Semantically Mapping the Web

Eduardo Ramirez and Ramon Brena

Tecnologico de Monterrey, Mexico

Abstract. The millions of web pages populating the internet seem to be unstructured and chaotic, but there are implicit semantic relations between them. In this paper we propose to make explicit the underlying semantic structure of the internet, by measuring joint keyword occurrences in web pages, around our notion of "Semantic Contexts". As a result, we can draw a "map" of semantic clusters which can be used as a reference for situating individual web pages in a complex semantic space. Further, the methods we propose could be used for disambiguating and refining web search queries, for refining translations, for spam filtering, and in general for semantic-enabling many internet applications.

1 Introduction

Internet is acknowledged as one of the big technological revolutions of our time; since its inception in the early 90s, the WWW has grown exponentially, reaching some 74.5 millions of websites with at least 11.5 billions indexed at the main web searchers [1]. Nevertheless, web pages normally have the limitation of not taking into account the meaning or the context of the included information content, but just its formatting. HTML tags indicate that a certain text is a title, or a series of items, but not what the document is about. In words of T. Berners-Lee, "Most of the Webs content today is designed for humans to read, not for computer programs to manipulate meaningfully" [2]. This is indeed a serious limitation; for instance, one very important issue is to determine what a given web page is about. The lack of an efficient semantic categorization undermines many internet applications, in particular web searches. Indeed, every internet user is confronted with the inconvenience of receiving from the search engines many irrelevant pages, due to the inability of search engines to contextualize keywords in meaningful concepts, areas, themes, etc.

Initiatives aiming to represent in web pages meaning, have been generically called "Semantic Web" [2] The Semantic Web initiative proposes markup languages, mainly based on XML [3], and develops technologies for defining and using concepts and relations among them in the so-called "ontologies".

Nevertheless, there has been problems for widely adopting semantic web. Some of the reasons are technical challenges, and other are practical issues. So we are turning our attention to quantitative approximate methods (sometimes called "soft" [4]) for characterizing internet semantic relations. In particular, we proposed to exploit the joint frequencies of keywords as representative of semantic closeness in the existing internet, not in an ideal or futuristic internet.

Advances in Computer Science and Engineering
Research in Computing Science 27, 2007, pp. 125-136

Received 24/02/07
Accepted 08/04/07
Final version 17/04/07

We propose in this paper a keyword-based quantitative semantic infrastructure that could make explicit an underlying internet semantic structure, as a collection of interrelated "Semantic Contexts", which constitute a sort of internet semantic "topography". The Semantic Contexts are a stable reference against which specific web pages or queries could be situated. In this paper we also present some practical applications of Semantic Contexts, in particular how to better focus web searches.

After this introduction, in the next section we give a technical presentation of our method, followed by some experimental results, then by a representative application, and then a comparison with related work, to end with a conclusion.

2 Our proposal

The basis of our approach is to make a semantic interpretation of joint keyword frequency. Central to our approach is the notion of "Semantic Contexts" (SC), which intuitively represent conceptual areas, around which a family of keywords appears frequently inside web pages. For instance, around a concept of "Tourism" there will be many keywords like hotels, reservations, flights, etc., which appear together in many web pages.

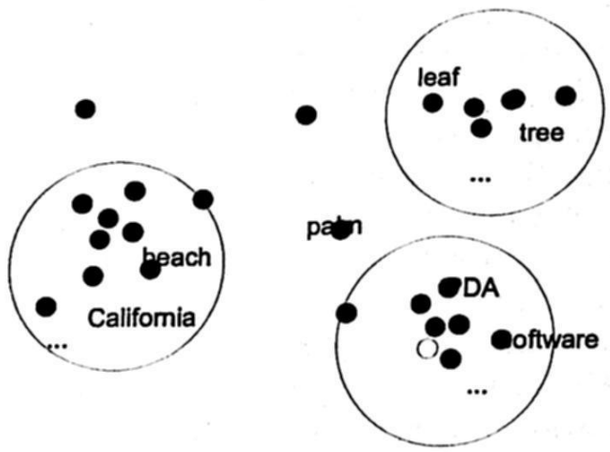

Fig. 1. Semantic Contexts related to "palm"

