Web Information Retrieval Based on Syntax Distances

Carlos J. Castillo, Héctor Valero, Josep Silva

Universidad Politécnica de Valencia
Camino de Vera s/n E-46022 – Valencia, Spain
{carcasg1,hecvalli}@posgrado.upv.es, jsilva@dsic.upv.es

***Abstract*—Retrieving information from Internet is a difficult task as it is demonstrated by the lack of real-time tools able to extract information from webpages. The main cause is that most webpages in Internet are implemented using plain (X)HTML which is a language that lacks of structured semantic information. For this reason much of the efforts in this area have been directed to the development of techniques for URLs extraction. It has produced good results implemented by modern search engines. But, contrarily, extracting information from a single webpage has produced poor results or very limited tools. In this work we define a novel technique for information retrieval from single webpages or collections of interconnected webpages. This technique is based on syntax distances to retrieve information. This allows the technique to work with any webpage and, thus, to retrieve information online. Our implementation demonstrates the usefulness of the technique.**

***Index Terms*—Information Retrieval, HTML Filtering, Slicing Websites.**

I. Introduction

INFORMATION retrieval is one of the major areas of interest in both the web and the semantic web. The lack of real-time online applications able to automatically extract information from the web shows the difficulty of the problem. Current techniques for information retrieval from Internet are mainly based on the recovering of webpages that are related to a specified query (see [1] for a survey of this kind of techniques). In this area, search engines such as Google or Bing implement very efficient and precise algorithms for the recovering of related webpages. However, for many purposes, the granularity level of the produced information is too big: a whole webpage.

In the semantic web setting, it is often possible to produce more precise results composed of texts that answer a given question. However, these techniques often need to pre-process the webpages that are going to be queried. An ontological model is constructed and the knowledge is modelled and queried using languages such as RDF [2] or OWL [3]. This imposes important restrictions on the webpages that can be processed, and thus the implemented tools are usually offline tools. One reason is that most of Internet pages have been implemented with plain (X)HTML. A similar problem is faced by the tools that use microformats [4,5,6] to represent knowledge.

In this work we introduce a novel technique for information retrieval that is based on syntax distances. Roughly speaking the technique looks for a term specified by the user, and it extracts from the webpage those elements that are syntactically close to this term. Therefore, the technique relies on the idea that syntactically close means semantically related. This idea is also extended to distances between pages and domains using hyperlink distances. We have implemented the technique and several experiments and extensive use by thousands of users show that this simple idea is very powerful in practice.

The main advantages of the technique are that it can work online (with any webpage) without any pre-compilation or pre-parsing phases; and that it can retrieve information at a very low level of granularity: a single word.

The rest of the article has been organized as follows: In Section II we present the technique from a practical point of view by showing an example of use. Section III presents the technique and introduces an algorithm to retrieve information from single webpages and an algorithm to retrieve information from multiple interconnected webpages. In Section IV we describe our implementation and, finally, Section V concludes and gives some directions for future work.

II. Motivation

This section presents a real example of information retrieval using the technique presented in this paper. We consider a user that is browsing on the Internet looking for information related to the research done at the Technical University of Valencia. In the usual scenario the user loads the main webpage of the Technical University of Valencia. Then she looks for links related to research, and for each link she navigates to a new

Manuscript received March 1, 2010. This work has been partially supported by the Spanish *Ministerio de Ciencia e Innovación* under grant TIN2008-06622-C03-02, by the *Generalitat Valenciana* under grant ACOMP/2009/017 and by the *Universidad Politécnica de Valencia* (Programs PAID-05-08 and PAID-06-08).

webpage, reads the information and, again, the links of interest are navigated. During this process, the user is forced to (i) read much information not related to research which is present in the visited webpages, and (ii) look for the links of interests.

In a second scenario the user has available our tool for web information retrieval. At the main webpage of the Technical University of Valencia she only has to specify that she is interested in the information related to research. A single click produces a new webpage that contains all the relevant information of the main webpage and those linked webpages related to research.

