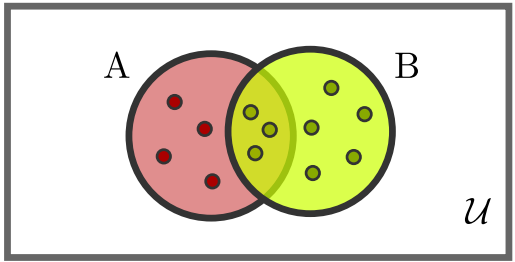
LSH for Jaccard Similarity



$$\mathcal{S}_J = \frac{|A \cap B|}{|A \cup B|}$$

$$\mathcal{S}_J(A, B) = \frac{3}{12} = 0.25$$

LSH for Jaccard Similarity



$$\mathcal{S}_J = \frac{|A \cap B|}{|A \cup B|}$$

$$\mathcal{S}_J(A, B) = \tfrac{3}{12} = 0.25$$

Claim: Min-Hashing is LS for Jaccard Similarity

A bijective function $\pi : \mathcal{U} \to [0, |\mathcal{U}|[$ is a ranking (ordering) function of $\mathcal{U}$.
The family $\mathcal{H}$ of hash functions

$$h_\pi(A) := \min_{x \in A} \pi(x)\,,$$

where π ranges over **all orderings** of $\mathcal{U}$, is **locality sensitive** for $\mathcal{S}_J$.

Proof: Min-Hashing is LS for Jaccard Similarity

Definitions:

- $\mathcal{S}_J(A, B) = \frac{|A \cap B|}{|A \cup B|}$
- $h_\pi(A) := \min_{x \in A} \pi(x)$
- Let $a := h_\pi(A)$ and $b := h_\pi(B)$.

So what is $\mathbf{P}[a = b]$?

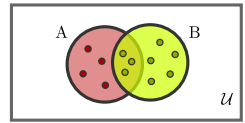
Proof: Min-Hashing is LS for Jaccard Similarity

Definitions:

- $\mathcal{S}_J(A,B) = \frac{|A \cap B|}{|A \cup B|}$
- $h_\pi(A) := \min_{x \in A} \pi(x)$
- Let $a := h_\pi(A)$ and $b := h_\pi(B)$.

So what is $\mathbf{P}[a = b]$?

- $a = b$ iff minimum over elements in $A \cup B$ is in $A \cap B$.
- $|A \cap B|$ successes out of $|A \cup B|$ possible events
- Thus, $\mathbf{P}[a = b] = |A \cap B| / |A \cup B| = \mathcal{S}_J(A,B)$.



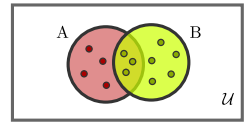
Proof: Min-Hashing is LS for Jaccard Similarity

Definitions:

- $\mathcal{S}_J(A,B) = \frac{|A \cap B|}{|A \cup B|}$
- $h_\pi(A) := \min_{x \in A} \pi(x)$
- Let $a := h_\pi(A)$ and $b := h_\pi(B)$.

So what is $\mathbf{P}[a = b]$?

- $a = b$ iff minimum over elements in $A \cup B$ is in $A \cap B$.
- $|A \cap B|$ successes out of $|A \cup B|$ possible events
- Thus, $\mathbf{P}[a = b] = |A \cap B| / |A \cup B| = \mathcal{S}_J(A,B)$.



Assumptions (min-hashing still useful if weakened)

- Elements of A, B (k-mers) are bijectively encoded, not hashed.
- Truly random permutations are used.

Sketches: Min-Hashing

Definition: Min-hashing Sketch

A **sketch** or **signature** for the **Jaccard similarity** of the form

$$h_i(A) := \min_{x \in A} \pi_i(x), \qquad i = 1, \ldots, r\,,$$

where each π_i is an independent random permutation of $\mathcal{U}$, is a **min-hashing sketch**.

Sketches: Min-Hashing

Definition: Min-hashing Sketch

A **sketch** or **signature** for the **Jaccard similarity** of the form

$$h_i(A) := \min_{x \in A} \pi_i(x), \qquad i = 1, \ldots, r\,,$$

where each π_i is an independent random permutation of $\mathcal{U}$, is a **min-hashing sketch**.

