

## **Extracting and Searching Useful Information Available on Web FAQs**

**Edson Oliveira , Altigran S. da Silva ,  
Edleno Silva de Moura , João M. B. Cavalcanti**

<sup>1</sup>Departamento de Ciência da Computação  
Universidade Federal do Amazonas  
Manaus AM

{ecco,alti,edleno,john}@dcc.ufam.edu.br

**Abstract.** *This paper presents new methods for structuring and searching for information stored on Web FAQs. These methods are based on the assumption that such pages are implicitly organized as a set of question-answer pairs (QAPs). The ultimate goal is to improve the retrieval of answers available in FAQs for queries that can be answered using the information contained in them. More specifically, we propose modifications for three of the main tasks performed by search engines: crawling, indexing and query processing. To evaluate our proposed methods, we used a large collection of documents from a real search engine. We present the results of this evaluation for each of the three tasks.*

**Keywords:** Web FAQs, Web Search, Data Extraction

### **1. Introduction**

Users of the World Wide Web, from its very early years to the present days, have been building and publishing semi-structured documents specialized in organizing knowledge from certain domains in the form of descriptions of common cases. For this purpose, very informal document formats such as *FAQ*, *HOWTO*, *Blog* and *Forum* have been spontaneously defined by communities of users and are now widely adopted by persons and institutions of all kinds.

Out of all these types of documents, FAQs are possibly the best known example. FAQ Web pages group together answers and solutions to most frequently asked questions by users on a given subject. People usually create FAQs to help new users of a community find quick solutions to common problems members might have. This also frees more experienced members from having to answer the same questions several times.

Although there are many FAQs on a broad range of subjects available on the Web, such information is distributed in different Web pages and seem to follow no standard structure. Since each FAQ has specific knowledge about a specific subject, the potential knowledge in the whole set of FAQs available on the Web is tremendous. However, such information is probably underused given that it is spread over a large number of resources following no standard structure.

Here we are interested in creating a database containing all the information available on FAQs using an unified structure for representing such information and mechanisms for automatically updating the database with new records. Such database could be

extremely useful for web users. It could be used, for instance, for creating new intelligent systems for processing knowledge or as a source of information for Question and Answering Systems [Burke et al. 1997, Harabagiu et al. 2000, Katz et al. 2003, Lin 2002, Lin and Katz 2003].

In this paper we propose a set of methods for automatically detecting and structuring FAQs on the Web. All these methods are based on the assumption that FAQ Web pages are implicitly organized as sets of *question-answer pairs (QAP)*. The ultimate goal is to provide a way for crawling and structuring FAQs available on the Web. We also show an example of how the structure of our FAQ database can be used for improving the retrieval of answers. More specifically, we propose modifications for three of the main tasks performed by search engine, crawling, indexing and query processing, adapting them for the problem of creating and processing information available on web FAQs.

For the crawling task, we propose heuristics to select among the visited pages those that are likely to contain a FAQ. For indexing and structuring the information available on these pages, we propose an algorithm for identifying and extracting QAPs, so that they can be used in the query processing. Finally, an adaptation of the traditional vector space model is also proposed for query processing.

To evaluate our proposed methods, we have used a collection of documents from a real search engine that has over 12 million Web pages. We present the results of this evaluation for each of the three tasks. First, we show that our FAQ selection heuristics were able to find 97% of the pages containing FAQs from the Web page database. Next, we report the results obtained using our algorithm for extracting QAPs from these pages. This algorithm was able to correctly extract questions and answers in nearly 80% of the cases. We also present an example of how the structured information obtained can be applied to improve search results on a FAQ database. In experiments with our proposed query processing strategy we achieved a significant improvement in the final ranking quality.

The paper is organized as follows: Section 2 discusses related work. Section 3 presents our heuristics for identifying Web pages that contain FAQs. Section 4 presents an algorithm for extracting QAPs from FAQs. Section 5 presents our proposed adaptation to the vector space model for retrieving solutions for questions given as queries. Finally, Section 6 presents conclusions and final remarks.

## **2. Related Work**

Several recent works in the literature have addressed the problem of extracting valuable information from semi-structured Web pages. A brief survey of these works is presented in [Laender et al. 2002b].

For this problem, a variety of techniques have been deployed, such as machine learning [Hsu and Dung 1998, Kushmerick 2000, Muslea et al. 2001], tree structure analysis [Crescenzi et al. 2001, de Castro Reis et al. 2004, Liu et al. 2000], database modeling [Laender et al. 2002a] and ontologies [Embley et al. 1999]. In all cases the ultimate goal of these works is to develop methods and tools to generate wrappers to extract data from a set of data-rich Web pages. All of them rely on samplings or examples provided by users that are generalized to build extraction patterns.

In tools such as DEByE [Laender et al. 2002a], WIEN [Kushmerick 2000], and

STALKER [Muslea et al. 2001], examples must be provided of the data items to be extracted, e.g., data on books in an electronic bookstore. Other tools such as RoadRunner [Crescenzi et al. 2001] and Nodes [de Castro Reis et al. 2004] required samples of the pages containing data of interest to be supplied. The Ontos tool [Embley et al. 1999] is based on specialist-crafted ontologies. Once an ontology is built for a certain domain, e.g., car adds, it can be used for several Web sites on that domain.