Fig. 1. Main webpage of the Technical University of Valencia.

Figure 1 shows the main webpage of the Technical University of Valencia. Very few information is related to research. Our algorithm is able to filter a webpage and only show the relevant information according to the given filtering criterion. For instance, with the webpage in Figure 1, the algorithm would produce the new filtered webpage shown in Figure 2.

Fig. 2. Slicing the main webpage of the Technical University of Valencia.

The topmost word in Figure 2 is a hyperlink to the research area webpage. It is shown in Figure 3 where relevant words have been highlighted by the tool. This is a second functionality that is complementary to the filtering. Figure 4 shows the same webpage after a filtering process. Observe that even images and the main horizontal menu have been filtered.

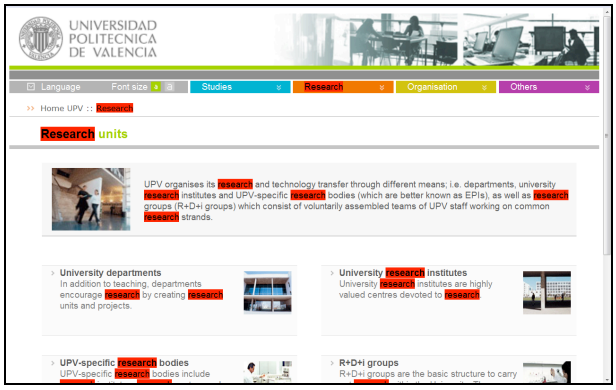

Fig. 3. Highlighting relevant information of a webpage.

All these transformations only affect a single webpage. However, in our example, we are interested in extracting information from a website, and we want to inspect several webpages by traversing relevant links.

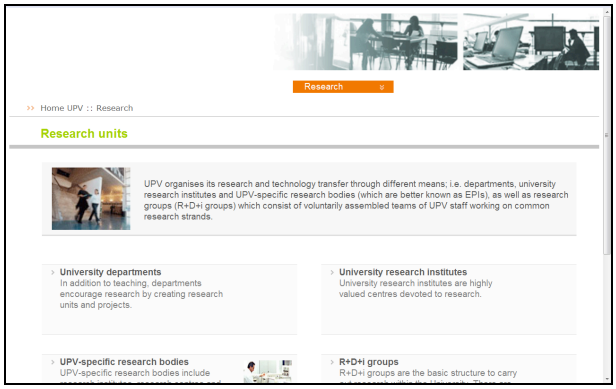

Fig. 4. Slicing relevant information from a webpage.

In order to extract the relevant information related to the criterion specified by the user, the tool must decide what links should be traversed, and for each of them, the tool must extract the relevant information (as it is done in Figures 2 and 4). Finally, all the information extracted for each webpage must be presented in a new webpage either hierarchically or organized with dynamically generated menus.

Figure 5 shows the result produced by our tool when retrieving information from the website of the Technical University of Valencia. Observe that this webpage is the result of extracting information from different webpages.

This tool does not need to use proxies [7] or to pre-process [8] the filtered webpages. It can work online with any webpage. Therefore, a very important point of the technique implemented in this tool is the speed achieved to produce and show the results. However, while processing webpages is a relatively fast task, loading webpages is a very slow task. Therefore, our algorithm process and shows the information incrementally page by page. This means that the user can start to see the results from the very beginning and, at the same time that she reads the already shown information, the tool is generating and showing new associated information.

Fig. 5. Information retrieval from a website.

Observe in Figure 5 that the collected information is shown in a single webpage so that the user can access to all the contents without navigating multiple pages. Moreover, those hyperlinks in the relevant text whose associated webpages were not considered relevant by the tool are still in the new webpage. Therefore, the user can still freely navigate to the contents that were discarded.

III. Information Retrieval Based on Syntax Distances

In this section we formalize our technique for information retrieval and visualization.