SC are defined formally in the next section, but let us first introduce a motivating example. Imagine the following scenario: a user is trying to find information about how to display pictures on a handheld device, so he/she issues the search query palm pictures. In a standard search service, like Google [5], this query would throw to the user answers about topics like: display of pictures on a handheld device, pictures of a place in California, pictures of an unbranched evergreen tree, etc. In a semantically-enhanced version of web search, the system would consult a base of indexed semantic clusters and would offer the user the following options: Search for palm pictures related to: 1) pda, software, 2)

California, beach, or 3) tree, leaf. Once the users would select one of these options, only pages of the corresponding interpretation of the word palm would be returned to the user. This, of course, would be of enormous utility to users, because search results would be much more focused.

2.1 Semantic Contexts

We see SCs as sets of interrelated keywords that appear together in a number of pages. We can visualize SC intuitively as "clouds" in a space of semantic closeness. like depicted in figure 1, for the "palm" example we just presented.

In order to formalize the notion of SCs, we consider the relative weights w_i of keywords k_i as a measure of how important these keywords are in a given topic. For instance, when we are talking about coffee, other related words like "sugar" or "roasted" have a high weight. We assume weights are normalized to be in the range $0 \leq w_i \leq 1$. Given a set K of n keywords, we define a SC as a function $\sigma : K \rightarrow [0, 1]$. As a SC represents a "topic", "subject" or "theme", important words in that theme have higher weights. We could also imagine SC as vectors $w_1, w_2, \ldots, w_n$ of weights for keywords $k_1, k_2, \ldots, k_n$.

Distances between SC can be readily calculated. First, we define SC similarity using a standard internal product formula [6]. Assume the vector $w_1, w_2, ..., w_n$ of weights for keywords $k_1, k_2, ..., k_n$ in a given semantic context SC is written as $\mathbf{w}$. Then, we can take the internal product of vectors as a similarity measure:

$$sim(SC_1, SC_2) = \frac{\mathbf{w_1} \bullet \mathbf{w_2}}{|\mathbf{w_1}|\,|\mathbf{w_2}|} \tag{1}$$

Then, from a similarity measure we could take the cosine inverse to obtain a difference measure as an angle [6]. Other similar distance metrics could be readily defined.

Next is the question of how a SC can be calculated. This can be done using specialized conjunctive queries, which we call "*k-cores*".

k-cores Now we introduce the notion of "k-cores", which are conjunctive queries, as follows. $F(w)$, called *frequency*, will represent in how many corpus pages the term w apears. By extension, the notation $F(\{w_1, \ldots, w_k\})$, for sets of k keywords $\{w_1, \ldots, w_k\}$, represents the count of web pages where all of $\{w_1, \ldots, w_k\}$ appear together (in the same document). Further, we use $F(\mathcal{P}\{w_1, \ldots w_k\})$ which is the set of frequencies, one for each subset of $(\mathcal{P}\{w_1, \ldots w_k\}$.

Then we define the "force" f of a keyword set $\{w_1, \ldots, w_k\}$ as follows, where c is a suitable constant, like 10^{12}; function g is explained below:

$$f(\{w_1, \ldots, w_k\}) = c\frac{F(\{w_1, \ldots, w_k\})}{g(F(\mathcal{P}(\{w_1, \ldots, w_k\})))} \tag{2}$$

and g is a function of joint frequencies of subsets of $\{w_1, \ldots, w_k\}$. One such function is the "disjoint frequency", which is the quantity of pages where a

given set of keywords (in this case $\{w_1, \ldots, w_k\}$) does *not* appear together, but some of $w_1, \ldots, w_k$ does appear.

Now we define "k-cores" as sets of k keywords of maximal force, meaning that replacing just one of its keywords by any other available word will decrease the force. k-cores can be seen as local maxima in a space of sets of keywords. This naturally suggests hill-climbing [7] as a method for finding them.