In comparison with the problem addressed by these methods and tools, the problem we addressed here has an important particularity: the FAQs from which we extract QAP are usually handmade by humans using text or HTML editors. In contrast, those generic methods are targeted to data-rich Web sites that are generated from databases using templates. Moreover, by basing our method on specific FAQ features, we were able to devise a single algorithm to automatically extract QAP from several distinct Web sites, while those generic methods usually require some form of sampling to take place before processing each Web site.

The idea of using the Web as a source of information for answering questions is not new. *Question Answering* (QA) systems have been deployed for finding answers on the Web to questions posed as queries [Kwok et al. 2001]. This usually involves searching on a very large document base of a general domain application. *AskJeeves*<sup>1</sup> and *Start*<sup>2</sup> are two examples of commercial search engines which deploy the techniques related to this area.

The approach used by early QA systems combined information retrieval (IR) and natural language processing (NLP) techniques [Harabagiu et al. 2000]. The main idea is to present an answer to a question based on the documents returned by a conventional search engine. The first step involves processing the query using NLP techniques in order to understand what the user wants to know. After the search process performed by a conventional search engine, the returned documents are also processed by NLP techniques trying to find evidences which might indicate the desired answer.

An example of a QA system base on FAQs is *FAQ Finder* [Burke et al. 1997]. This system uses NLP techniques to “understand” the meaning of a query posed by users and try to match this query against a set of questions from FAQs. Then, the system produces as a result a set of possible answers to the query.

In addition to NLP, several other approaches have been proposed for the implementation of QA systems [Katz et al. 2003, Lin 2002, Lin and Katz 2003]. In all cases, these systems and approaches can benefit from the work we present here, since our methods can be used to automatically build and maintain structured FAQ databases with questions and answers extracted from the Web.

Another approach adopted for searching answers to specific questions in domain specific applications is *Case-Based Reasoning* (CBR) [Bartsch-Spörl et al. 1999]. CBR techniques focus on reusing knowledge acquired from solutions given to previous problems. Problem-solution pairs are stored in a semi-structured knowledge base that can be used to solve similar problems.

---

<sup>1</sup><http://www.ask.com>

<sup>2</sup><http://www.ai.mit.edu/projects/infolab/>

When the knowledge base is organized as a non-structured document collection, the problem is usually named *textual CBR* [Bartsch-Spörl et al. 1999, Lenz et al. 1998]. In this case documents are considered as cases and queries are the problems to be solved.

CBR techniques assume a specific domain [Bartsch-Spörl et al. 1999], therefore allowing the reuse of knowledge obtained from previous solutions to solve and improve solutions for new problems. Specific terms related to the problem domain are identified and their occurrence in documents is exploited during the searching process. In order to determine the similarity between cases, it is also assumed that problems with similar descriptions have similar solutions.

The *FALLQ* project [Lenz and Burkhard 1997] is a CBR tool for finding documents related to a query as answers in FAQs, in the domain of technical support of a telecommunication company.

Although the ideas here presented aim at producing general subject FAQ databases, they can also be applied to address FAQs available on specific domain Web portions. Thus, our methods are also able to produce inputs for CBR systems.

### **3. Identifying FAQs**

The first step to allowing a search engine to deal with FAQs is to identify the Web pages that contain them. Here we present two simple heuristics for this identification step that can be easily adapted to a typical search engine crawler. We also present the results obtained when testing these heuristic over a large collection of real Web pages.

First, we investigate the main characteristics of FAQs and of the Web pages containing them. For this purpose we gathered a small sample of 50 FAQs. This sample was obtained with the first 50 FAQs returned as a result of a query submitted to a real popular search engine using the term “FAQ”. After a thorough analysis of the returned documents, the following observations can be formulated about FAQ Web pages:

1. All documents have the string “faq” in the HTML title or in the URL of the page;
2. Over 70% of the pages describe the FAQ subject in the HTML title;
3. Over 95% of the pages present specific HTML format features for the questions and answers text;
4. All questions correspond to single paragraphs in the page;
5. Around 90% of the questions are sentences ending in question mark;
  - (a) many questions begin with some form of enumeration, such as 1.), A), i-, etc;
  - (b) most questions ending in question mark have an interrogative term such as “what”, “when”, “who”, “where”, “how”, etc;
  - (c) some pages present questions which are not related to a FAQ, as for example “forgot your password ?”
6. Around 60% of the pages have a question index, which include links to the answers.
  - (a) most links are internal (to the page) and the questions are usually repeated near the corresponding answers;
  - (b) in some FAQs, the links point to other HTML pages;
  - (c) many FAQs have both internal and external links.

Given those observations, we formulated two heuristics to determine when a page can be considered as a FAQ.

**Heuristic 1** *In order to be considered as a FAQ, a page must have the string “faq” either in the content of the tag TITLE or in its URL.*

**Heuristic 2** *In order to be considered as a FAQ, a page must have a set of one or more questions in its textual content.*

As an example, Figure 1 presents two distinct portions of a same Web page containing a FAQ. The first portion (a) corresponds to a question index which, according to our observations, is quite common in FAQs, and the second portion (b) corresponds to the page region containing the questions along with its answers. This page satisfies the conditions stated in both Heuristics 1 and 2. The string “faq” appears in its URL, circled in Figure 1. Furthermore, there are several questions identifiable in the page’s text.