A. Information retrieval from single webpages

In [10] a technique for information retrieval from single webpages was proposed, but a formalization of the algorithm was not defined. In this section we introduce this algorithm and adapt it to be later used for our algorithm for information retrieval from multiple webpages.

The Document Object Model (DOM) [9] is an API that provides programmers with a standard set of objects for the representation of HTML and XML documents. Our technique is based on the use of DOM as the model for representing webpages. For the sake of concreteness, in the following we will assume that a DOM tree is a data structure that represents each single element of a webpage with a node labelled with a text. This simplification assumes that all nodes have a single attribute, and it allows us to avoid in our formalization and algorithms low-level details such as the distinction between different kinds of HTML elements' attributes. For instance, in our implementation we have to distinguish and query different properties depending on the element that we are analyzing, e.g., image nodes are queried by using their *alt*, *longdesc* and *src* attributes.

Definition 1 (DOM tree): A *DOM tree* $t=(V,E)$ is a tree whose vertices V are labelled nodes connected by a set of edges E.

In the following, we will refer to the label of a DOM node n with $l(n)$; and we will refer to the root of a DOM tree t with $root(t)$. We also use the notation $n \rightarrow^{x} n'$ to denote that there exists a path of size less or equal to x between nodes n and n'.

Definition 2 (Webpage): A *webpage* is a pair (u,t) where u is an URL and t is a DOM tree.

Queries of the user can often contain multiple words and metadata such as *""* for exact search, and boolean operators (*and*, *or*, *not*) to produce complex combinations of texts that force a particular order of words, or force the existence or inexistence of a particular (sub)text. However, for simplicity, in our formalization we will assume that queries are composed of a single word. The extension of the technique for multiple words is trivial and it only requires the iteration of the method over all the words of the query. This has been already done in our implementation, and thus, the interested reader is referred to its (open) source code for implementation details.

Definition 3 (Query): A *query* is a pair (w,d) where w is a word that is associated to the information which is relevant for the user; and d is an integer that represents the precision required in the search.

In our setting, the precision represents the distance between nodes of the DOM tree. The precision is used to decide what elements of the DOM tree are related to the user specified word. With a precision of 0, only elements that contain the specified word should be retrieved. With a precision of 1, only elements that contain the word and those that are in a distance of 1 to them should be retrieved, and so on.

We are now in a position to present our algorithm for information retrieval of a single webpage:

Algorithm 1: Information retrieval from single webpages

Input: A webpage $P=(u,t)$ and a query $q=(w,d)$
Output: A webpage $P'=(u,t')$

Initialization: $t=(v,e)$, $t'=(\emptyset,\emptyset)$

(1) key_nodes = $\{n \in v \mid l(n)=w\}$
(2) relevant_nodes = $\{n \in v \mid n \rightarrow^{d} n' \wedge n' \in$ key_nodes$\}$
(3) ancestors = $\{n \in v \mid n_0 \rightarrow^{*} n \rightarrow^{*} n_1 \wedge n_0=root(t) \wedge n_1 \in$ key_nodes $\cup$ relevant_nodes$\}$
(4) successors = $\{n \in v \mid n_0 \rightarrow^{*} n \wedge n_0 \in$ relevant_nodes $\}$
(5) edges = $\{(n,n') \in e \mid n,n' \in$ (successors $\cup$ ancestors)$\}$

return $P'=(u,$(successors $\cup$ ancestors, edges))

Clearly, Algorithm 1 has a cost linear with the size of the DOM tree. In essence, it finds the *key_nodes* that are those whose label is equal[1] to the searching word. From these nodes, the *relevant_nodes* are computed which are those whose syntax distance to the relevant nodes is lower than the precision specified by the user. This idea is an important contribution of this technique because it is a novel method to retrieve semantically related information. Our experiments and implementation together with massive use of anonymous users demonstrate the practical utility of this syntax distance. All the *ancestors* and *successors* of the relevant nodes form the final nodes of the filtered DOM tree. The final *edges* are those induced by the set of nodes. Therefore, the final webpage (that we will call in the following *slice*) is always a portion of the original webpage, and this portion keeps the original structure of information because the paths between retrieved elements are maintained.

