CS 172 INFORMATION RETRIEVAL

Duplicate Detection

Detecting Duplicates

- Duplicate and near-duplicate documents occur in many situations
  - Copies, versions, plagiarism, spam, mirror sites
  - 30% of the web pages in a large crawl are exact or near duplicates of pages in the other 70%
- Duplicates consume significant resources during crawling, indexing, and search
  - Little value to most users
- Exact duplicate
  - The content is identical bit by bit
- Near duplicate
  - Very similar but not exact

Duplicate Detection

- Exact duplicate detection is relatively easy

- Naïve Approach
  - Compare every document against all others
  - O(n^2) approach

- Better Approach
  - Create hash of each document and check if two documents have the same hash code
  - O(n) approach

Duplicate Detection

- Checksum techniques
  - A checksum is a value that is computed based on the content of the document
    - e.g., sum of the bytes in the document file
  - However, its possible for files with different text to have same checksum

- Functions such as a *cyclic redundancy check (CRC),* have been developed that consider the positions of the bytes
  - Popularly used in networks for detecting errors in data transmission.
  - Maintains a certain number of check bits (checksum) which are appended to the message

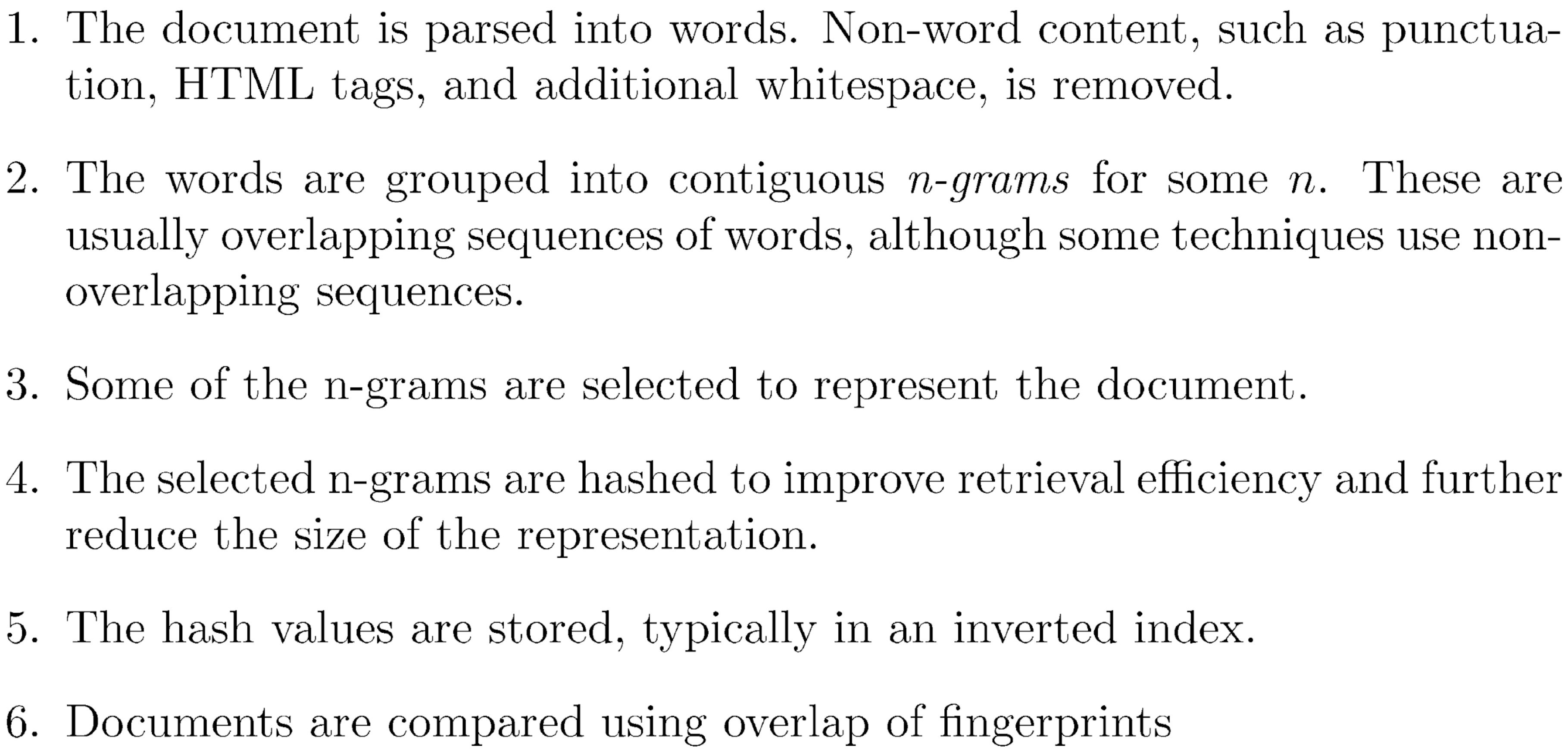
T	r	o	p	i	c	a	l		f	i	s	h	*Sum*
54	72	6F	70	69	63	61	6C	20	66	69	73	68	508