Depending on the application, the size k of k-cores could take different values. Of course, any value smaller than 2 does not make any sense, and even a value of 2 will normally be too small to represent a meaningful theme. In our experiments we use mostly a value of $k = 4$. Choosing the "right" value of k is an open question right now, and we have been rather pragmatic on this issue, generally taking a value of 4, with which the experiments gave meaningful results (see future work at the end of this paper).

k-cores are a key component of our method. They are conjunctive queries that represent a topic or subject. Once a k-core ω is determined, given a certain corpus C, the subset Ω of C with documents containing simultaneously all of the keywords in ω, can be readily obtained using web indexing technology [6]. From Ω, keyword weights can be computed using standard tf-idf measures [6], with a formula like:

$$w_{x,j} = f_{x,j} \times \frac{idf_x}{max_i\ idf_i} \tag{3}$$

where $f_{x,j}$ is the normalized frequency of term k_x in document d_j, and idf_i is the inverse document frequency for a generic term k_i.

In order to "mine" a set of web pages for finding k-cores, there is a trivial hill-climbing algorithm, as follows:

1: *Input*: A set P of web pages and a number k (for calculating size-k k-cores).
2: *output*: A set S_k of k-cores.
3: From P filter a set W of keywords.
4: $S_k \leftarrow \emptyset$ –The set of k-cores is initially empty.
5: **repeat**
6: $K \leftarrow$ a random subset of W of size k.
7: $F \leftarrow f(K)$
8: **for all** $w \in W - K$ **do**
9: **for all** $wk \in K$ **do**
10: $K' \leftarrow$ replace w for wk in K
11: $F' \leftarrow f(K')$
12: **if** $F' > F$ **then**
13: $F \leftarrow F'$; $K \leftarrow K'$
14: **end if**
15: **end for**
16: **end for**
17: $S_k \leftarrow S_k \cup K$
18: **until** k-cores are "stable"

The condition at the end of the **repeat** loop means that there are no changes in the set of current k-cores, meaning that this set is a fixpoint of the algorithm.

In practice, for efficiency reasons, this condition could be replaced (and actually has been replaced in our experiments) by a fixed number of iterations.

In order to assess the complexity of an algorithm we can evaluate first the inner loops. From the first *for* loop, the call to f function in the inner loop (which is normally the most costly operation) will be executed $|W - K|k$ times, and considering that k is kept constant and that $|W - K|$ is basically $|W|$, we can see that this algorithm is linear in the keyword set size $|W|$. This result stands of course if the outer loop is replaced by a fixed number of iterations, like we do in the experiments presented below.

We have introduced a small optimization to this algorithm: instead of starting with random cores, we select "promising" cores obtained in the following way:

1: From a random page p in the corpus we obtain the k most relevant terms using a TF-IDF measure [6, 8]
2: The starting k-core is the set of those k terms.

Mining a corpus for k-cores can be seen as locating the "topics" to which documents belong at least partially. We view the set of k-cores as the summits in a semantic topography, where altitude is calculated by the "force", given by equation 2. k-core calculation could be a computationally costly process, but it would be done offline in servers, so it does not affect the performance with respect to user queries, which we present in the following section. In the experiments section we show an example of k-cores calculation in a controlled environment.

3 Experiments

In order to validate the ideas presented above, we setup an experimental framework described in the following.

We installed an indexer and web searcher (Apache Lucene, [9]), and gather a small collection of 1168 web pages in the following topics: investments, java development, architecture, music, middle ages history and travel and tourism.

In order to provide an objective basis for classifying pages in topics, we used the Google and Yahoo directories, and using the APIs of these services for automatic downloads, avoiding in this way to introduce an involuntary bias. Of course, the Google and Yahoo directories were made by humans as well, but at least they were made by many people, and not including ourselves.

Then we ran the indexer in order to enable web searching inside our controlled set of web pages. The indexer created the index file and a table of keyword frequencies. We had a set of 50,025 words. In order to consider only meaningful keywords, we performed an automated filtering of "stop-words" (meaningless words, like "above", "etc", etc). 12,745 terms were filtered out, which is about a quarter of the total, leaving 37,280 keywords.

