Document duplication
(exact or approximate)

Duplicate documents

- The web is full of duplicated content
  - Few **exact duplicate** documents
  - Many cases of **near duplicates** docs
    - E.g., Last modified date
    - Malicious
    - Spam
    - …

Exact-Duplicate Detection

- **Obvious (slow) techniques**
  - Checksum – no worst-case collision probability guarantees
  - MD5 – cryptographically-secure string hashes
- **Karp-Rabin's (fast) scheme**
  - Rolling hash: split doc in many pieces
  - Algebraic technique: arithmetic on primes
  - Efficient and other nice properties…

Karp-Rabin Fingerprints

- Consider – m-bit string $A = 1\ a_1\ a_2\ \ldots\ a_m$
- Basic values:
  - Choose a prime p in the universe U, such that 2p uses few memory-words (hence $U \approx 2^{64}$)
- Fingerprints: $f(A) = A \bmod p$
  - Nice property is that if $B = 1\ a_2\ \ldots\ a_m\ a_{m+1}$
  - $f(B) = [2 * (A - 2^m - a_1\ 2^{m-1}) + 2^m + a_{m+1}\ ] \bmod p$
- Prob[false hit btw A vs B] = Prob p divides (A-B) = #div(A-B)/ #prime(U)

  $\approx$ (log (A+B)) / #prime(U)

  = m log U/U

Near-Duplicate Detection

- **Problem**
  - Given a large collection of documents
  - Identify the near-duplicate documents

- **Web search engines**
  - Proliferation of near-duplicate documents
    - Legitimate – mirrors, local copies, updates, …
    - Malicious – spam, spider-traps, dynamic URLs, …
    - Mistaken – spider errors
  - 30% of web-pages are near-duplicates [1997]

Shingling: from docs to sets of shingles

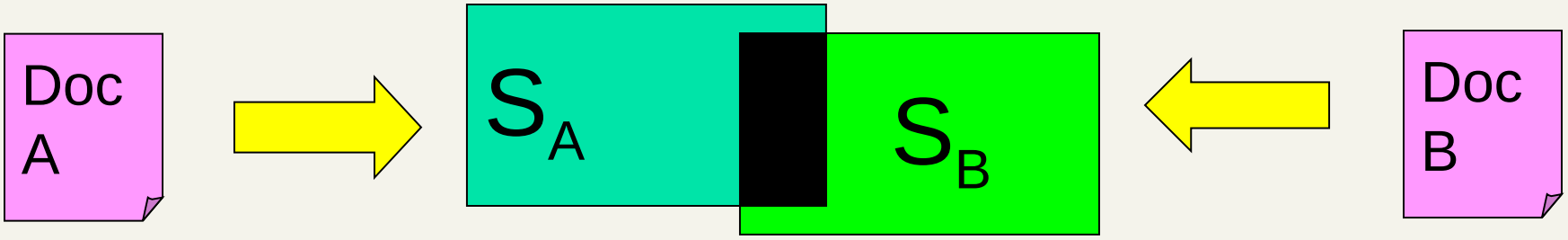
- dissect document into q-grams (shingles)

T = I live and study in Pisa, ...

If we set q=3 the 3-grams are:

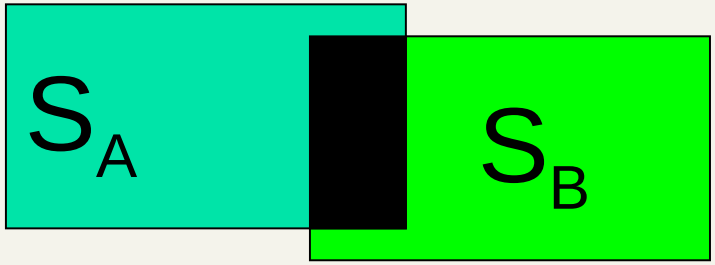
<I live and><live and study><and study in><study in Pisa>...

- represent documents by sets of hashes/shingles



The near-duplicate document detection problem reduces to set intersection among integers (shingles)

