Estimating similarity of web pages

Angelo Romano
Course of Web Mining 2005/2006
University of Pisa, Italy

Summary

Similarity

- *Similarity in the Web, and related issues*
- *Math concepts (formulas)*
- *Shingling*

Locality Sensitive Hashing

- *How it works*
- *The LSH function family*
- *Some basic theorems and rules*

How to generate a page sketch

- *Clean-up issues on documents from the Web*
- *Introduction to min-wise independent permutations*
- *A practical sample method*

Introduction

Why to estimate similarity between pages?

- Good estimation for clustering/classifying pages
  - *“Similar” pages get into the same class of pages*
- A way to avoid crawling duplicates
  - *The web is quite full of duplicate pages, or with a little percentile of differences*
- A way to improve quality of crawlings
  - *Refinement of web search result*

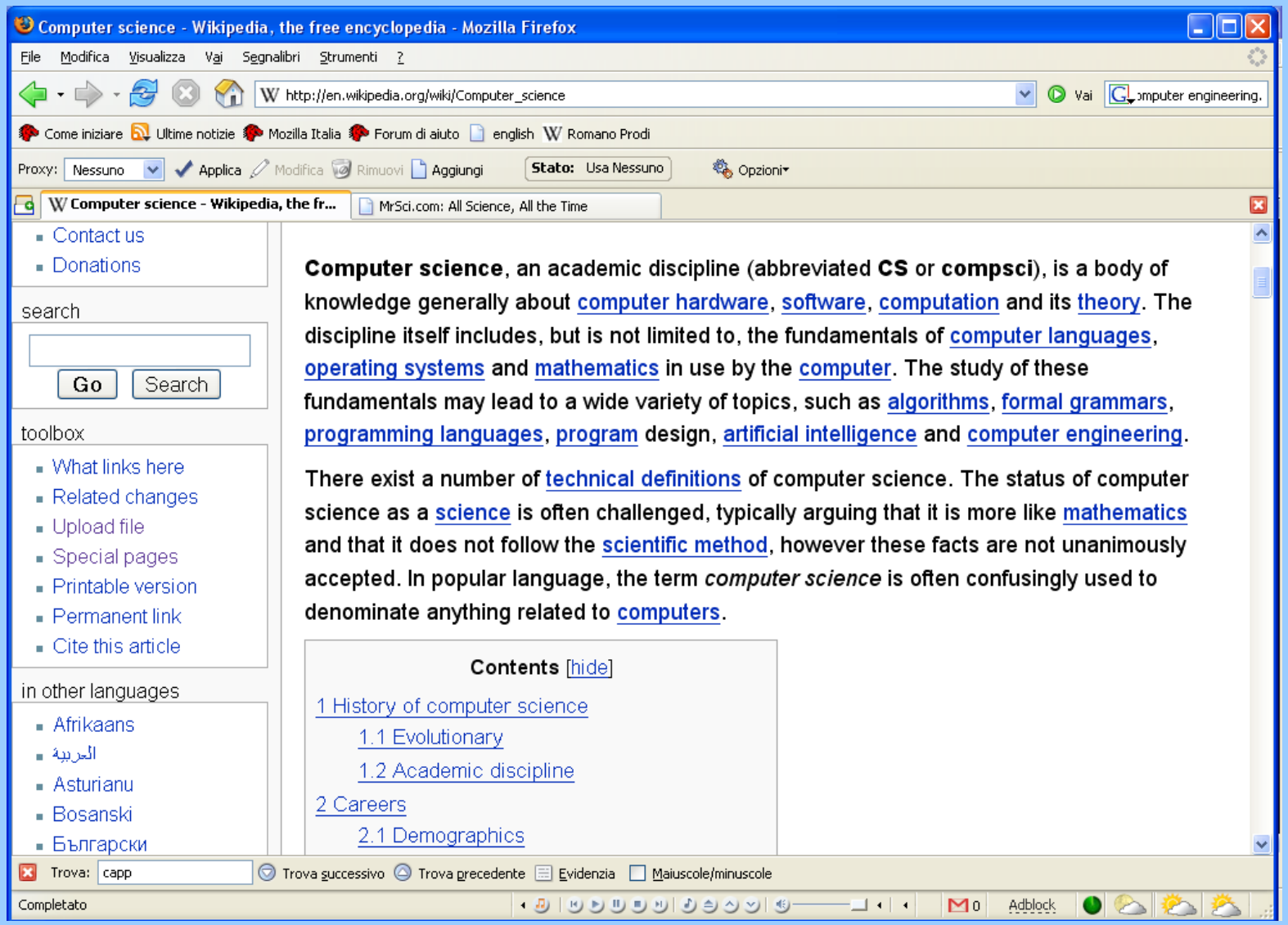
Similarity in the web

- Very similar, or identical, pages quickly proliferate in the web jungle
  - *5.3mln out of 30mln are identical (Broder 1997)*
  - *Online documentations, mirrored documents, legal documents, RFCs...*
- A URL-based approach is not a solution
  - *URLs may be instable (e.g. dynamic web pages)*
  - *The associated content might change in the future*

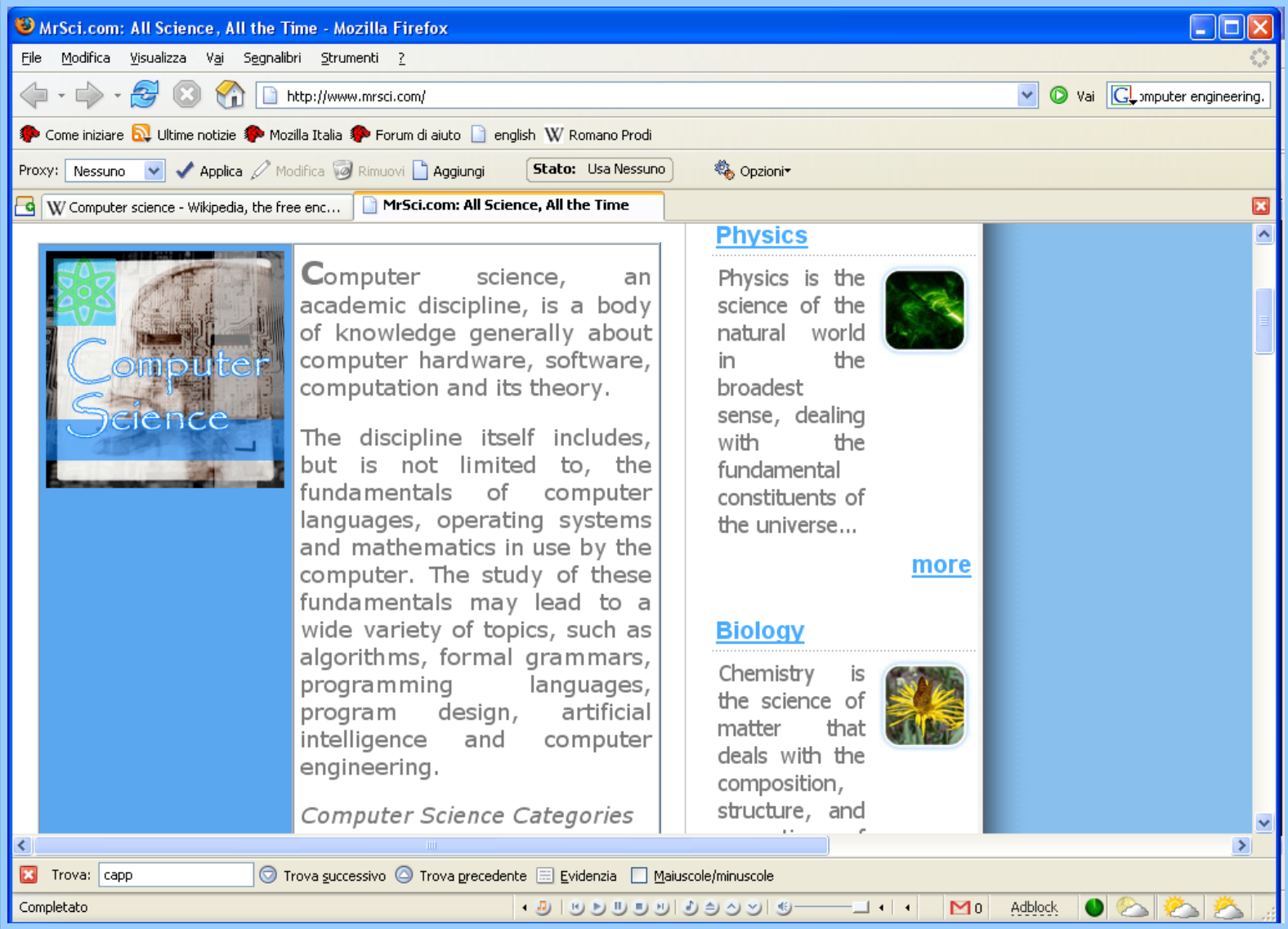
Similarity in the web (II)