The next step was removing variants of the same words, like run, running, etc; this process is known as "stemming" [10]. In our prototype we used a stemming algorithm provided by the "Snowball" implementation of [11], which is not part of this research. We also added a database of similar words like "built" and "build" that were not caught by the Snowball system, so when the replacement

algorithm find a word similar to one in the current core, only is replaced if it is the word we are currently replacing, and the force is increased, otherwise it is discarded and we continue with the algorithm as presented in section 2.1.

Even in a small document sample like the one we have, with just over 1000 pages, the quantity of possible cores of a size 4 or 5 is quite impressive: there are 80,467,864,076,000,270 combinations of 37,280 words taken in groups of 4. This is the number of possible 4-cores, which of course excludes any brute-force algorithm for finding the best cores.

Actually, most of the cores have a force of exactly 0, because the numerator of the force formula 2 is the number of pages simultaneously having all of the considered keywords in it. The space of all possible cores contains a few (comparatively) sparse non-zero cores. In previous papers [12] we have found that the proportion of non-zero-force cores is about 0.018 percent. Taking into account this huge proportion of zero-force cores we can see that any refinement which avoid considering 0-force candidates would be a great improvement. We are using TF-IDF measures [8] of keyword relevance for selecting the best candidates, as we pointed out before. Consider that any word participating in a 4-core would necessarily be in some 2-cores (that is, sets of 2 words). For instance, two words appearing each in just 10 pages have a probability of appearing together in a given page of 7×10^{-8}, so it could be discarded. In our implementation we are forming initial cores ("seeds") by first selecting randomly one page in the corpus, and then selecting the 4 highest-valued words taking an TF-IDF measure. For instance, the most relevant words of a randomly selected web page, which was about japanese architecture, were *shinden*, *domestic*, *zukuri*, and *architecture*. We take this as a "seed" for the hill-climbing algorithm.

For the experiments of this paper, we implemented a variation of the algorithm in section 2.1, implemented in a "horizontal" way, meaning that we first calculated a single round of force increment, from initial seeds, and then calculated the second round from current cores, and so on. instead of going all the way to the maximum from initial seeds. This experiment is exhaustive for the small corpus we took, because initial seeds were calculated for every single page in the collection of over 1000 pages. In figure 2 we present the way the force of cores gets incremented from 2 rounds up to 6 rounds. We can see there that from 4 rounds-on variation is minimal, meaning that in practice there is no point repeating the "repeat" loop of algorithm in section 2.1 more than 4 times.

To illustrate the process, in the following table we present results from the "horizontal" hill-climbing algorithm, showing how many pages in the corpus, which produced corresponding initial seeds, are "concentrated" in the same 4-core. This means that some cores get frequently "merged" into the same core, resulting in gradually fewer and fewer cores as the algorithm proceeds.

As we can see in this table, for example the seed "architecture, building, design, house", which initially had just one page, then received the contribution of other cores that ended becoming identical to it by word substitution (see the algorithm), and had 52, 62 and finally 64 pages represented by it.

Core	*rnd 2*	*rnd 3*	*rnd 4*	*rnd 5*	*rnd 6*	Size var
{architecture,building,design,house}	1	52	62	64	64	63
{hotels,paris,rooms,rue}	57	83	91	92	92	35
{ages,life,middle,people}	59	88	90	90	90	31
{buffett,chairman,letter,warren}	6	25	25	25	25	19
{calderon,omar,reggaeton,tego}	17	22	22	22	22	5
{ages,feudal,middle,religion}	28	31	31	31	31	3
{band,blues,dance,jazz}	15	18	19	19	19	4
{application,code,developers,java}	5	9	12	12	12	7
{investors,premium,shares,stock}	0	6	9	9	9	9
{attractions,hotels,tourism,travel}	3	8	9	9	9	6
{berkshire,buffett,chairman,warren}	2	8	8	8	8	6
{cheap,hotel,reviews,star}	10	12	12	12	12	2
{artist,blues,jazz,pop}	1	5	5	5	5	4
{maze,mazes,pyramids,sphinx}	2	0	0	0	0	-2
{architectural,design,house,style}	2	0	0	0	0	-2
{ages,medieval,middle,weapons}	2	0	0	0	0	-2
{building,gate,great,middle}	2	0	0	0	0	-2
...						...