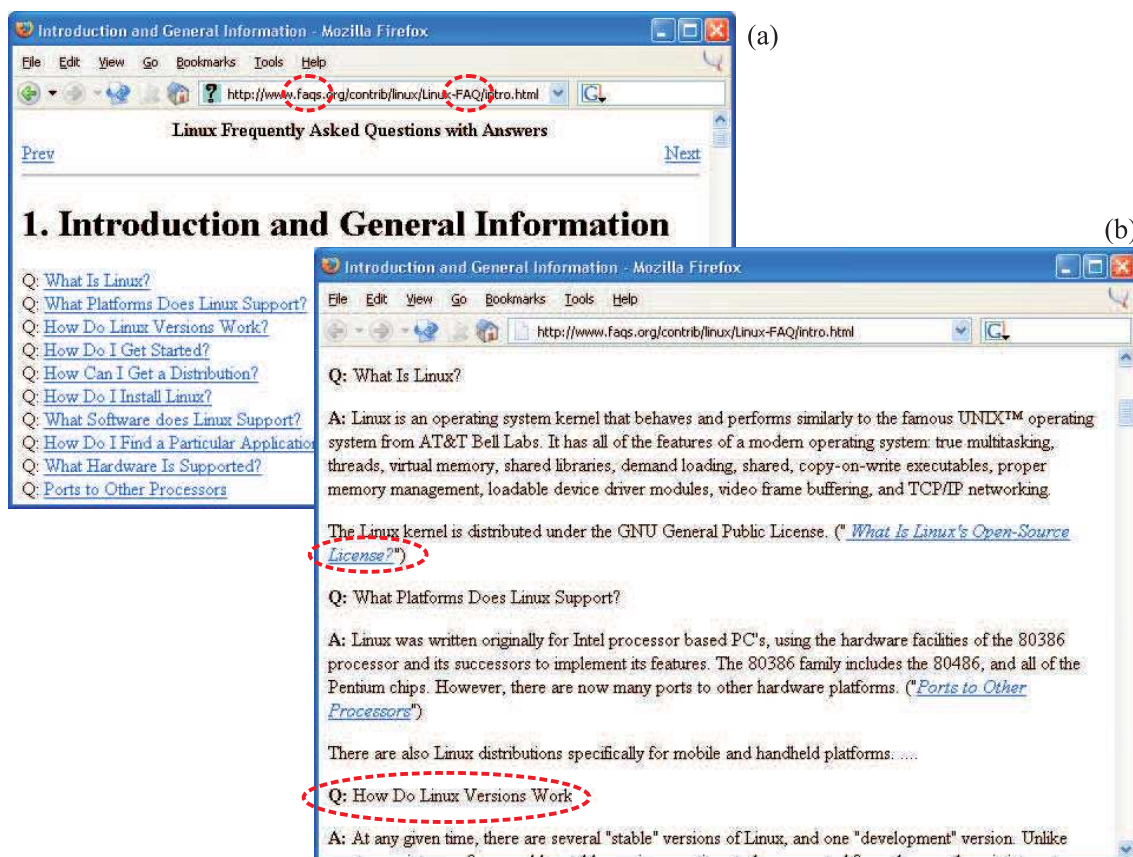


Figure 1. Example of Web page containing a FAQ.

In order to implement Heuristic 1, finding the word “faq” in the title or in the URL of a HTML page involves a simple string search operation. However, the implementation of Heuristic 2 consists in finding questions in a page. This is a more complicated operation because it requires delimiting paragraphs which end with a question mark within the HTML page.

As HTML pages are designed to facilitate its visualization, it is common to find HTML tags within the text. In some situations these tags are used to emphasize a word

or part of a text (e.g. tags B and I). Thus, by simply removing these tags we could reconstitute the paragraphs with plain text. However, other tags change the position of a text to other locations in a page. For instance, the tag H is used to format the header of HTML pages and the header text usually does not have any punctuation sign. If this sort of tags are removed, the header text would be wrongly concatenated to the body text. Given that the header does not have any punctuation the whole text would be considered as one single paragraph. For that reason we have devised a strategy for identifying paragraphs in HTML pages.

Our strategy for separating a page text in distinct paragraphs considers a subset of the W3C<sup>3</sup> standard HTML tag set of paragraph separators. This subset was defined after inspecting the pages in our sample. These tags, which in some way modify the text positioning on the page presented by a browser, we call *structural tags*. They are presented in Table 1.

Tag Type	Tag list
Basic	HTML, BODY, P, BR, HR
Heading	H1, H2, H3, H4, H5, H6
Table	TABLE, THEAD, TBODY, TFOOT, TR, TD, TH
List	UL, OL, LI, DL, DT, DD
Other	CAPTION, DIR, MENU, CENTER

**Table 1. HTML tags we consider as structural.**

In order to avoid collecting parts of text from scripts, frames and HTML forms (we consider that these elements are very likely not to be part of a FAQ), all these elements are removed before extracting the paragraphs. Paragraphs composed of symbols or spaces only (including tabs, CR, LF, etc) are also removed. After separating the paragraphs of a page, we simply select those ending with a question mark. If a number  $k$  of questions is found, we say that the condition in Heuristic 2 is met.