- And a page may be identical, even if NOT identical (sorry for the wordplay!)
  - *Different formatting*
  - *Different, site specific, links*
  - *Splitted into smaller documents, or included in a larger one*

Similarity in the web (III)



Similarity in the web (III)



Similarity in the web (IV)

- Thus, a strict comparison of page contents will not work in the end
- A ideal similarity approach should work against all these issues
- And of course, a quick computation time for all of this

But...what actually similarity is?

Let us define a concept of "similarity measure" or "resemblance"

- sim (a,b), valued between 0 and 1 (included)
  - *How much two pages are intended to be "similar" each other*
  - *When this value is strictly close to 1, the related documents are defined "roughly the same"*
  - *It enjoys the reflexive property: sim(a,b)=sim(b,a)*
  - *...and sim(a,a)=1 (naturally)*
- How to define it?
  - *Several different formulas*
    - *Jaccard, Dice, and other probabilistic methods*
    - *Keep in mind vector-based approaches*

And then... containment!

- *cnt*(a,b), valued between 0 and 1 (included)
  - *A estimation of "how much" a document is part of another one*
  - *When this value is strictly close to 1, **a** is defined "roughly contained within **b**"*
  - *It does not enjoy the reflexive property*

- Why this measure?
  - *Remember: a document may be splitted in several ones, or (vice versa) be part of another one*

Some other notation

- Shingles
  - *A contiguous subsequence of words in a document*
  - *e.g. The 4-shingling of*
    `(a,rose,is,a,rose,is,a,rose)`
    is
    `{(a,rose,is,a),(rose,is,a,rose),(is,a,rose,is)}`
  - *Bag-of-words approach, similarity calculated basically with Jaccard coefficient:*
    - $r(A,B)=|S(A)\cap S(B)| \,/\, |S(A)\cup S(B)|$
    - $c(A,B)=|S(A)\cap S(B)| \,/\, |S(A)|$
- Resemblance distance
  - *How much a pair of documents are quantitavely distant each other, according to their similarity*

Introduction to LSH

- LSH (Locality Sensitive Hashing)
  - *Introduced by Indyk and Motwani*
  - *For nearest-neighbour search issues with small memory space needings*
  - *A family of hashing function in which, given h(x) its hash function,*
    - *Prob [h(A)=h(B)] = sim(A,B)*
  - *We may use the notion of **min-wise independent permutations** (introduced later) to construct hash functions LSH-valid under the Jaccard similarity coefficient*
  - *There are also vector-based approaches for this family, which however needs some geometrical knowledges and different similarity measures*

Introduction to LSH (II)

- Theorem: If *sim(A,B)* admits a LSH function family, then *d(A,B) = 1 – r(A,B)* **obeys the triangle inequality**
  - *Supposing to have a function family such that* *sim(A,B) = Prob* [*h(A) = h(B)*], then *1 - sim(A,B) = Prob* [*h(A) ≠ h(B)*]; *let Δh(A,B) be the indicator variable for the event h(A) ≠ h(B), being 1 when the condition is true, and 0 otherwise, we claim that it satisfies the triangle inequality:*
    - *Δh(A,B)+ Δh(B,C) ≥ Δh(A,C)*
    - *that is also: E* [*Δh(A,B)*] + *E* [*Δh(B,C)*] ≥ *E* [*Δh(A,C)*] *Then we may observe that*
    - *E* [*Δh(A,B)*] = *Prob* [*h(A) ≠ h(B)*] = *1 - sim(A,B)*