In the lower part of the table (below the horizontal line) we have some examples of cores which lost pages in the process, giving them away to stronger cores. These can be considered as meaningless word combinations in the corpus. The dotted last row represents the remaining 28 initial seeds of the experiment. Above the horizontal line we have all the 13 cores that ended with a positive size variation, which could be considered as the possible topics of the corpus.

Table 1. Cores found in our corpus

Topic	Cores
Investments	buffett,chairman,letter,warren
	investors,premium,shares,stock
	berkshire,buffett,chairman,warren
Programming	application,code,developers,java
Travel	hotels,paris,rooms,rue
	attractions,hotels,tourism,travel
	cheap,hotel,reviews,star
Music	calderon,omar,reggaeton,tego
	band,blues,dance,jazz
	artist,blues,jazz,pop
Architecture	architecture,building,design,house
Middle-ages history	ages,life,middle,people
	ages,feudal,middle,religion

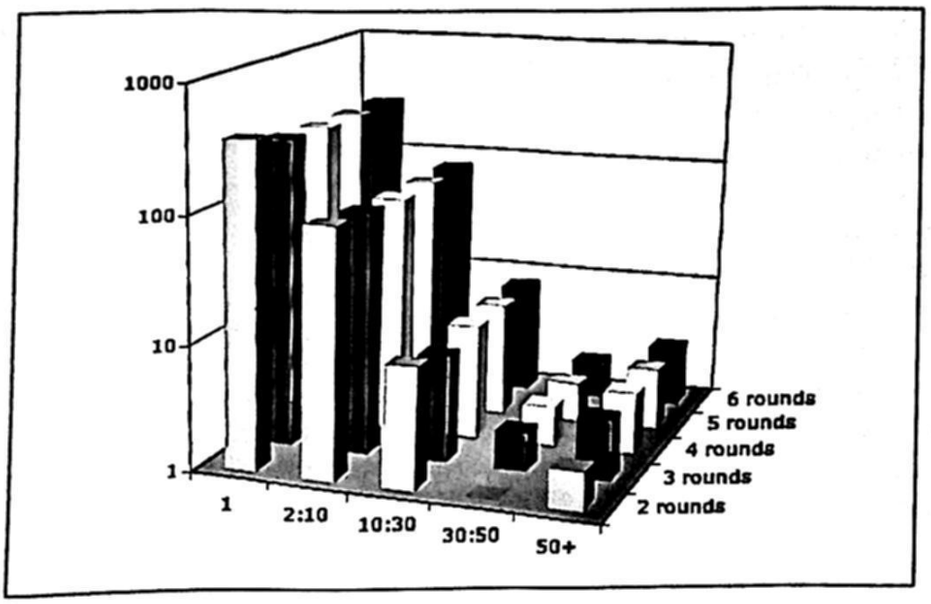

Fig. 2. Evolution of best core force through several rounds

Because we know in advance the topic of every page in the corpus, we can manually classify the 4-cores with positive variation into the 5 topics of the corpus, to see how the former represent the latter. This is done in table 3. From this table we can see that *all of the 5 topics got represented by at least one 4-core.* This is an important experimental result, because with it we showed that *the method was able to find all the relevant topics of the corpus.*

Nevertheless, some of the topics were represented by more than one 4-core. We believe that some of the cores are actually representative of a subtopic, like for instance "calderon, omar, reggaeton, tego", which is a subtopic of music. In order to test this hypothesis, we calculated the distances between all of the 13 SCs found, to see whether distances between semantically-related SCs were indeed smaller than distances between unrelated SCs. This is done in figure 3.