Weaker versions in practice

1. Elements of A, B are not integer-encoded, but hashed:
   Additional collisions, higher apparent similarity

Sketches: Min-Hashing

Definition: Min-hashing Sketch

A **sketch** or **signature** for the **Jaccard similarity** of the form

$$h_i(A) := \min_{x \in A} \pi_i(x), \qquad i = 1, \ldots, r\,,$$

where each π_i is an independent random permutation of $\mathcal{U}$, is a **min-hashing sketch**.

Weaker versions in practice

1. Elements of A, B are not integer-encoded, but hashed:
   Additional collisions, higher apparent similarity
2. Permutations are chosen from a limited set, not perfectly random,
   e.g. $\pi(x) = (a \cdot (x \oplus b)) \bmod 4^k$ with odd a, some b.

Sketches: Min-Hashing

Definition: Min-hashing Sketch

A **sketch** or **signature** for the **Jaccard similarity** of the form

$$h_i(A) := \min_{x \in A} \pi_i(x), \qquad i = 1, \ldots, r\,,$$

where each π_i is an independent random permutation of $\mathcal{U}$, is a **min-hashing sketch**.

Weaker versions in practice

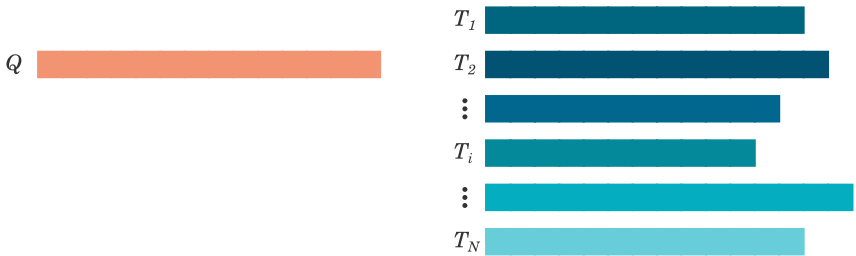
1. Elements of A, B are not integer-encoded, but hashed:
   Additional collisions, higher apparent similarity
2. Permutations are chosen from a limited set, not perfectly random,
   e.g. $\pi(x) = (a \cdot (x \oplus b)) \bmod 4^k$ with odd a, some b.
3. Computing r hash values is expensive; can one suffice?
   - Take r smallest values of one h instead of minima from r functions.
   - Partition universe into r subsets, take minimum in each subset separately.

Querying String Sets by Similarity

Sequence Similarity Search

Given a set of texts $\mathcal{T} = \{T_1, T_2, ..., T_N\}$ and a query sequence Q,
find all texts in $\mathcal{T}$ that are (locally) similar to Q (above a threshold).

Most sequence similarity search algorithms use seed-and-extend approach.

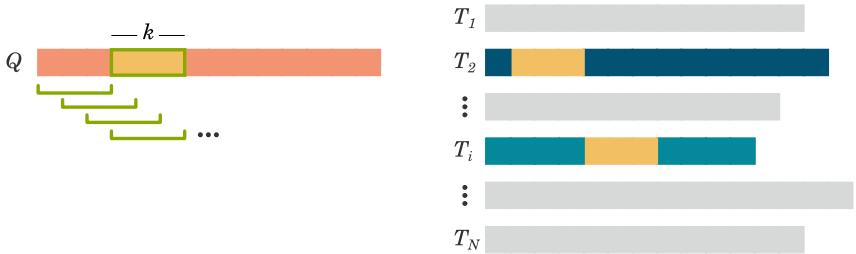


Querying String Sets by Similarity

Sequence Similarity Search

Given a set of texts $\mathcal{T} = \{T_1, T_2, ..., T_N\}$ and a query sequence Q,
find all texts in $\mathcal{T}$ that are (locally) similar to Q (above a threshold).

Most sequence similarity search algorithms use seed-and-extend approach.