Example 1: Consider the webpage in Figure 6 (left), and a user's query *("members",0).*

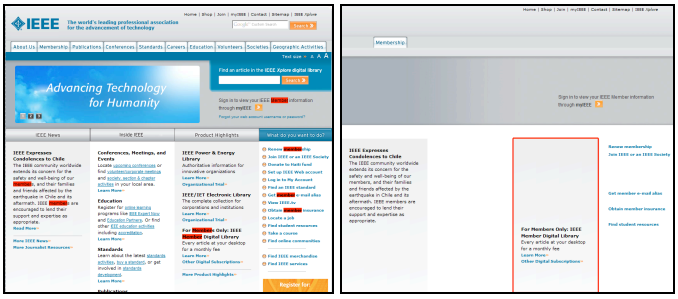

Fig. 6. IEEE main webpage (left) and its filtered version (right).

This webpage is produced from a huge DOM tree. The part of this tree that produces the central gray table on the right is shown in Figure 7.

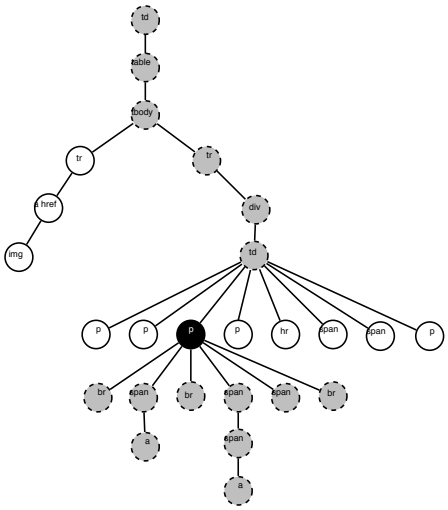

Fig. 7. A subtree of IEEE webpage's DOM tree.

In the figure, the black node is a key node (i.e., it contains the word *members*). Because the precision is 0, the black node is also the only relevant node. The ancestors are the top five gray nodes, and the successors are the other gray nodes at the bottom. All the white nodes are not related to the query and they must be filtered out. After filtering the whole DOM tree with Algorithm 1, we get the webpage in Figure 6 (right) where only the information related to IEEE members remains.

B. *Information retrieval from interconnected webpages*

In this section we extend our algorithm for information retrieval of interconnected webpages. In the following we will assume that the user has loaded a webpage (that we call *initial webpage*) and she specifies a query to extract information from this webpage, the webpages that are linked to it (either as incoming or outgoing links), the webpages included in it (e.g., as frames or iframes) and the webpages to which it belongs (e.g., as a frame or an iframe). We call all this pages the *interconnected webpages*; and observe that they are not necessarily in the same domain.

Frames and iframes can be modeled by considering that their DOM trees are subtrees of the webpage that contains them. Therefore, Algorithm 1 is able to extract relevant information from composite webpages structured with frames. For hyperlinks, we can assume that the label of some nodes in a DOM tree is a link pointing to a webpage. This is enough to define the notions of reachable webpages and search hyperspace used in our information retrieval algorithm.

Definition 4 (Reachability): Given a webpage P_0 we say that webpage P_n is reachable from P_0 iff $\exists\ P_0, P_1, \ldots, P_n \mid \forall\ P_i=(u,(v,e)),\ 0 \leq i \leq n\text{-}1, \exists\ n \in v\ .\ l(n)=u' \wedge P_{i+1}=(u',t)$.

Roughly speaking, a webpage is reachable from another webpage if it is possible to follow a sequence of hyperlinks that connect both pages from the later to the former.

Definition 5 (Search Hyperspace): Given a webpage $P=(u,t)$ the *search hyperspace* of P is the set of webpages that either are reachable following hyperlinks from nodes of P, or that can reach P from their hyperlinks.