Near-Duplicate Detection

- More challenging task
  - Are web pages with same text context but different advertising or format near-duplicates?

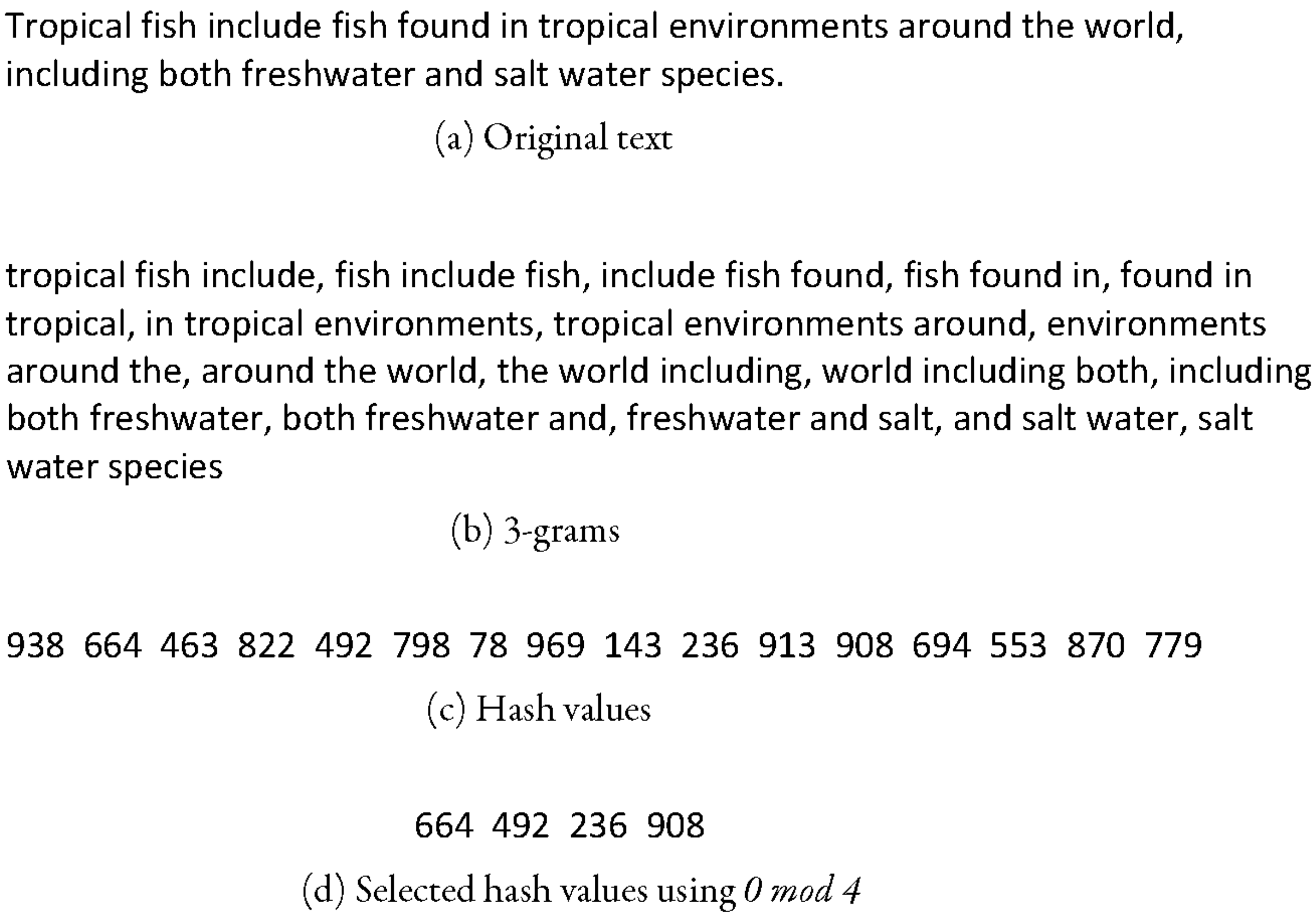
- A near-duplicate document is defined using a threshold value for some similarity measure between pairs of documents
  - e.g., document D1 is a near-duplicate of document D2 if more than 90% of the words in the documents are the same