Querying String Sets by Similarity

Sequence Similarity Search

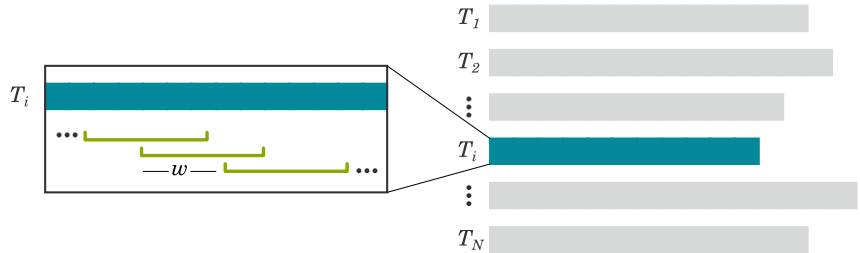
Given a set of texts $\mathcal{T} = \{T_1, T_2, ..., T_N\}$ and a query sequence Q,
find all texts in $\mathcal{T}$ that are (locally) similar to Q (above a threshold).

Most sequence similarity search algorithms use seed-and-extend approach.

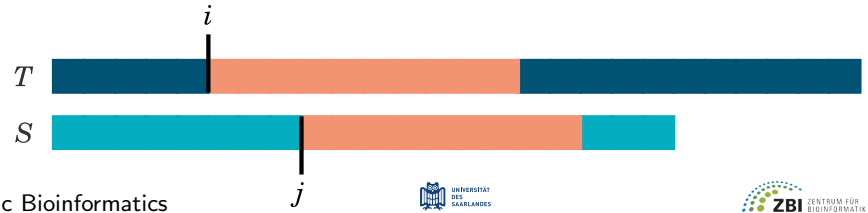
- Many methods catalogue all k-mers in the database: ***k*-mer index**.
- **Goal:** Use only a subset of k-mers: k-mer **sampling**.

K-mer Sampling: How **NOT** to do it

Bad idea: Only consider every w-th k-mer in a string



If $|j - i|$ is not a multiple of w, then substrings at i and j do not share any k-mers:



K-mer Sampling with Minimizers

A **minimizer scheme** is a much better approach to sample k-mers.

- Fix an ordering (permutation) of all k-mers: π.
- Consider a **window** of w consecutive k-mers.
- Choose the/a k-mer x^* such that $\pi(x^*)$ is minimum among all $\pi(x)$ in the window.
- Such an x^* is called a (w, k)-**minimizer**.
- Sliding the w-window over the text, we collect all such minimizers.

Example:

Properties of Minimizers

- If two strings have a **sufficiently long exact match** (length $w + k - 1$), then they are **guaranteed to share a (w, k) minimizer**
- Even without an exact match of length $w + k - 1$, **similar strings** (Jaccard similarity of k-mer sets) share a minimizer with **high probability**.
- Only a small fraction of all k-mers need to be stored: For a random string this fraction is about $2/(w + 1)$ on average, i.e., minimizers do not change frequently.
- Larger w: Smaller sample, but requires higher similarity for guarantees. Also slightly higher probability of "random hits".

Practical Considerations

Long vs. short k-mers on DNA

- $k \leq 32$: Encode bijectively
- $k > 32$: Hash k-mers to 64-bit integers (additional collisions possible)

Practical Considerations

Long vs. short k-mers on DNA

- $k \leq 32$: Encode bijectively
- $k > 32$: Hash k-mers to 64-bit integers (additional collisions possible)

Random permutations?

- For 4^k objects, there are $(4^k)!$ possible orderings (permutations).
  For $k = 3$: $64! = 126886932185884164103433389335161480802865516174545192198801894375214704230400000000000000$
- Impossible to pick one truly randomly with a pseudo-random number generator: Restrict to much smaller sets in practice,
- Two-parameter version: $\pi_{a,b}(x) = (a \cdot (x \oplus b)) \bmod 4^k$ with odd a, any b.
  Randomly choose b and odd a only.

Practical Considerations

Canonical k-mers vs. both strands

- DNA sequence is equivalent to its reverse complement: AAAG = CTTT.
- Store or sample both strands (twice the size) ?
- Alternative: Use canonical k-mers (encodings, hash values):
  Among x and its reverse complement $\bar{x}$, pick the smaller (or larger) one.
- When using min-hashing, it may be better to use $\max\{x, \bar{x}\}$.