### **Evaluation of the Heuristics**

In order to evaluate our heuristics for identifying Web pages containing FAQs we used a large document database from a real search engine composed by more than 12 million of pages. The results are presented in Tables 2, 3 and 4.

In Table 2 we show that 0.08% of the pages in the document database were selected as containing a FAQ based on our two heuristics. Notice that 6.90% of the documents in the database are not HTML pages and were discarded.

Table 3 shows that our simple heuristics achieved a very good precision result, since 97,57% of the pages selected were confirmed as containing a FAQ. This evaluation was made by manually inspecting each page selected by the heuristics.

To estimate the recall achieved by our heuristics we randomly selected 5,890 Web pages from our document database and inspected each of them manually. Then, we apply our heuristics over this subset. The results in Table 4 show that the two evaluations agreed, which led to an estimated recall of 100%.

<sup>3</sup><http://www.w3.org/TR/REC-html40/>

Total Docs	12,029,981
Pages Selected	9,306 (0.08%)
Non-HTML Docs	829,260 (6.90%)

**Table 2. FAQ identification results**

Pages Selected	9,306
Page FAQ confirmed	9,080
Estimated Precision	97.57%

**Table 3. Estimated precision in FAQ identification**

Total Pages	5,890
Pages Selected	7 (0.12%)
Page FAQ found	7 (0.12%)
Estimated Recall	100%

**Table 4. Estimated recall in FAQ identification**

In the next section we discuss how to extract question-answer pairs present in the pages selected according to our heuristics.

## 4. Extracting Question-Answering Pairs

In this section we detail our method for extracting question-answer pairs (QAPs) from the Web pages selected as containing FAQs by our two heuristics for FAQ identification. This method is based on other heuristics we formulated based on observations we made after analyzing the set of FAQ samples described in Section 3. These heuristics for the basis of a QAP extraction algorithm we propose and present here along with experimental results that evaluate its effectiveness. At the end of this section we discuss a simple strategy for extracting the subject of the FAQs. Determining the subject of FAQs is useful for the search method discussed in Section 4.

### 4.1. Common Features in QAP

In almost all of the sample FAQs, we found that question-answer pairs (QAP) form a sequence in which questions and answers occur interleaved. Also, questions are usually consistently formatted with some presentation features that make them easy for users to identify, such as distinct font type, color or size.

Based on this, we concluded that to extract QAPs we can begin by locating questions in the page, so that a text region densely populated with questions would indicate where QAPs are. Furthermore, once we identify each question, all the answers, except for the last one, are placed between two questions. Thus, answers can be trivially extracted once questions have been located.

In some cases, larger FAQ pages present a *question index*, which is a list where the questions of the FAQ appear contiguously and, in most cases, each of these questions have links to the actual question and its answer in the FAQ. Question indexes occur in the beginning of the FAQ page. This situation is illustrated in the Web page presented in Figure 1. Notice that for the sake of QAP extraction, we are primarily interested in locating the questions appearing in Figure 1(b).

For applying the extraction heuristics, we first consider a Web page as a sequence of paragraphs  $P = \langle p_1, p_2, \dots, p_n \rangle$ . If the web page contains a FAQ, then there is a subsequence  $L \subset P$  such that  $L = \langle q_1, a_1, q_2, a_2, \dots, q_k, a_k \rangle$ , where each pair  $\langle q_i, a_i \rangle$

corresponds to a questions and its answer, respectively. Thus, our initial goal is to find the set  $Q = \{q_1, q_2, \dots, q_k\}$  in the page and then extract the set  $\{a_1, a_2, \dots, a_k\}$  of answers.

Our strategy for locating questions consists in initially taking the paragraphs in the page's text that were identified as being questions because they ended with a question mark. This step corresponds to our first heuristic. However, since it is too rough, we apply other heuristics to refine the extraction. These heuristics are used in the QAP extraction algorithm we discuss in the following.

#### 4.2. QAP Extraction Algorithm

Our QAP extraction algorithm is presented in Figure 3. It takes as input a Web page  $W$ , in which we assume that there is a FAQ, and outputs a set of question-answer pairs. In Line 3 of this algorithm we break the page  $P$  into paragraphs, assuming the same procedure we use in Section 3. Then, following the strategy outlined above, in Line 4, we build a first approximation of the set of questions  $Q$  by selecting all paragraphs in  $p$  that end with a questions mark. In Line 5 we remove all sequences of  $k > 2$  consecutive questions in  $P$ . This accounts for the occurrence of questions index before the actual questions and answers appear in the FAQ. To exemplify this, Line 5 would filter out the questions appearing in Figure 1(a).

Up to this point,  $Q$  stores all paragraphs in the target page that end with a question mark, except for those that could have appeared in an index. Notice that not all paragraphs in  $Q$  are questions and there could be questions in the FAQ that do not end with a question mark and, thus, may not be in  $Q$ . To deal with that, we resort to a simple heuristic based on the common presentation features of the questions.