Fingerprints



1. The document is parsed into words. Non-word content, such as punctuation, HTML tags, and additional whitespace, is removed.
2. The words are grouped into contiguous *n-grams* for some n. These are usually overlapping sequences of words, although some techniques use non-overlapping sequences.
3. Some of the n-grams are selected to represent the document.
4. The selected n-grams are hashed to improve retrieval efficiency and further reduce the size of the representation.
5. The hash values are stored, typically in an inverted index.
6. Documents are compared using overlap of fingerprints

Fingerprint Example



Tropical fish include fish found in tropical environments around the world, including both freshwater and salt water species.

(a) Original text

tropical fish include, fish include fish, include fish found, fish found in, found in tropical, in tropical environments, tropical environments around, environments around the, around the world, the world including, world including both, including both freshwater, both freshwater and, freshwater and salt, and salt water, salt water species

(b) 3-grams

938 664 463 822 492 798 78 969 143 236 913 908 694 553 870 779

(c) Hash values

664 492 236 908

(d) Selected hash values using *0 mod 4*

Computing Similarity

- Features:
  - Segments of a document (natural or artificial breakpoints)
  - Shingles (Word N-Grams)
  - ***a rose is a rose is a rose*** → 4-grams are

    a_rose_is_a

    rose_is_a_rose

    is_a_rose_is

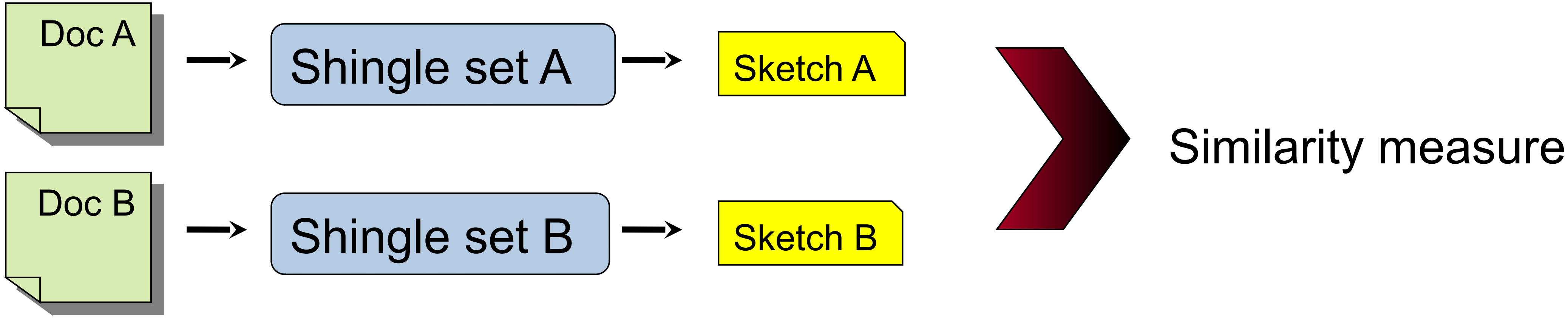
    a_rose_is_a

- Similarity Measure between two docs (= sets of shingles)
  - Set intersection
  - Specifically (Size_of_Intersection / Size_of_Union)

Shingles/nGrams/Features + Set Intersection

- Computing exact set intersection of shingles between all pairs of documents is expensive/intractable
  - Approximate using a cleverly chosen subset of shingles from each (a *sketch*)

- Estimate (size_of_intersection / size_of_union) based on a short sketch



What is a similarity measure?

