DOCUMENT DEDUPLICATION

The webs is full of copies of the same contents, and we would like to avoid the same things more than once. The bloom filter is not enough to avoid duplication, because it may returns false positive, so we need other deterministic techniques to double-check

Types of Duplicate → Exact duplicated documents

Types of Duplicate → Near duplicated documents

Technique to identify exact duplicates

- Checksum
- MD5

SLOW BUT SECURE

- **Karp-Rabin**

FAST, BASED ON BOUNDED PROB.

Given a prime number p, the fingerprint of an m bit string A is calculated as $f(A) = A \bmod p$. The probability of a fingerprint collision between any A, B is equal to the probability that p divides A - B

$$P(f(A) == f(B)) = P(p \text{ divides } (A-B)) =$$

$$= \frac{\#\text{ divisori of } (A-B)}{\#\text{ prime in a set } U} =$$

$$\simeq \frac{\log(A+B)}{\#\text{ primes in a set } U} =$$

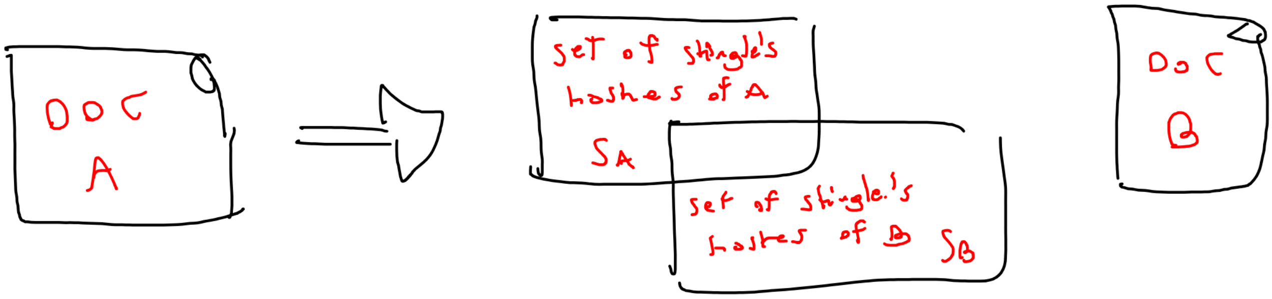
$$= m \cdot \frac{\log U}{U}$$

very low!

Techinquer to identify near-duplicates

Shingling

Dissect the document in q-grams (or shingles) and generates their fingerprints with a hash function. Therefore each document is represented by a set of shingles and hashes, and the near-duplicate document detection problem is reduced to the set similarity problem, using the hashes



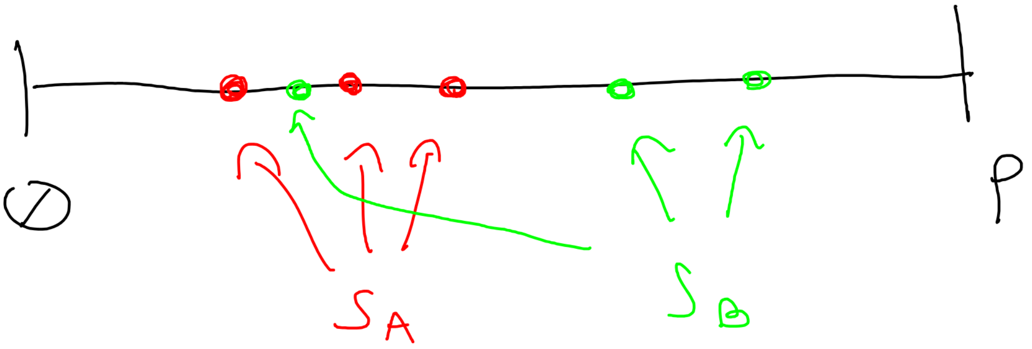
The set similarity is often computed using the Jaccard similarity