First, we consider that all questions in a FAQ share similar formatting features. Thus, we try to determine what these features are in order to (1) filter out all paragraphs in  $Q$  that do not display these features and (2) include in  $Q$  paragraphs from  $P$  that have the same features. For instance, all three questions in Figure 1(b) share similar formatting features, however, as the third one does not end with a question mark, it was not included in  $Q$ . On the other hand, the paragraph ended by "What Is Linux's Open-Source License?" was wrongly included in  $Q$ .

To determine the formatting features common to the questions in the FAQ, in Line 6 we take the underlying DOM tree  $H$  of the target page  $W$ . As we know that the text in any question  $q$  in  $Q$  is spanned into one or more leafs, we consider that the formatting features of  $q$  are determined by the *formatting tags* that encompass these nodes in  $H$ . We call this the *context* of  $q$  in  $H$ . In Table 5 we present the subset of the HTML set tags we consider as forming tags. This notion is more precisely stated as follows

Let  $p$  be a paragraph extracted from a Web page  $P$  whose DOM tree is given by  $H$ . Let  $\eta(p) = \{n_1, \dots, n_m\}$  ( $m \geq 1$ ) be a set of leaf nodes in  $H$  such that  $T(n_1) \bullet \dots \bullet T(n_m) = p$ , where  $T(n_i)$  is the text contained in the leaf  $n_i$  and  $\bullet$  denotes the string concatenation operation. Also, let  $\tau(p)$  be the deepest common ancestor of all nodes in  $\eta(p)$  and let  $\pi(p)$  the set of nodes in the path from  $\tau(p)$  to the root of  $H$ . We define the **context** of  $p$ ,  $\Pi(p)$ , as the set of forming tags found in the nodes of  $\pi(p)$ .

Figure 2 illustrates the concept of context. In this figure, we have 2 paragraph (or questions), say  $q_1$  and  $q_2$ , placed respectively in nodes labeled  $a$  and  $b$ , i.e.,  $\eta(q_1) = \{a\}$

and  $\eta(q_2) = \{b\}$ . Also, a third paragraph/questions, say  $q_3$ , spans throughout nodes labeled  $c_1, c_2$  and  $c_3$ . Thus,  $\eta(q_3) = \{c_1, c_2, c_3\}$ . This occurred because the author of the FAQ choose to format the word “new” in boldface, hence it is enclosed in a BF tag. In this case, the deepest common ancestor of nodes  $c_1, c_2$  and  $c_3$  is the node corresponding to the tag FONT. Thus, the context of the three paragraphs/questions is the set of formatting tags found in the path, from nodes FONT to the root HTML, that is,  $\Pi(q_1) = \Pi(q_2) = \Pi(q_3) = \{\text{FONT, DIV, DIV, ...}\}$ .

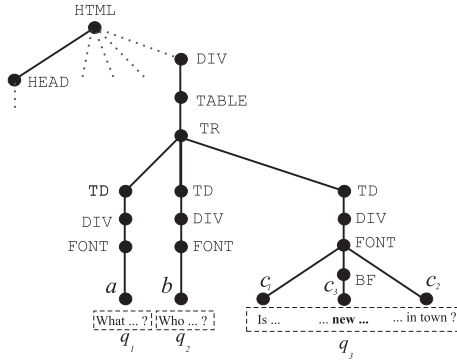


Figure 2. Illustration of three paragraph/questions common context

In Line 7 of the algorithm we use the function  $\Pi(p)$  that gives the context of paragraph  $p$  to identify the largest set  $R$  of paragraphs in  $Q$  that have a same context or, in other words, the same formatting features.  $R$  corresponds to the set of questions in the FAQ. Thus, we replace the paragraphs in  $Q$  by those in  $R$  in Line 8. This removes all paragraphs in  $Q$  that do not share a same context with the majority of other questions in  $Q$ . Next, in Line 9, we look for other paragraphs in  $P$  that have this same context and include them in  $Q$ . Finally, the loop in Lines 10–12 builds the set of QAPs.

Tag Type	Tag list
Font	FONT (SIZE, COLOR, FACE, WEIGHT), BASEFONT
Style	B, I, U, S, STRIKE, SUB, SUP, TT, PRE, BLINK
Predefine Style	BLOCKQUOTE, Q, EM, STRONG, CITE, CODE
	SAMP, KBD, VAR, BIG, SMALL, ADDRESS
Other	A, H, DIV, SPAN

Table 5. List of HTML tags we consider as formatting tags.

### 4.3. Experimental Results with QAP Extraction

We now present experimental results with the QAP extraction algorithm. For the experiments we have used the 9,306 Web pages containing FAQs that were selected from our Web page. A total of 96,651 QAP were extracted from these pages. Therefore, in average we found more than ten QAP per FAQ.