More precise: Jaccard similarity



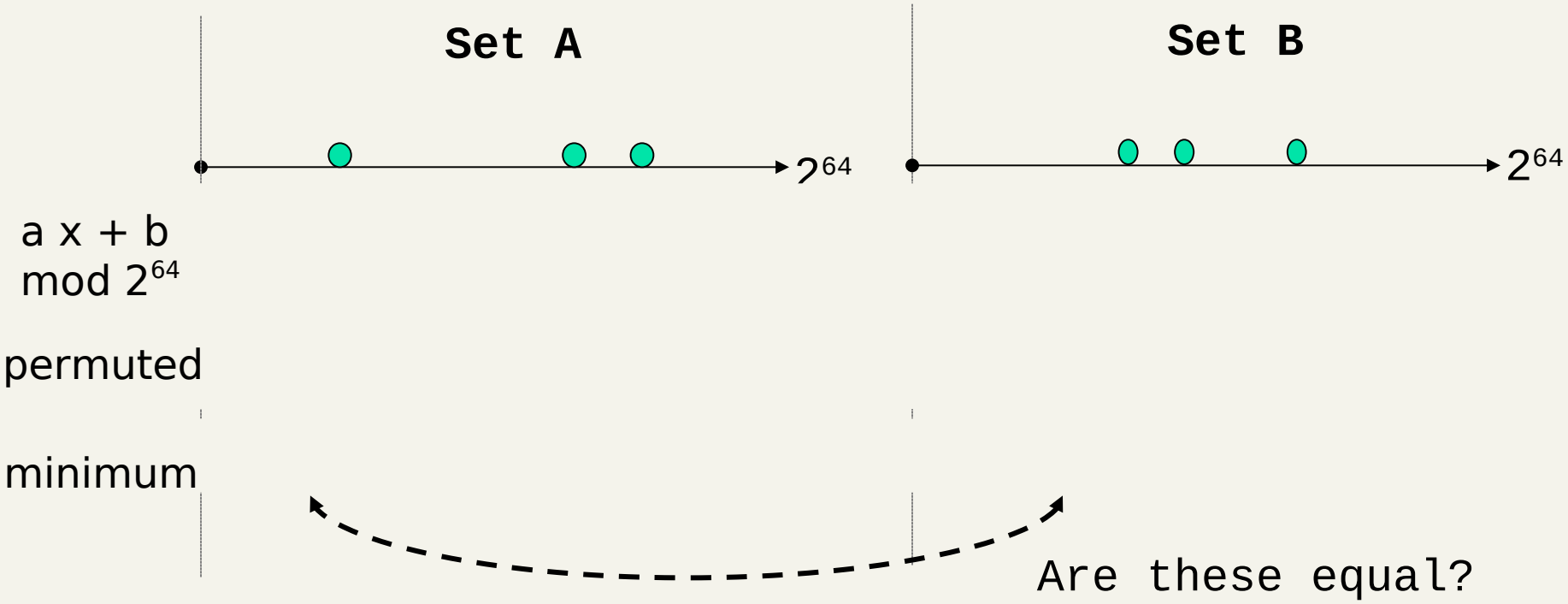
$$\text{sim}(S_A, S_B) = \frac{|S_A \cap S_B|}{|S_A \cup S_B|}$$

Set similarity → *Jaccard* similarity

Desiderata

- **Storage:** only small **sketches** of each document.
- **Computation:** the fastest possible
- **Stream Processing:**
  - once sketch computed, source is unavailable
- **Error Guarantees**
  - problem scale → small biases have large impact
  - need formal guarantees – heuristics will not do

Min-Hashing to estimate Jaccard-sim(S_A, S_B)



Lemma: Prob[α=β] is exactly the Jaccard-sim(S_A, S_B)

Use 200 random permutations (*minimum*), or pick the 200 smallest items from one random permutation, thus create one **200-dim vector per set** and evaluate **Hamming distance**

$\cos(\alpha) = p \bullet q / ||p|| * ||q||$

Cosine distance btw *p* and *q*

Construct a random hyperplane **r** of **d**-dim and unit norm

- Sketch of a vector p is $h_r(p)=\text{sign}(p \bullet r) = \pm 1$
- Sketch of a vector q is $h_r(q)=\text{sign}(q \bullet r) = \pm 1$

- **Lemma:**

$$P[h_r(p) = h_r(q)] = 1 - \frac{\alpha}{\pi}$$

Other distances

LSH Algorithm and Implementation (E2LSH)

Locality-Sensitive Hashing (LSH) is an algorithm for solving the approximate or exact Near Neighbor Search in high dimensional spaces. This webpage links to the newest LSH algorithms in Euclidean and Hamming spaces, as well as the **E2LSH package**, an implementation of an early practical LSH algorithm.

- **Algorithm description:**
  - **Newest (not quite) LSH algorithms (2014):** These algorithms achieve performance better than the classic LSH algorithms by using *data-dependent* hashing. They improve over classic LSH algorithms for both Hamming and Euclidean space. These algorithms are not dynamic however, in contrast to the classic LSH algorithms, which use *data-independent* hashing and hence allow updates to the pointset.

    Optimal Data-Dependent Hashing for Approximate Near Neighbors (by Alexandr Andoni and Ilya Razenshteyn). In STOC'15 (to appear). Full version in arXiv:1501.01062.

    Beyond Locality Sensitive Hashing (by Alexandr Andoni, Piotr Indyk, Huy L. Nguyen, and Ilya Razenshteyn). In *SODA'14*.
    Slides: Here are some slides by Alexandr Andoni on the early version from SODA'14.

  - **Survey of LSH in CACM (2008):** "Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions" (by Alexandr Andoni and Piotr Indyk). *Communications of the ACM*, vol. 51, no. 1, 2008, pp. 117-122. (CACM disclaimer).
    also available directly from CACM (for free).