Introduction to LSH (III)

- I know, guys, it is all so damn annoying, but what is the real thing?
  - *The theorem shown before can be used to prove that some similarity measures does not fit at all for the LSH function family*

$$sim_{Dice}(A,B) = \frac{2|A \cap B|}{(|A| + |B|)} \qquad sim_{Overlap}(A,B) = \frac{|A \cap B|}{\min(|A|,|B|)}$$

  - *It may easily be proven that LSH functions does not exists for Dice and Overlap coefficient (related formulas shown above)*

Introduction to LSH (IV)

- **Theorem**: given an existing LSH function family F, having similarity function *sim(A,B)*, we can obtain another function family F', that maps objects binarily, with similarity function *(1+sim(x,y)) / 2.*
- ***Proof**: Suppose to have a hash function family F, such that* $sim(x, y) = \text{Prob}_{h \in \mathcal{F}}[h(x) = h(y)]$
- *Let B be a pairwise independent family of hash function in the domain of the functions in F, and map elements in the domain* {0,1}.

$$\text{Prob}_{b \in \mathcal{B}}[b(u) = b(v)] = 1/2 \text{ if } u \neq v$$
$$\text{Prob}_{b \in \mathcal{B}}[b(u) = b(v)] = 1 \text{ if } u = v$$

- *Thus, the composition family of F and B, named F', respects the following rule:*

$$\begin{aligned} \text{Prob}_{h \in \mathcal{F}, b \in \mathcal{B}}[b(h(x)) = b(h(y))] &= sim(x, y) + (1 - sim(x, y))/2 \\ &= (1 + sim(x, y))/2 \end{aligned}$$

How to generate a signature

- Now that we have showed what LSH is, we should get in touch with some appropriate hashing generation function.
  - **Min-wise independent permutations**
    - *It is the commonest method for generating signatures according to LSH*
  - It is appropriate also to have a **pre-processing** cleaning step first
    - *Any information different than real content (e.g. JavaScript code, HTML comments, HTML tags...)*
    - *Non-alphabetic information*
    - *No stopwords*
    - *Porter's stemming algorithm for simplifying words*

Pre-processing step

- Much more important than it actually seems, even a key step for higher-quality crawlings
- **Stopwords**
  - *Very frequent in linguistical distribution of words (that is represented by a Zipf law)*
  - *Like “the”, “a”, “of”, “with”... in English*
  - *We do not need them, they are actually does not represent relevant information*
- **Porter's stemming algorithm**
  - *Every word has different variants, derivations, etc...*
  - *“connect” => “connection”, “connected”, “connections”, “connector”, “connecting”...*
  - *It is thus useful to normalize these words and represent them with just one keyword*

Min-wise independent permutations

- *Min-wise independence condition*

$$\mathbf{Pr}\left(\min\{\pi(X)\} = \pi(x)\right) = \frac{1}{|X|}$$

  - We require all the elements of any fixed set X to have an equal chance to become the minimum element of the image of X under π
- In our case, π is a permutation (i.e. π: X → X)
- We assume π to be chosen uniformly at random in a family of permutations
- *String-defined distances, such as Hamming, do not fit good for our problem (pairwise computation of entire large documents, among the disadvantages)*
- *We use **sketches**: quick to compute, small enough to be compared, and good for representing a document*

Min-wise independent permutations (II)

- *Sketches: linear time, in the size of the document*
- Resemblance between two document also computed in linear time, in the size of the sketches
- Let us choose a random permutation π over S_n, that is the set of permutations of [*n*]*;* we can demonstrate that

$$\mathbf{Pr}\big(\min\{\pi(S_A)\} = \min\{\pi(S_B)\}\big) = \frac{|S_A \cap S_B|}{|S_A \cup S_B|} = r(A,B)$$

- Hence, we choose a number of independent random permutations (100, we say), and store the list

$$\bar{S}_A = (\min\{\pi_1(S_A)\}, \min\{\pi_2(S_A)\}, \ldots, \min\{\pi_{100}(S_A)\})$$

- Then we can readily estimate resemblance between different documents just by computing how many corresponding element are between their sketches.
- Nice, eh?!?