For this experiment we executed each of the cores as queries to the index, and then generated a weight vector using the resulting documents for each of the 13 selected cores, The weight of each term is its relative frequency in the result set. Then, the weight vector is unit-normalized so that vector distance metrics can be applied to each of the vectors in the set. As we can see in figure 2, k-cores in the same topic, like "buffett, chairman, letter, warren", "investors, premium, shares, stock" and "berkshire, buffett, chairman, warren" (all about the "investments" topic) are strikingly close to each other. The same could be said about same-topic k-cores in the other topics. Take, for instance the music-related k-cores: "band, blues, dance, jazz" is very close to "artist, blues, jazz,pop", and even "calderon, omar, reggaeton, tego", which does not share a single keyword with the other two k-cores, appears as semantically close in the figure. This validates the hypothesis that k-cores refer to subtopics of a general theme.

4 Application of SC to internet search

Now let us assume a set of web pages has already been mined for its k-cores, which will be considered each a representative of a SC. We will show how this structure could be used in order to guide an internet search.

The relevance of a keyword set or query Q to a given SC with k-core K and weights w_k, written as $R(SC_i, Q)$ is defined as the average weight w_k of the words $k \in Q$ in the SC_i.

The next step in SC-guided search is to calculate the relevance of Q with respect to the available SCs $SC_1 \ldots SC_n$. We order the SC Σ_i in descending order of relevance, and we take the first m relevant SC, where m is a small number like 2 or 3. These first m SC will be considered as the closest to the user query, and the associated k-cores will be presented to the user to choose from like in the introductory example.

Once the user selects one of the proposed k-cores, say K_i, the system will propose to him the result of queries $Q \cup \{k_i, k_j, \ldots\}$, where $k_i, k_j, \ldots$ are members of the selected k-core. This means that user queries can be enriched with words from the core, so that the search is narrowed. Notice that this will have a more restricted result than the original search Q, which is the intended effect.

Resuming, the algorithm for a SC-guided search is as follows (we assume that all k-cores satisfying a force threshold have already been calculated):

1: *Input*: A set SC_i of SC and a query Q.
2: *output*: Results from an enriched query.
3: Calculate the relevances $R(SC_i, Q)$ to SC, which are Σ_i.
4: Construct a list L of SC with decreasing relevance to the query.
5: Present to the user the m first k-cores from the SC in L.
6: The user selects one of the k-cores of the preceding step, let it be K_j.
7: New queries of the form $Q \cup \{k_i, k_j, \ldots\}$, where $k_i, k_j, \ldots$ are members of K_j, are constructed.
8: The user receives the result of the enriched queries.

Our experiments about SC-enhanced search are reported elsewhere [12].

5 Related work

As we mention in the introduction, there are a number of quantitative corpus-based approaches to analyze texts [13, 14], but none of them offers, as we do, a perspective of semantically structuring the web space; the cited works belong to the Natural Language Processing field.

In the field of Information Retrieval some works propose to extend search engines functionalities [15], by different means than the ones proposed in our paper, like "retrieving any type of data and collecting information to do better web mining", and other improvements like dealing with multimedia data. The cited author does mention the use of "Soft Computing" methods [16], but without proposing a specific approach or application.