To evaluate the quality of the extraction, we manually extracted 731 QAP from 80 FAQ randomly picked out of the 9,306 FAQs. The manual extraction results were compared with the extraction results obtained with the QAP extraction algorithm. The evaluation of this comparison is presented in Tables 6 and 7, where the standard precision, recall and F measures are used. In each of these tables, we present the average of

1 **Algorithm:QAP Extraction**

**Input:** A Web page  $W$  containing a FAQ

**Output:** A set  $QA = \{\langle q_1, a_1 \rangle, \langle q_2, a_2 \rangle, \dots, \langle q_n, a_n \rangle\}$  of QAP

2 **begin**

```

3   let  $P = \{p_1, \dots, p_n\}$  be a list of paragraphs extracted from  $W$ 
   /*First approximation: All paragraphs ending with a “?” */
4    $Q \leftarrow \{p \in P \mid p \text{ ends with a “?”}\}$ 
   /*Filters out the questions in the index */
5   if there is  $I = \langle p_i, p_{i+1}, \dots, p_{i+k} \rangle \subset P$  where  $p_{i+j}$  and  $k > 2 \in Q$  then
    $Q \leftarrow Q - I$ 
6   let  $H$  be the DOM tree of page  $W$ 
   /* $R$  is subset of questions with the most frequent context */
7   let  $R = \{r_1, \dots, r_m\}$  be the largest subset of  $Q$  where
    $\Pi(r_1) \dots = \Pi(r_m)$ 
   /*Considers only the questions with a same most frequent context */
8    $Q \leftarrow R$ 
   /*Adds other paragraphs having the same context as questions */
9    $Q \leftarrow Q \cup \{p \in P - Q \mid \Pi(p) = (r_1)\}$ 
   /*Builds the set of QAPs */
10  foreach  $p_i \in P \cup Q$  do
11  |  $QA \leftarrow QA \cup \{\langle p_i, p_{i+1} \rangle\}$ 
12  end
13 end

```

Figure 3: QAP Extraction Algorithm

these measures considering the QAP in each FAQ individually, which corresponds to the row labeled “each FAQ”, and considering the whole set of QAP from all FAQ, which corresponds to the row labeled “all FAQ”.

Table 6 presents the results considering the QAP present in each page, that is, it compares the QAP manually found in each page with the QAP extracted by the algorithm. Table 7 shows a more detailed result. For each QAP extracted, it presents an evaluation of how correctly the questions and answers were extracted with respect to the terms found in the text of these questions and answers. In this table, the F measure was calculated using the average of the precision and recall from questions and answers.

Average for	Precision	Recall	F
each FAQ	88.02%	84.23%	86.08%
all FAQ	89.57%	70.26%	78.75%

Table 6. QAP extraction evaluation

Average for	Questions		Answers		F
	Precision	Recall	Precision	Recall	
each FAQ	86.90%	88.02%	82.96%	73.32%	82.65%
all FAQ	88.95%	89.57%	84.42%	83.13%	86.51%

Table 7. Questions and Answer extraction evaluation

From the figures in Tables 6 and 7 we observe that the extraction as performed by the QAP extraction algorithm achieves good quality level. Notice, however, that there is a clear difference between F measure values obtained in Tables 6 and 7 considering each FAQ (86.08%) and for all FAQ (78.75%). This can be explained by the fact that there are a few FAQs with very few questions (say, 3 or less) in our FAQ collection. In this case, the heuristic for determining a common context behaves poorly. Still, the difference between the two values is under 9 points.

As a final remark, it is worth noticing that the results in Tables 6 and 7 are similar to those obtained with site specific Web data extraction methods [Laender et al. 2002b] in terms of quality.

#### **4.4. Extraction of FAQ Subject**

To complement the information extracted from questions and answers in the FAQ, it is often useful to add contextual information provided by the FAQ subject. For instance, the search method we discuss in Section 5 takes advantage of such information, when it is available. For extracting this subject we simply take the text present in the TITLE of the page. Examining the random sample of 80 FAQs we use to evaluate the QAP extraction algorithm, we verified that in approximately 50% of them, the subject was expressed in the title of the page. Although this could be much improved using a more sophisticated method, we did not do additional investigation on this matter, since, as we show in the next section, this extraction rate was good enough for the requirements of the searching method.

### **5. Searching on FAQ Databases**

In this section we discuss how to take advantage from the structured information obtained by the QAP extraction over pages containing FAQ when developing search systems. In this section we assume that, after performing FAQ extraction, we obtain a set of tuples of the form  $\langle s, q, a \rangle$ , where  $s$  is the subject of a FAQ,  $q$  is a question and  $a$  is an answer to  $q$ . This tuples are called *SQA tuples* or simply *SQA* and the set of all SQA extracted from a set of FAQ Web pages is called a *FAQ Database*.

Contrary to document databases of ordinary Web search engines, whose elements are monolithic pages, the elements in FAQ databases are SQA, which exhibits an explicit structural information. We claim that this information can be used to derive a search system specific for retrieving information from FAQ databases. In this search system, SQA are the basic units to be retrieved and will also be named here as documents.

On the Information Retrieval field, one of the most popular models do represent documents and queries is the Vector Space Model [Baeza-Yates and Ribeiro-Neto 1999]. In this model, queries and documents are represented as vectors in a space with dimension determined by the number of distinct terms (words) of the collection of documents searched. The coordinates of each document in this model are determined by computing the weight of each term in the document. The weight formula is usually given by:

$$w_{d,t} = tf_{d,t} \times idf_t \quad (1)$$

where  $tf_{d,t}$  is a function of the number of occurrences of term  $t$  in document  $d$  and  $idf_t$  accounts the importance of term  $t$  for the whole collection. A typical way to compute

$tf_{d,t}$  is simply counting the occurrences of  $t$  in  $d$ , while the  $idf$  for a term is given by the formula  $idf_t = 1 - \frac{N}{f_t}$ , where  $f_t$  is the number of documents where term  $t$  occurs and  $N$  is the number of documents in the collection.