Min-wise independence issues

- Unfortunately, it is impossible to choose uniformly at random π in S_n; we must actually think to consider smaller families of permutations that still satisfies the min-wise independence condition.
- In practice, we accept small relative errors, under a policy of **approximate min-wise independence**:

$$\left|\mathbf{Pr}(\min\{\pi(X)\} = \pi(x)) - \frac{1}{|X|}\right| \leq \frac{\epsilon}{|X|}$$

- Another relaxation we allow is **restricted min-wise independence**:

$$\mathbf{Pr}(\min\{\pi(X)\} = \pi(x)) = \frac{1}{|X|}, \qquad |X| \leq k$$

- Thirdly and finally, it is important to maintain a distribution as uniform as possible in order to have qualitatively good results.

A practical method

- Fix a shingle size *w*; let U be the set of all shingles of size *w*. Fixed a parameter *s*, we define $MIN_s(W)$, with W subset of U, and $MOD_s(W)$ as

$$MIN_s(W) = \begin{cases} \text{the set of the smallest } s \text{ elements in } W, & \text{if } |W| \geq s, \\ W, & \text{otherwise} \end{cases}$$

$$MOD_m(W) = \text{the set of elements of } W \text{ that are } 0 \bmod m$$

- With π permutation over U chosen uniformly at random, S the shingling function over a document, $F(A)=MIN_s(\pi(S(A)))$ and $V(A)=MOD_s(\pi(S(A)))$, we may demonstrate these unbiased estimates of r(A,B)

$$\frac{|MIN_s(F(A) \cup F(B)) \cap F(A) \cap F(B)|}{|MIN_s(F(A) \cup F(B))|} \quad \textit{and} \quad \frac{|V(A) \cap V(B)|}{|V(A) \cup V(B)|}$$

- and the unbiased estimate of c(A,B) $\frac{|V(A) \cap V(B)|}{|V(A)|}$

A practical method (II)

- Pratically, we can proceed keeping for each document a sketch consisting only of the set F(D) and/or V(D), chosen a random permutation.
  - *F(D) has a fixed size, but allows to estimate resemblance only*
  - *V(D) grows as D grows, but it allows to estimate both resemblance and containment*
    - *For limiting its size, we can use a "modulus" system, by choosing the parameter $m = 2^i$ for document sized between $100*2^i$ and $100*2^{i+1}$, causing a set size $V_i(D)$ always between 50 and 100.*
    - *We can easily compute $V_{i+1}(D)$ from $V_i(D)$, just keeping only those element divisible by 2^{i+1}*
    - *Disadvantage: error proneness for estimating containment of very short documents into much larger ones*

The shingle issue

Very common shingles are heavily reduced

- HTML tags, stopwords, and other extremely common sequences have been removed during the pre-processing phase

Exactly identical documents

- We can define two documents as quite identical just comparing their shingles each other
- But sketches are calculated over shingles, thus they will also be the same

The shingle issue (II)

Super shingles

- Generated by sorting the sketch's shingles and then shingling them
- Good way to estimate similarity between sketches

Then we could generate meta-sketches from super shingles.

If the number of shingles in a super shingle is chosen correctly, it is highly probably to have at least one common super shingle in two similar documents

But there is also a loss of accuracy (especially for smaller documents), and impossibility to detect containment

Bibliography

- [1] Broder A., "On the resemblance and containment of documents", 1997
- [2] Broder A., Charikar M., Frieze A., Mitzenmacher M., "Min-Wise Independent Permutations", 1998
- [3] Broder A., Glassman S., Manasse M., Zweig G., "Syntactic Clustering of the Web", 1997
- [4] Charikar M., "Similarity Estimation Techniques from Rounding Algorithms", 2002
- [5] Haveliwala T., Gionis A., Indyk P., "Scalable Techniques for Clustering the Web", 2000
- [6] Haveliwala T., Gionis A., Klein D., Indyk P., "Evaluating Strategies for Similarity Search on the Web", 2002
- [7] Indyk P., Motwani R., "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality", 1999

Questions?