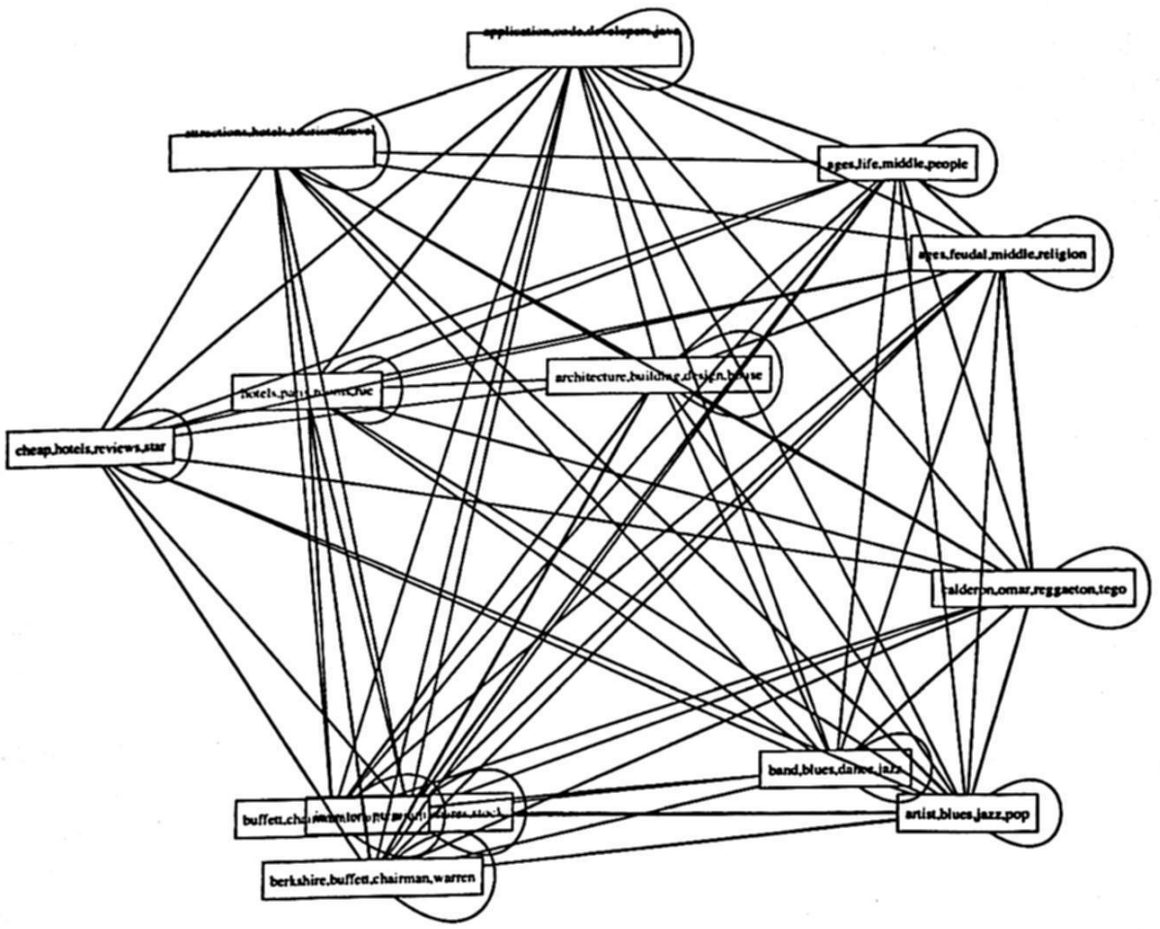

Fig. 3. Distances between SCs

In [17] quantitative information-theoretic measures of Semantic Similarity are explored using a tree-based notion of semantic similarity. Our work does not rely on graph comparison, but entirely on joint frequency measures, which are efficiently calculated by web search engines.

Some other works [18–20] propose clustering methods for sets of documents. For instance, in [21] "chat" sessions are put in relation to possible contexts using the web as a reference corpus; the author uses a clustering algorithm to identify candidate contexts. In his approach, a web search is done first, and clustering is applied to the search results. Our approach is not to directly cluster document sets, or search results like [21] or the Clusty search and clustering engine [22], but instead to first mine a corpus, that could be a document set or the whole internet, for Semantic Contexts, represented by their k-cores, and only then, match documents or queries against the k-cores; this last step is done efficiently using algorithms very similar to those used by search engines, which

were described in section 4. One advantage of doing so is that our k-cores are static, that is, they do not change from query to query, but only through years of internet evolution, and thus they can be calculated off-line, that is, prior to user querying, reducing this way the user waiting time.

A work in NLP similar in ideas to our work is [23], where the author presents a vector representation of keyword occurrences together. Topics are represented by the centroid of a set of vectors in a multidimensional space. There are complexity issues though, as the author declares: "a global optimization of cooccurrence constraints is necessary, an operation so complex that only a supercomputer can perform it". Our reliance on web search technology, in contrast, gives us, we think, better chances to scale up to the whole internet.

