Document Similarity and De-duplication

Distinct URLs May Deliver the Same Page

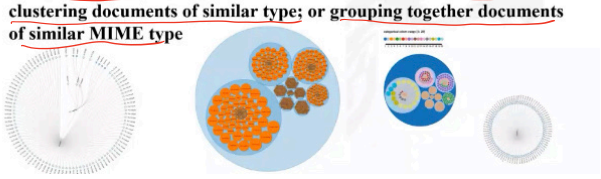
- The 3 URLs below all point to the identical page
- Structural Classification of Proteins
  - http://scop.mrc-lmb.cam.ac.uk/scop
  - http://scop.berkeley.edu/
  - http://scop.protres.ru/

The Definition of De-Duplication

- *De-duplication* – the process of identifying and avoiding essentially identical web pages/files

The term is often used in connection with *cloud computing* where only a single copy of a file is stored and multiple links to the single file are managed

With respect to *content detection and analysis*, de-duplication essentially refers to the identification of identical and nearly identical documents and indexing only a single version to return as a search result; or clustering documents of similar type; or grouping together documents of similar MIME type



What are Web Duplicates?

- **One example is the same page, referenced by different URLs**

http://espn.go.com http://www.espn.com

•How can two URLs differ yet still point to the same page?
- the URL's host name can be distinct (virtual hosts),
- the URL's protocol can be distinct (http, https),
- the URL's path and/or page name can be distinct

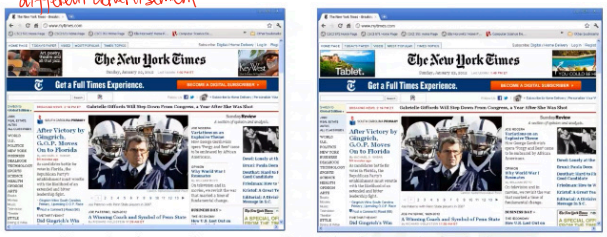
Distinct URLs May Deliver the Same Page

- The 3 URLs below all point to the identical page
- Structural Classification of Proteins
  - http://scop.mrc-lmb.cam.ac.uk/scop
  - http://scop.berkeley.edu/
  - http://scop.protres.ru/

Near Duplicates-An Example

Another example is two web pages whose content differs slightly

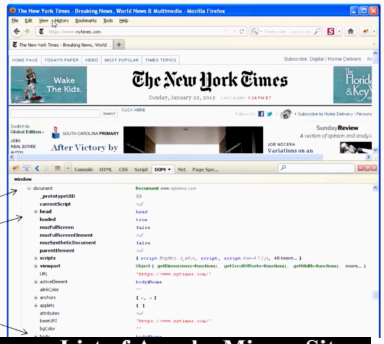
different advertisement



Two copies of www.nytimes.com snapshot within a few seconds of each other;
The pages are essentially identical except for the ads to the left and right of the banner line that says ***The New York Times***;

Spotting Near Duplicates

- In examining a web page a search engine may want to ignore ads, navigation links and other elements that do not specifically relate to the contents of the web page
- The structure of a web page can be used to identify the content blocks in the page
- The Document Object Model for HTML displays a web page as a tree hierarchy
- Document
- Head
- Body



Web Duplicates-Mirroring

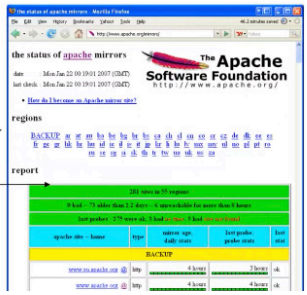
to serve more people, increase redundancy

- Mirroring is the systematic replication of web pages across hosts.
  - Mirroring is the **single largest cause** of duplication on the web
- Host1/α and Host2/β are mirrors iff
  For all (or most) paths p such that when
  http://Host1/ α / p exists
  http://Host2/ β / p exists as well same structure
  with identical (or near identical) content, and vice versa.