The search hyperspace is the collection of webpages that are related to the initial webpage, and that should (ideally) be inspected by our information retrieval algorithm. However, the search hyperspace of a webpage is potentially infinite (even more when we surf dynamic webpages [11]), and it is often huge. Therefore we need to reduce it by discarding some of the hyperlinks. In addition, we want our technique to be online. This implies that time response is a critical factor, but each analysis of a webpage implies loading it, which is a time-consuming task. Therefore, reducing the number of webpages that are analyzed is a major objective of the technique.

With this aim, we define an ontological distance between nodes of the search hyperspace. This distance is used to decide what hyperlink nodes are more related to the query specified by the user and should be explored. The others are discarded. Using syntax distances to approximate semantic relations is an

[1] The restriction of being equal is taken for simplicity of presentation. In the implementation, a lexicon could be activated to also consider synonyms. Of course, standard semantic distances for word similarity can also be used.

idea that is supported by experimental evaluation of different works. For instance, Micarelli and Gasparetti [12] obtained empirical results demonstrating that webpages pointed by closer hyperlinks are more related semantically than webpages pointed by hyperlinks that are syntactically separated.

In order to define an ontological distance, we use the following concepts:

- *DOM distance (d_T):* It is the length of the path between two nodes of a DOM tree.
- *Page distance (d_P):* It is the lower number of hyperlinks that must be traversed to reach one webpage from another webpage.
- *Domain distance (d_D):* It is the lower number of domains that must be traversed to reach one webpage from another webpage following a path of hyperlinks.

We use the initial webpage and the key nodes as the reference to compute distances. Therefore, for a given node, (i) its DOM distance is the length of the path between this node and the closest key node in its DOM tree; (ii) its page distance is the lower number of hyperlinks that must be traversed to reach the webpage of this node from the initial webpage.; and (iii) its domain distance is the lower number of domains that must be traversed to reach the webpage of this node from the initial webpage following a path of hyperlinks.

Definition 6 (Ontological Distance, Relevance): Given a DOM node *n*, the *ontological distance* of *n* is:

$$D = d_T + K_P \cdot d_P + K_D \cdot d_D$$

where K_P and K_D are numeric constants used to weight the equation. The *relevance R* of a DOM node is the inverse of its ontological distance:

$$R = \frac{1}{D}$$

The constants K_P and K_D determine the importance that we give to the fact that the word specified by the user is in another page, or in another domain.

Example 2: Consider the following search hyperspace:

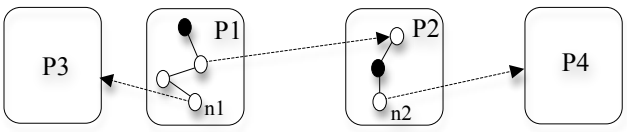

Fig. 8. Example of a hyperspace

where two nodes contain the word specified by the user (those in black); the first node is in the initial webpage (P1), and the second node is in webpage P2 and thus it has a page distance of 1. Now, observe that nodes n1 and n2 are hyperlinks to other webpages. The question is: *which hyperlink is more related to the query of the user and should be explored first by the algorithm?* The answer is clear: the node more relevant and thus with a smaller ontological distance. According to Definition 6, relevance strongly depends on the values of the constants K_P and K_D. Assuming that all the webpages are in the same domain and if K_P=1, then D(n1)=3 and D(n2)=2, thus n2 is more relevant. In contrast, if K_P=10, then D(n1)=3 and D(n2)=11, thus n1 is more relevant.

After several experiments and intensive testing we took the following design decisions:

1. Those hyperlinks that are in the initial webpage are more relevant than those in another webpage. And the same happens as the page distance is increased. Hence, the DOM distance is more important than the page distance.
2. Those hyperlinks that are in the same domain as the initial webpage are more relevant than those in another domain. And the same happens as the domain distance is increased. Hence, the page distance is more important than the domain distance.
3. The algorithm should never analyze a webpage with a page distance greater than 5. This is also supported by previous studies (see, e.g., Baeza and Castillo's discussion in [11]) that demonstrate that, in general, three to five levels of navigation are enough to get 90% of the information which is contextually related with the webpage selected by the user for the web search.