- Numerical measure of how alike two data objects are.
  - A function that maps pairs of objects to real values
  - Higher when objects are more alike.
- Often falls in the range [0,1], sometimes in [-1,1]

- Desirable properties for similarity
  1. s(p, q) = 1 (or maximum similarity) only if p = q. (Identity)
  2. s(p, q) = s(q, p) for all p and q. (Symmetry)

Similarity between sets

- Consider the following documents

apple releases new ipod

D1

apple releases new ipad

D2

new apple pie recipe

D3

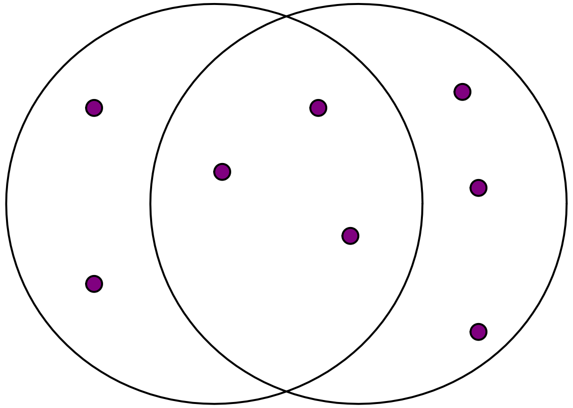
- Which ones are more similar?
- How would you quantify their similarity?

Use number of words in common …

Sim(D2,D1) = 3, Sim(D1,D3) = Sim(D2,D3) =2

Jaccard Similarity

- The Jaccard similarity (Jaccard coefficient) of two sets S_1, S_2 is the size of their intersection divided by the size of their union.
  - JSim $(C_1, C_2) = |C_1 \cap C_2| / |C_1 \cup C_2|$.



3 in intersection.
8 in union.
Jaccard similarity
= 3/8

- Extreme behavior:
  - Jsim(X,Y) = 1, iff X = Y
  - Jsim(X,Y) = 0 iff X,Y have no elements in common
- JSim is symmetric

Jaccard Similarity between sets

- The distance for the documents

apple releases new ipod

D1

apple releases new ipad

D2

new apple pie recipe

D3

Vera releases new book with apple pie recipes

D4

- JSim(D2,D1) = 3/5
- JSim(D1,D3) = JSim(D2,D4) = 2/6
- JSim(D1,D4) = JSim(D2,D3) = 3/9

- Whats the difference between Jaccard and Cosine similarity?

Cosine Similarity

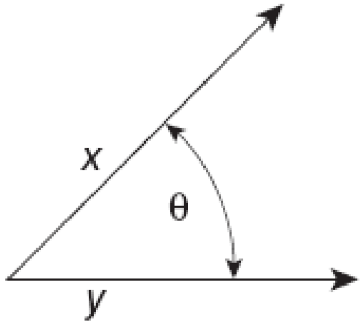


Figure 2.16. Geometric illustration of the cosine measure.

- Sim(X,Y) = cos(X,Y)
  - The cosine of the angle between X and Y

- If the vectors are aligned (correlated) angle is zero degrees and cos(X,Y)=1
- If the vectors are orthogonal (no common coordinates) angle is 90 degrees and cos(X,Y) = 0

- Cosine is commonly used for comparing documents, where we assume that the vectors are normalized by the document length.

Cosine Similarity - math

- If d_1 and d_2 are two vectors, then

  $\cos(d_1, d_2) = (d_1 \cdot d_2) / ||d_1||\ ||d_2||$,

  where · indicates vector dot product and || *d* || is the length of vector *d*.

- Example:

  $d_1 =$ **3 2 0 5 0 0 0 2 0 0**

  $d_2 =$ **1 0 0 0 0 0 0 1 0 2**

  $d_1 \cdot d_2 = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5$

  $||d_1|| = (3*3+2*2+0*0+5*5+0*0+0*0+0*0+2*2+0*0+0*0)^{0.5} = (42)^{0.5} = 6.481$

  $||d_2|| = (1*1+0*0+0*0+0*0+0*0+0*0+0*0+1*1+0*0+2*2)^{0.5} = (6)^{0.5} = 2.245$

  $\cos(d_1, d_2) = .3150$

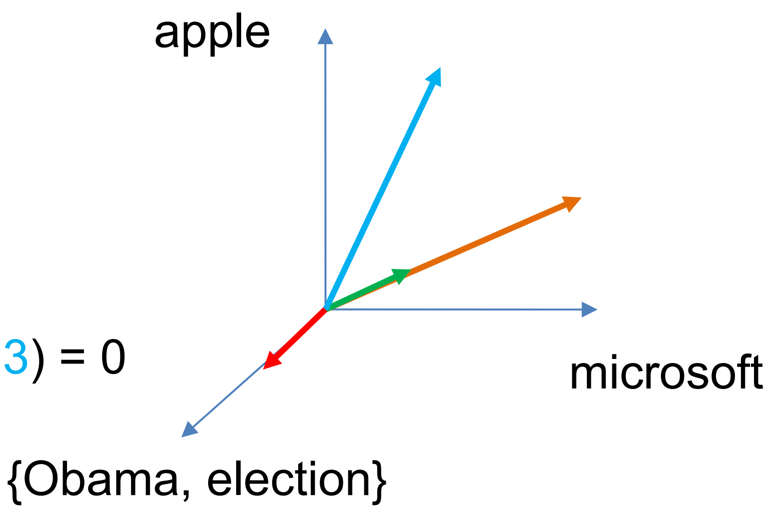
Example

document	Apple	Microsoft	Obama	Election
D1	10	20	0	0
D2	30	60	0	0
D3	60	30	0	0
D4	0	0	10	20

Cos(D1,D2) = 1

Cos (D3,D1) = Cos(D3,D2) = 4/5

Cos(D4,D1) = Cos(D4,D2) = Cos(D4,D3) = 0



Simhash

- Similarity comparisons using word-based representations more effective at finding near-duplicates
  - Problem is efficiency

- Simhash combines the advantages of the word-based similarity measures with the efficiency of fingerprints based on hashing

- Similarity of two pages as measured by the cosine correlation measure is proportional to the number of bits that are the same in the simhash fingerprints

Simhash

- Basic Idea
  - Tokenize, remove-words, (sometimes stem)
  - assign weights to words (frequencies or TF-IDF)
  - Compute a unique b-bit binary hash for every word
  - Convert 0 to -1 and multiple by word weight
  - Add by columns, set to 1 if the sum is > 0, and 0 otherwise.

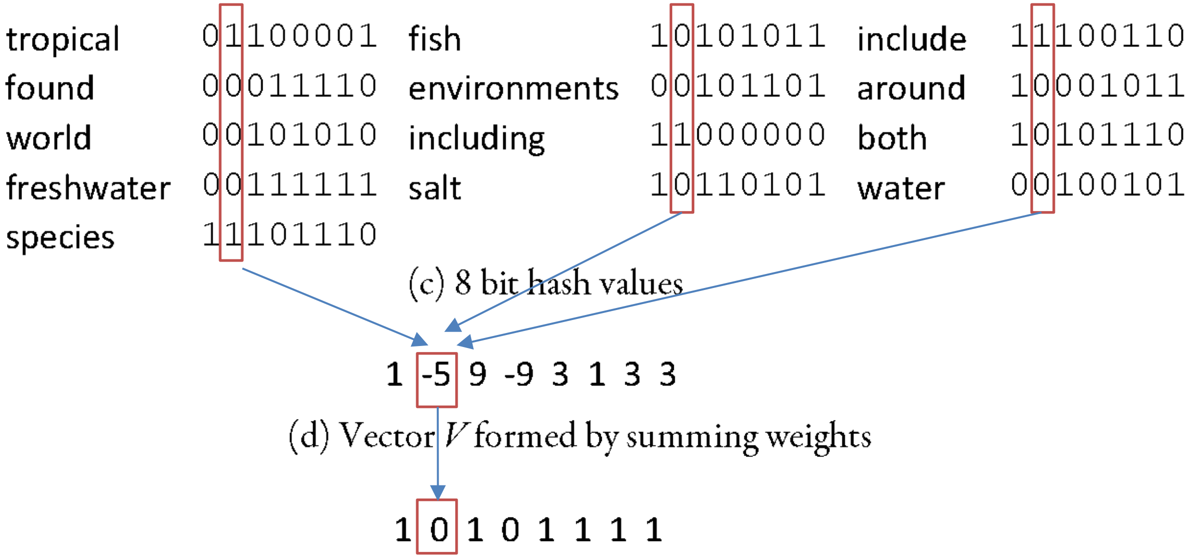
Simhash Example

Tropical fish include fish found in tropical environments around the world, including both freshwater and salt water species.

(a) Original text

tropical 2 fish 2 include 1 found 1 environments 1 around 1 world 1
including 1 both 1 freshwater 1 salt 1 water 1 species 1

(b) Words with weights



tropical	01100001	fish	10101011	include	11100110
found	00011110	environments	00101101	around	10001011
world	00101010	including	11000000	both	10101110
freshwater	00111111	salt	10110101	water	00100101
species	11101110				

(c) 8 bit hash values

1 -5 9 -9 3 1 3 3

(d) Vector *V* formed by summing weights

1 0 1 0 1 1 1 1

(e) 8-bit fingerprint formed from *V*