Practical Considerations

Canonical k-mers vs. both strands

- DNA sequence is equivalent to its reverse complement: AAAG = CTTT.
- Store or sample both strands (twice the size) ?
- Alternative: Use canonical k-mers (encodings, hash values):
  Among x and its reverse complement $\bar{x}$, pick the smaller (or larger) one.
- When using min-hashing, it may be better to use $\max\{x, \bar{x}\}$.

Gapped k-mers

- If error rates or evolutionary distances are moderately high,
  a few equidistant differences may destroy all common k-mers.
- Can use gaped k-mers (masks like `#.##...##.#`) instead.
- Can use different masks together with different permutations in sketches.
- Possibilities are endless... Interesting research topics!

Applications

Minimap2 (Read mapping using seed-and-extend with minimizers)

- map noisy long reads to genomes or assemblies
- Li, H. (2018). Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics, 34:3094-3100. `https://github.com/lh3/minimap2`

Applications

Minimap2 (Read mapping using seed-and-extend with minimizers)

- map noisy long reads to genomes or assemblies
- Li, H. (2018). Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics, 34:3094-3100. `https://github.com/lh3/minimap2`

Sourmash v4 (Similarity Estimation)

- computes hash sketches from DNA sequences, compares them, estimates sequence similarity between large datasets quickly and accurately.
- Brown and Irber (2016). sourmash: a library for MinHash sketching of DNA. Journal of Open Source Software, 1(5), 27

Applications

Minimap2 (Read mapping using seed-and-extend with minimizers)

- map noisy long reads to genomes or assemblies
- Li, H. (2018). Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics, 34:3094-3100. `https://github.com/lh3/minimap2`

Sourmash v4 (Similarity Estimation)

- computes hash sketches from DNA sequences, compares them, estimates sequence similarity between large datasets quickly and accurately.
- Brown and Irber (2016). sourmash: a library for MinHash sketching of DNA. Journal of Open Source Software, 1(5), 27

Kraken2 (Metagenomics)

- Finds species of origin for each read, estimates species abundance.
- Wood, D.E., Lu, J. & Langmead, B. (2019). Improved metagenomic analysis with Kraken 2. Genome Biol 20, 257.

Applications

Minimap2 (Read mapping using seed-and-extend with minimizers)

- map noisy long reads to genomes or assemblies
- Li, H. (2018). Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics, 34:3094-3100. `https://github.com/lh3/minimap2`

Sourmash v4 (Similarity Estimation)

- computes hash sketches from DNA sequences, compares them, estimates sequence similarity between large datasets quickly and accurately.
- Brown and Irber (2016). sourmash: a library for MinHash sketching of DNA. Journal of Open Source Software, 1(5), 27

Kraken2 (Metagenomics)

- Finds species of origin for each read, estimates species abundance.
- Wood, D.E., Lu, J. & Langmead, B. (2019). Improved metagenomic analysis with Kraken 2. Genome Biol 20, 257.

Xengsort (Xenograft sorting, cancer research)

- Split reads of xenograft samples into several categories
- Zentgraf % Rahmann (2021). Fast lightweight xenograft sorting. Algorithms for Molecular Biology 16:2.

Summary

- Locality sensitive hashing for Jaccard similarity: Min-hashing
- Sketches and alternative implementations using a single hash function
- Sampling DNA sequences by using k-mer minimizers:
  1. reduction of size
  2. built-in error tolerance
- Technical details to consider
- Alignment-free methods based on k-mers
- Applications

Possible Exam Questions

- Prove that min-hashing is LS for Jaccard similarity
- What is a sketch?
- Why are sketches useful for similarity search in high-dimensional spaces?
- What are minimizers (precisely, (w, k)-minimizers) of a sequence?
- What property does the set of minimizers of a sequence have?
- What is the effect of changing the window size w?
- What is the effect of changing the k-mer size k?
- Name some application areas of (w, k)-minimizers in bioinformatics.