Therefore, considering the amount of nodes in a webpage, we take the following values: $K_P=10^3$ and $K_D=10^6$. Hence:

$$D = d_T + 10^3 \cdot d_P + 10^6 \cdot d_D$$

Example 3: Consider an initial webpage P1 and its search hyperspace shown in Figure 9:

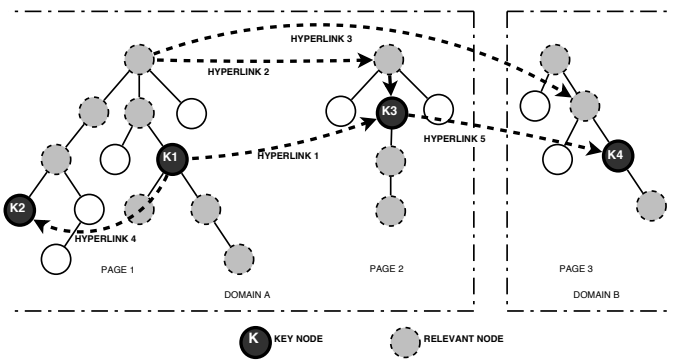

Fig. 9. Relevant information hyperlinked through different pages and domains

Assume that Algorithm 1 has analyzed the three webpages and thus, dark nodes are relevant (key nodes are black) and white nodes are discarded. In order to determine what hyperlinks are more relevant, we compute the relevance of their DOM nodes. Their ontological distance and relevance are computed in Table 1. The algorithm uses this information to decide what hyperlinks must be analyzed first. Observe in the example that the ontological distance of node *k4* is $0+2*10^3+1*10^6$. This node has a lower relevance because it is in another domain.

Hyperlink	d_T	d_P	d_D	D	R
1	0	0	0	0	∞
2	2	0	0	2	0.5
3	2	0	0	2	0.5
4	0	0	0	0	∞
5	0	1	0	1000	0.001

Table. 1. Ontological distance of the hyperlinks in Figure 9.

In a DOM tree we can distinguish between hyperlinks that belong to the slice and hyperlinks that do not belong to the slice. Those hyperlinks that do not belong to the slice are often related to webpages of none interest for the user. Therefore, to ensure the quality of the retrieved information we take a fourth design decision:

4. Hyperlinks that do not belong to the slice are discarded.

One important problem of extracting information from webpages happens in presence of dynamic webpages: A dynamic webpage could generate another dynamic webpage that contains the word specified by the user. This new dynamic webpage could do the same, and so on infinitely. This situation is known as black hole because robots searching in this webpages have an infinite space of search where they always find what they are looking for. Therefore they are trapped forever if no limit is specified in the search [11].

Observe that the combination of design decisions 3 and 4 avoid this problem because the search is stopped when a webpage does not contain key nodes, or when its page distance is greater than 5.

There is a fifth design decision related to the time response of the technique. Usability rules [13] establish that 10 seconds is (as an average) the time limit that users spend waiting for a webpage to be loaded. Therefore,

5. The maximum time spent to retrieve and show the information is 10 seconds.

The time used to show the retrieved information is constant, but the time used to load a webpage is variable. Therefore, the technique uses a mechanism to iteratively load webpages in relevance order and extract information from them. When the time spent is close to the limit of 10 seconds, the technique must stop the analysis and show the results.

Algorithm 2 summarizes the technique for information retrieval of interconnected webpages. It uses the following functions that implement the ideas and equations explained in this paper:

timeout(): This function controls that the algorithm never runs more than 10 seconds. When the time is over, it returns *True.*

getSlice(): It computes a slice of a webpage with Algorithm 1.

show(): This function shows in the browser a collection of DOM nodes. It should be implemented in a way that visualization is incremental.

getLinks(): It extracts the link nodes of a set of nodes.

getMostRelevantLink(): It computes the ontological distance of a set of nodes to determine what is the most relevant node.

load(): Loads a webpage.