Given a query  $Q$ , the Vector Space Model obtains the ranking of documents by computing the similarity of each document  $d$  with respect to  $Q$ , and sorting the documents in decreasing order of similarity. The similarity formula adopted is usually the cosine of the angle between the vectors of  $d$  and  $Q$ :

$$sim(Q, d) = \frac{\vec{Q} \cdot \vec{d}}{|\vec{Q}| \cdot |\vec{d}|} \quad (2)$$

The Vector Space Model considers that there is no structure within the documents, considering them just as sets of words. However, in our FAQ database, each document is represented by a SQA tuple  $\langle s, q, a \rangle$ . Thus, a direct application of the vector space model would not consider the structure of cases, and would consider them as unstructured documents. However, given that we know each element of a case has a different meaning to the users, we decided to compute frequencies of terms in each element separately. We then experimented with different combinations of these values to check how good each field in a SQA tuple is as a source of evidence to determine the importance of a SQA to a given query. To perform these experiments, we adopted the following formula for computing  $tf$ :

$$tf_{d,t} = \alpha \times tf_{d_s,t} + \beta \times tf_{d_q,t} + \gamma \times tf_{d_a,t} \quad (3)$$

where  $tf_{d_s,t}$  is the frequency of  $t$  in the subject of  $d$ ,  $tf_{d_q,t}$  is the frequency of  $t$  in the question of  $d$  and  $tf_{d_a,t}$  is the frequency of  $t$  in the answer of  $d$ . The values  $\alpha$ ,  $\beta$  and  $\gamma$  are coefficients that represent the usefulness of each field and should be adjusted through experiments.

Since the subject of a FAQ (field  $s$ ) determines the context where each document is embedded, this field will most likely play an important role in the retrieval process. When considering the question (field  $q$ ) and answers (field  $a$ ), it is expected that the terms of  $q$  will be more similar to a user query when a document is relevant to that query, while the answer may not have a content similar to the user query, even when it is the best answer for such query. Therefore, the field  $q$  is expected to be important to determine the ranking and field  $a$  is not expected to be so useful.

We performed experiments to evaluate results achieved with the new weight scheme of Equation 3 and to select the best coefficients for each field in our SQA tuples. The experiments were conducted with the same FAQ collection we use for the experiments in Section 4. We defined 15 question queries related to topics covered in the FAQ database, and submitted these queries to our system varying the values of  $\alpha$ ,  $\beta$  and  $\gamma$ , from 0 to 10.

Given each combination of coefficients, for each query submitted we manually searched for the first result that gave an appropriate answer to the query (the first relevant answer). We then computed the mean reciprocal ranking (MRR) for each result. The MRR is a measure which computes how near is the first relevant answer from the top of the ranking, varying from 0 to 1, where 1 means that the answer is in the top and 0 means

that the answer is far from the top. MRR is usually deployed for evaluating IR systems in which the number of relevant answers is expected to be small. In scenarios such as this, which is our case, other measures such as precision are not suitable.

For the sake of simplicity, Table 8 presents only the best combinations and a combination with values (1,1,1), which is similar to the original vector space model. As it can be observed, using our proposed weight schema we were able to improve the MRR from 0.60 up to 0.76, which means the system with separate weights for each field presented relevant answers closer to the top when compared to the original Vector Space Model.

The best combination of values for the coefficients were  $\alpha = 4$ ,  $\beta = 4$  and  $\gamma = 1$ , (4, 4, 1), which indicates that the subject and the question fields are indeed more useful than the answer field. However, notice that for combination  $\alpha = 4$ ,  $\beta = 4$  and  $\gamma = 0$ , (4, 4, 0), the results were slightly worse, which means the answer field still have some positive contribution to the ranking. This conclusion can also be obtained when looking at (8, 8, 1), where again the results were worse than (4, 4, 1), meaning that the ideal combination in our experiments also took the answer as much less useful information.

The overall gain in MRR when comparing the system with the non-structured search was roughly 25%, which is a significant improvement. A final comment is that the high MRR values achieved in all cases is due to the high quality collection extracted and the separation of cases provided by the previous methods of our proposal.

Weights ( $\alpha, \beta, \gamma$ )	(1,1,1)	(4,4,1)	(4,4,0)	(8,8,1)
MRR	0.60	<b>0.76</b>	0.64	0.70

**Table 8. MRR values achieved with different weight schema.**

This section presented an example of how the structured information achieved in the FAQ extraction process may be exploited to improve the search process. This first example shows the potential benefits of extracting cases from FAQs to the search process. Other alternative ideas and models could be derived from the structure.

## 6. Conclusions

In this paper we have presented methods for automatically detecting and structuring web FAQs. The ultimate goal was to create a structured database to store all the information available on FAQs created by users on the Web. We argue that FAQs are rich sources of very specialized knowledge, but current search systems fail in properly exploiting their potential since they consider them as ordinary Web pages and since their information seems to be unstructured. Throughout the paper, we have seen that FAQs in fact exhibit a very consistent structure, that although implicit and simple, can be exploited to improved the way information in it is processed.

Based on this assumption, we have proposed a specific search engine for creating a FAQ database, modifying three of the main tasks performed by search engines: crawling, indexing and query processing.