6 Conclusion

As we show in this paper, part of the underlying semantic structure of the web could be made explicit by means of our "Semantic Contexts", represented each by corresponding keyword weights and "k-cores". We have presented the notion of Semantic Context as "clouds" in a keyword space, we have formally defined them as weighting functions over keywords, and we have shown how they can be calculated. Further, the experiments we present show that it is possible to produce automatically k-cores representing all of the topics in the given corpus. As we are able to define distances over SC, we see the collection of SC in a corpus like a "map" of its semantic concentration points, or as a semantic "topography", with summits associated to cores with maximal force.

As a practical application of Semantic Contexts, search engines utility could be improved, using semantic contexts as a guidance. We think SC could be applied to automatic "tagging", to natural language translation, and in general to serve as an objective semantic reference that could make semantic-aware many internet applications.

To the best of our knowledge, our work, proposing the explicit construction of a static interrelated collection of semantic themes representative structures (Semantic Contexts and their k-cores), and their application for refining searches, is completely original.

Our future work includes a larger scale validation, the refinement of the algorithms to ensure scalability, the investigation of the effect of adjusting the k size of k-cores, as well as developing practical applications like the search refinement.

Acknowledgement: This work was supported by the CAT-011 Monterrey Tech's research chair.

References

1. Gulli, A., Signorini, A.: The indexable web is more than 11.5 billion pages. In Ellis, A., Hagino, T., eds.: Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005 - Special interest tracks and posters, ACM (2005) 902–903

2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (2001)
3. Andersen, A.: Construction of XML. XML journal (2001)
4. Zadeh, L.: Soft computing and fuzzy logic. Software, IEEE **11** (1994) 48–56
5. google. (`http://www.google.com`)
6. Baeza-Yates, R., Ribeiro-Neto, B., et al.: Modern information retrieval. Addison-Wesley Harlow, England (1999)
7. Russell, S., Norvig, P.: Artificial Intelligence a Modern Approach. AI. Prentice_Hall (1995)
8. Church, K., Gale, W.: Inverse document frequency (IDF): A measure of deviations from Poisson. Proceedings of the Third Workshop on Very Large Corpora (1995) 121–130
9. lucene. (`http://lucene.apache.org/java/docs/`)
10. Xu, J., Croft, W.: Corpus-based stemming using cooccurrence of word variants. ACM Transactions on Information Systems (TOIS) **16** (1998) 61–81
11. Porter, M.: An algorithm for suffix stripping. Program **14** (1980) 130–137
12. : (Self-citations excluded for blind review)
13. Brill, E., Mooney, R.J.: An overview of empirical natural language processing. The AI Magazine **18** (1998) 13–24
14. Ng, H.T., Zelle, J.M.: Corpus-based approaches to semantic interpretation in NLP. AI Magazine **18** (1997) 45–64
15. Baeza-Yates: Information retrieval in the web: Beyond current search engines. IJAR: International Journal of Approximate Reasoning **34** (2003)
16. Zadeh, L.A.: Fuzzy logic, neural networks, and soft computing. Communications of the ACM **37** (1994) 77–84
17. Maguitman, A., Menczer, F., Erdinc, F., Roinestad, H., Vespignani, A.: Algorithmic computation and approximation of semantic similarity. WWW Journal (2006)
18. Zhao, Y., Karypis, G.: Evaluation of hierarchical clustering algorithms for document datasets. In Kalpakis, K., Goharian, N., Grossmann, D., eds.: Proceedings of the Eleventh International Conference on Information and Knowledge Management (CIKM-02), New York, ACM Press (2002) 515–524
19. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering techniques (2000)
20. Adami, G., Avesani, P., Sona, D.: Clustering documents in a web directory. In Chiang, R.H.L., Laender, A.H.F., Lim, E.P., eds.: Fifth ACM CIKM International Workshop on Web Information and Data Management (WIDM 2003), New Orleans, Louisiana, USA, November 7-8, 2003, ACM (2003) 66–73
21. Segev, A.: Identifying the multiple contexts of a situation. In: Proceedings of IJCAI-Workshop Modeling and Retrieval of Context (MRC2005). (2005)
22. clusty. (`http://www.clusty.com`)
23. Schutze, H.: Dimensions of meaning. In: Proceedings Supercomputing'92, Minn., MN, IEEE (1992) 787–796