Algorithm 2: Information retrieval from multiple webpages

Input: A set of interconnected wegpages with an initial webpage *P,* and a query *q*

Output: A webpage *P'*

Initialization: currentPage=P
pendingLinks=∅

while not(*timeout*())
(1) relevantNodes = *getSlice*(currentPage,q)
(2) *show*(relevantNodes)
(3) pendingLinks = pendingLinks ∪*getLinks*(relevantNodes)
(4) link = *getMostRelevantLink*(pendingLinks)
(5) pendingLinks = pendingLinks / link
(6) currentPage = *load*(link)

return *P'* (it is incrementally shown by the *show* function)

C. *Visualization of the Relevant Information.*

Algorithm 2 is able to collect all the DOM nodes of webpages that are relevant. Moreover, for each page, we know that the slice extracted is a valid webpage as shown in Section III-A. In addition, the information extracted is semantically related via hyperlinks and the semantic relation is weighted with the computed relevance for each DOM node. Therefore, it is possible to use standard techniques for hierarchical visualization of the retrieved information.

In our implementation reconstructing DOM trees is possible thanks to the DOM API's command:

```
documentNew.appendChildNode(documentOld.getElementById('myID'))
```

The command *documentOld.getElementById* allows us to extract from a DOM tree a specific element with a particular identifier *myID*. Then, the properties of this node can be queried, and if necessary, it can be inserted into another DOM tree with the command *documentNew.appendChildNode*.

According to Algorithm 2, the visualization of the final webpage is done incrementally. For each analyzed webpage, we extract the slice with Algorithm 1, and then, this slice is inserted into the current webpage. Then, the webpage is refreshed, and thus, the technique produces results from the very beginning and, while the user inspects them, new results are appended to the results webpage. In Example 4 we show a complete example that shows the complete process of information retrieval.

Example 4: Consider again the initial webpage P1 and its search hyperspace of Figure 9. Initially, Algorithm 2 extracts the slice of webpage P1. This slice is constructed from two key nodes (*K1* and *K2*). Then, this information is shown to the user in a new webpage. Next, the algorithm tries to find the most relevant link to retrieve information from related

webpages. According to Table 1, the most relevant hyperlinks are *H1* and *H4*. But *H4* is discarded because it points to the initial webpage that has been already processed. Therefore, hyperlink 1 is selected and webpage P2 is loaded, processed and its slice shown to the user. Observe in Figure 10 that the information of webpage P2 is shown immediately after the information of *K1*, because, when this information is added to the webpage, it is placed close to the nodes that pointed to it. Hyperlink 2 is then discarded because it points to a webpage that has been already processed (P2). Table 1 says that hyperlink 3 is more relevant that hyperlink 5. Therefore, hyperlink 3 is selected and webpage P3 is loaded, processed and shown to the user. Finally, hyperlink 5 is discarded because webpage P3 has been already processed.

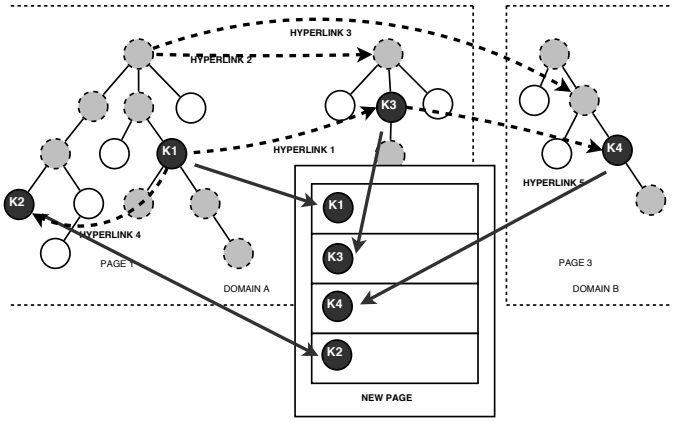

Fig. 10. Relevant Information retrieved and composed into a new page.