List of Apache Mirror Sites

List of countries

281 sites in 55 regions



Why Detect Exact Duplicates

- ***Smarter crawling***
  - Avoid returning many duplicate results to a query
  - Allow fetching from the fastest or freshest server
- ***Better connectivity analysis***
  - By combining in-links from the multiple mirror sites to get an accurate PageRank
  - Avoid double counting out-links
- ***Add redundancy in result listings***
  - "If that fails you can try: <mirror>/samepath"
- ***Clearly***: Crawlers need not crawl pages that are identical or near identical
- ***Ideally***: given the web's scale and complexity, priority must be given to content that has not already been seen before
  - Saves resources (on the crawler end, as well as the remote host)
  - Increases crawler politeness
  - Reduces the analysis that a crawler will have to do later

Why Detect Near Duplicates

- **Clustering – BIG USE CASE FOR CONTENT DETECTION AND ANALYSIS**
  - Given a news article some people might wish to see "related articles" describing the same event
- **Data extraction**
  - Given a collection of similar pages, e.g. movie reviews, try to identify the schema underlying the collection so one can extract and categorize the information
- **Plagiarism**
  - Identify pairs that seem to have significantly borrowed from each other
- **Spam detection**
  - Spammers typically send similar emails en masse, so one can use near-similarity techniques to identify the spam
- **Duplicates in domain-specific corpora**
  - To identify near-duplicates arising out of revisions, modifications, copying or merging of documents

Solving the Duplicate/Near-Duplicate Detection Problem

1. ***Duplicate***: **Exact match**
   - Solution: compute fingerprints using cryptographic hashing
   - SHA-1 and MD5 are/were the two most popular cryptographic hashing methods
2. ***Near-Duplicate***: **Approximate match**
   - Solution: compute the syntactic similarity with an edit-distance measure, and
   - Use a similarity threshold to detect near-duplicates
     - e.g., Similarity > 80% => Documents are "near duplicates"

Identifying Identical Web Pages – Four Approaches

1. ***Compare character by character*** two documents to see if they are identical
   - very time consuming !!
2. ***Hash just the first few characters and compare*** only those documents that hash to the same bucket
   - But what about web pages where every page begins with <HTML> ??
3. ***Use a hash function*** that examines the entire document
   - But this requires lots of buckets
4. ***Better approach*** - pick some fixed random positions for all documents and make the hash function depend only on these:
   - This avoids the problem of a common prefix for all or most documents, yet we need not examine entire documents unless they fall into a bucket with another document
   - But we still need a lot of buckets

Identifying Near Identical Web Pages - Two Approaches

1. ***Test for similar fingerprints*** - Treat web documents as a set of features, constituting an *n*-dimensional vector, and transform this vector into an *f-bit* fingerprint of a small size
   - Use Simhash and Hamming Distance to compute the fingerprint (more about this later)
   - Compare fingerprints and look for a difference in at most k bits
   - E.g. see Manku et al., WWW 2007, Detecting Near-Duplicates for Web Crawling, http://www2007.org/papers/paper215.pdf
2. ***Compute syntactic similarity***
   - Shingling / w-grams
     - Compute w-shingling (more about shingling later)
     - Broder et al., WWW 1997, Finding Near Duplicate Documents

General Paradigm

1. Define a function *f* that captures the contents of each document in a number
   - E.g. hash function, signature, fingerprint
2. Create the pair <*f*(doc$_i$), *ID of doc*$_i$> for all doc$_i$
3. Sort the pairs
4. Documents that have the same *f*-value or an *f*-value within a small threshold are believed to be duplicates

General Properties of Distance Measures

- **Distance measure must satisfy 4 properties**
  1. No negative distances
  2. d(x,y) = 0 iff x=y
  3. d(x,y) = d(y,x) symmetric
  4. d(x,y) <= d(x,z) + d(z,y) triangle inequality
- **There are several distance measures that can play a role in locating duplicate and near-duplicate documents**
  1. *Euclidean distance* – d([x_1…x_n], [y_1,…,y_n]) = sqrt(Sum(x_i-y_i)^2) i=1…n
  2. *Jaccard distance* – d(x,y) = 1 – SIM(x,y) or 1 minus the ratio of the sizes of the intersection and union of sets x and y
  3. *Cosine distance* – the cosine distance between two points (two *n* element vectors) is the angle that the vectors to those points make; in the range 0 to 180 degrees
  4. *Edit distance* – the distance between two strings is the smallest number of insertions and deletions of single characters that will convert one string into the other
  5. *Hamming distance* – between two vectors is the number of principal components in which they differ (usually used on boolean vectors)