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

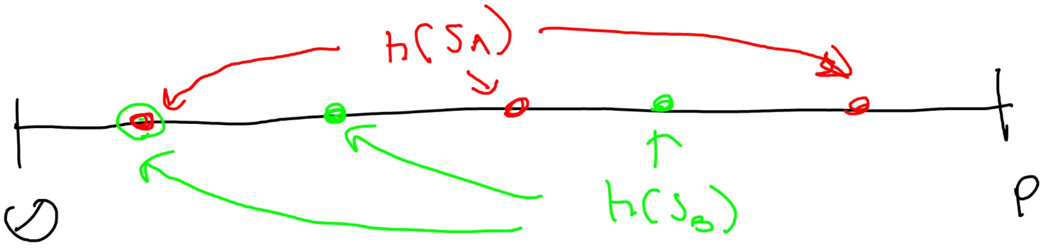
However it requires a big amount of space and the full cost of the set intersection.

The Jaccard similarity can be approximated using the Min-Hashing technique

1) Since S_A and S_B are sets of integers, each value will go from 0 to p (assuming that the hash function used is something like $h(x) = x \bmod p$)



2) Define coprimi $h(x) = (\underline{a}x+b) \bmod \underline{p}$ and computes the new sets $h(S_A)$ and $h(S_B)$. The resulting sets will be again in the range $[0, p)$



3) Takes the minimum values of $h(S_A)$ and $h(S_B)$: if they are equal, it means that this value has been taken from the intersection $S_A \cap S_B$

$$P(\min(h(S_A)) == \min(h(S_B))) = J(S_A, S_B)$$

By executing L times step 2) and step 3),

The approximation becomes more presise,

because we just have to count how many times the minimum are equals and divide by L (this way we normalize the result and get the expected value in $[0,1]$)

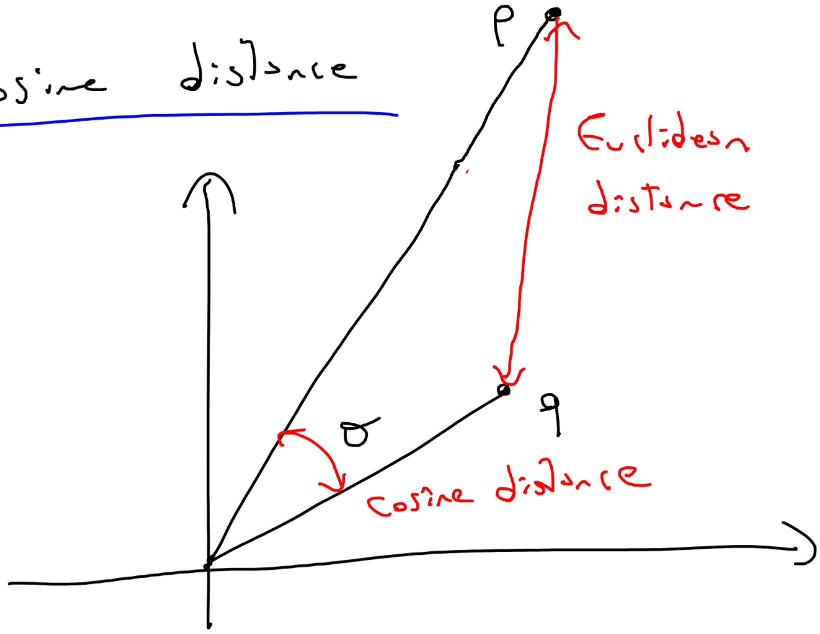
Indeed:

$$\frac{E(\#\text{ equal components})}{L} =$$

$$= \frac{L \cdot P(\min(h(S_A)) == \min(h(S_B)))}{L} =$$

$$= J(S_A, S_B)$$

- Cosine distance



(NOT A GOOD METRIC FOR SIMILARITY)

$$\cos(\theta) = \frac{p \cdot q}{||p|| \cdot ||q||}$$

Construct a random hyperplane r of d dimensions and unit norm

$$h_r(x) = \text{sign}(x \cdot r) = \pm 1$$

where x is a vector

$$P\left(h_r(p) == h_r(q)\right) = 1 - \frac{\alpha}{r}$$