IV. Implementation

Our current implementation has been integrated in version 1.5 of the Firefox's *Web Filtering Toolbar*. This tool is an official extension of the Firefox web browser that has been tested and approved by Firefox's developers experts area, and that has more than 10.000 downloads at the time of writing these lines.

The most stable version of the tool can be downloaded from Firefox's addons area:

https://addons.mozilla.org/es-ES/firefox/addon/5823

The last version binaries, the development version and source code of the tool are publicly available at:

http://users.dsic.upv.es/~jsilva/webfiltering/

The examples presented in this paper and all the screenshots shown are real slices produced by the web filtering toolbar.

V. Conclusions and Future Work

This work introduces a novel information retrieval technique based on syntax distances. The technique is able to work online and extract information from websites without any pre-compilation, labeling, or indexing of the webpages to be analyzed. We are now analyzing the impact of a lexicon. Using synonyms and semantic relations will allow us to increase the precision of our algorithms, but the efficiency of the technique will be affected. Empirical experimentation is needed to decide whether it is better to analyze many webpages without the use of a lexicon or few webpages with a lexicon. A balance between amount of information retrieved and the quality of this information must be studied.

Currently, we give the same semantic value to all the kinds of nodes in a DOM tree. Future work will include the analysis of what nodes provide information that can be considered as more reliable. For instance, meta-tags and microformats are good candidates.

Another source of information that is not being currently exploited are the webpages that point to the initial webpage or that are ontologically related to it but they are not reachable from it. Precisely, because they are not reachable from the links of the initial webpage they are never processed. In this respect, using the power of current search engines can be a fast and reliable solution. For instance, Goggle's Query option *"link:"* allows us to find webpages that point to a given webpage. Similarly, Goggle's Query option *"related:"* searches for webpages ontologically related to other webpage.

References

[1] J.M. Gómez Hidalgo, F. Carrero García, E. Puertas Sanz. *Named Entity Recognition for Web Content Filtering* International Conference on Applications of Natural Language, NLDB2005, pages 286-297, 2005

[2] W3C Consortium, Resource Description Framework (RDF). www.w3.org/RDF

[3] W3C Consortium, Web Ontology Language (OWL). www.w3.oeg/2001/sw/wiki/OWL

[4] Microformats.org. The Official Microformats Site. http://microformats.org/, 2009.

[5] R. Khare, T. Çelik Microformats: a Pragmatic Path to the Semantic Web. Proceedings of the 15h International Conference on World Wide Web. International World Wide Web Conference. Poster Sessions pages 865-866, 2006.

[6] R. Khare. Microformats: The Next (Small) Thing on the Semantic Web? *IEEE Internet Computing,* 10(1):68–75, 2006.

[7] Suhit Gupta, Gail E. Kaiser et al. *Automating Content Extraction of HTML Documents*. World Wide Archive vol.8 issue.2, pages 179-224, 2005.

[8] Po-Ching Li, Mind-Dao Liu, Ying-Dar Lin, Yuang-Cheng Lai *Accelerating Web Content Filtering by the Early Decision Algorithm.* IEICE – Transactions on Information and Systems vol. E91-D, pages 251-257, 2008.

[9] W3C Consortium, Document Object Model (DOM). www.w3.org/DOM

[10] J. Silva, *Information Filtering and Information Retrieval with the Web Filtering Toolbar*. Electronic Notes in Theoretical Computer Science, vol. 235, pages 125-136, 2008.

[11] R. Baeza-Yates, C. Castillo, *Crawling the Infinite Web: Five levels are enough.* WAW, Lecture Notes in Computer Science, vol.3243, pages 156-167. Ed. Springer 2004.

[12] A. Micarelli, F. Gasparetti, *Adaptative Focused Crawling.* The Adaptative Web, pages 231-262, 2007.

[13] Jakob Nielsen. *"Designing Web Usability: The Practice of Simplicity"*; New Riders Publishing, Indianapolis ISBN 1-56205-810-X; 2010.