Computing Near Similarity

- **Definition of Shingle**:
  - a contiguous subsequence of words in a document is called a *shingle;* The 4-shingling of the phrase below produces a bag of 5 items:
    "a rose is a rose is a rose" => a set S(D,w) is defined as
    { (a_rose_is_a), (rose_is_a_rose), (is_a_rose_is), (a_rose_is_a), (rose_is_a_rose)}
  - *S(D,w)* is the set of shingles of a document *D* of width *w*
- **Similarity Measures**
  - *Resemblance(A,B)* is defined as
    size of (S(A,w) intersect S(B,w)) / size of (S(A,w) union S(B,w))
  - *Containment(A,B)* is defined as
    size of (S(A,w) intersect S(B,w)) / size of (S(A,w))
  - 0 <= Resemblance <= 1
  - 0 <= Containment <= 1

- See *On the resemblance and containment of documents*, Conf. on Compression and Complexity, DEC Research Center, 1997

Shingling Example

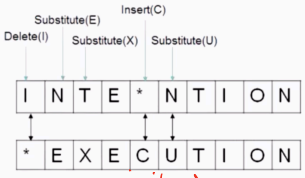
- **Original text**
  - "Tropical fish include fish found in tropical environments around the world, including both freshwater and salt water species"
- **All 3-grams (there are 16 of them)**
  - (Tropical fish include), (fish include fish), (include fish found), (fish found in), (found in tropical), (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)
- **Hash values for the 3-grams**
  - 938, 664, 463, 822,492, 798, 78, 969, 143, 236, 913, 908, 694, 553, 870, 779
- **Select only those hash values that are divisible by some number, e.g. here are selected hash values using *0 mod 4***
  - 664, 492, 236, 908
- **Near duplicates are found by comparing fingerprints and finding pairs with a high overlap**
  - Example taken from *Search Engines: Information Retrieval in Practice*

Jaccard Similarity and Shingli

- The Jaccard similarity of sets A and B, J(A,B), is defined as
- | A intersect B | / |A union B |
- A *k-shingle* of a document is any substring of length *k* found in the document
  - We ignore multiple blanks, tabs, newlines
- Choosing an appropriate value for *k*
  - If *k* is too small, then most sequences of length *k* will occur in most documents
  - *k* should be picked large enough that the probability of any given shingle appearing in any document is low
  - E.g. suppose in email only letters and white space occur (this is a simplified example); then there are $27^5 = 14,348,907$ possible 5-shingles; since emails are generally small, the value *k=5* should work well
- In the next few slides we will show how **computing shingles is equivalent to computing the Jaccard Similarity**

Edit distance for short documents

- **Consider the feature set of two documents A, B as a set of characters**
- **How many changes to go from set A to set B?** edit distance
- **Tuning parameters**
  - **Cost of each operation**



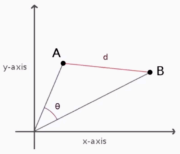
With a distance metric you can *cluster*

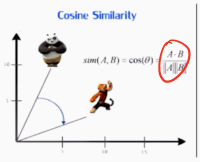
why close similar do
many cluster algorithms are based on distance

Cosine similarity

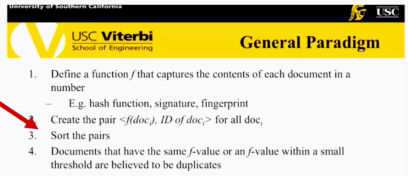
important

- **Don't treat features as sets per-se, instead graph them**
- **Then compute the cosine of the angle between the two graphs**
  - **1-d shown**
  - **Can you imagine for multiple features?**





- **Basic Clustering**
  - **Hierarchical**



  - **With <jaccard/edit/cosine> distance as f<doc_i>**
  - **Define a *threshold* that defines the max accepted delta between f(doc_i) and f(doc_i+1)**