For the crawling task, we proposed heuristics to select among the pages crawled those that are likely to contain a FAQ. Our experiments show that these heuristics, although very simple, are very effective, being able to correctly select Web FAQs in more than 97% of the cases.

We also show that instead of indexing Web FAQs as monolithic pages, we can make their inner structure explicit by extracting question-answer pairs (QAP). This way, instead of processing whole pages, QAP will be the information unit to be processed. We have presented an algorithm for QAP extraction that correctly extracted questions and their answers in nearly 80% of the cases.

Furthermore, we have presented an example of how the QAP structure could be used for improving effectiveness of a search system based on the vector space model and developed for search on the FAQ database. For processing queries over the QAP extracted, we have proposed a scheme that assigns distinct weights to questions and answers to be used with the vector space model. This scheme also incorporates the subject of the FAQ when it is available, also assigning to it a specific weight. After experimenting with several weight values, we reached a particular configuration that gave an overall gain of 25% in comparison with the traditional weight scheme used with the vector space model, achieving 0.77 in the MRR measure.

As future directions for the work here presented, we intend to study how to develop domain focused automatic detection and extraction of information available on FAQs. For this, we foresee the needing of improving our strategy for identifying the subject of the FAQs, since our current strategy for this is too shallow.

The lessons learned with the development of the methods presented here lead us to the conclusion that by exploiting the inner features of specific types of documents on the Web it is possible to improve the way they are currently processed, and assist Web users in using such types of documents more effectively. This seems a promising path for the future of research on Web information systems. In particular, as *blogs* are rapidly growing as a popular way of publishing information on the Web, we feel that some of the ideas here developed could be used for dealing with such type of documents more effectively.

### **Acknowledgments**

This work was partially supported by the Brazilian Research Council – CNPq, through projects SiteFix (grant 55.2197/02-5) and GERINDO (grant 55.2087/05-5), and by individual grants to Altigran S. da Silva (303032/ 2004-9) and to Edleno S. de Moura (303576/2004-9).

### **References**

- Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison-Wesley-Longman, 1st edition.
- Bartsch-Spörl, B., Lenz, M., and Hübner, A. (1999). Case-based reasoning: Survey and future directions. In *Proc. of XPS-99: Knowledge-Based Systems - Survey and Future Directions*, pages 67–89, Würzburg, Germany.
- Burke, R. D., Hammond, K. J., Kulyukin, V. A., Lytinen, S. L., Tomuro, N., and Schoenberg, S. (1997). Question answering from frequently asked question files: Experiences with the FAQ finder system. Technical Report TR-97-05, Computer Science Department, University of Chicago.
- Crescenzi, V., Mecca, G., and Merialdo, P. (2001). RoadRunner: Towards automatic data extraction from large Web sites. In *Proc. of the 26th International Conference on Very Large Data Bases*, pages 109–118, Rome, Italy.

- de Castro Reis, D., Golgher, P. B., da Silva, A. S., and Laender, A. H. F. (2004). Automatic web news extraction using tree edit distance. In *Proc. of the 13th international conference on World Wide Web*, pages 502–511, New York, USA.
- Embley, D. W., Campbell, D. M., Jiang, Y. S., Liddle, S. W., Lonsdale, D. W., Ng, Y.-K., Quass, D., and Smith, R. D. (1999). Conceptual-model-based data extraction from multiple-record web pages. *Data & Knowledge Engineering*, 31(3):227–251.
- Harabagiu, S. M., Pasca, M., and Maiorano, S. J. (2000). Experiments with open-domain textual question answering. In *Proc. of the 18th International Conference on Computational Linguistics*, pages 292–298, Saarbrücken, Germany.
- Hsu, C.-N. and Dung, M.-T. (1998). Generating finite-state transducer for semi-structured data extraction from the web. *Information Systems*, 23(8):521–538.
- Katz, B., Lin, J., Loreto, D., Hildebrandt, W., and Bilotti, M. (2003). Integrating web-based and corpus-based techniques for question answering. In *Proc. of the 12th Text Retrieval Conference*, pages 426–435, Gaithersburg, USA.
- Kushmerick, N. (2000). Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence*, 118(1-2):15–68.
- Kwok, C. C. T., Etzioni, O., and Weld, D. S. (2001). Scaling question answering to the web. *ACM Transactions Information Systems*, 19(3):242–262.
- Laender, A. H. F., Ribeiro-Neto, B., and da Silva, A. S. (2002a). DEByE – Data Extraction by Example. *Data and Knowledge Engineering*, 40(2):121–154.
- Laender, A. H. F., Ribeiro-Neto, B. A., da Silva, A. S., and Teixeira, J. S. (2002b). A brief survey of web data extraction tools. *SIGMOD Record*, 31(2):84–93.
- Lenz, M. and Burkhard, H.-D. (1997). CBR for document retrieval: The FALLQ project. In *2nd. International Conference on Case-Based Reasoning Research and Development*, pages 84–93, Providence, USA.
- Lenz, M., Hübner, A., and Kunze, M. (1998). Question answering with textual CBR. In *3rd. International Conference on Flexible Query Answering Systems*, pages 236–247, Roskilde, Denmark.
- Lin, J. J. (2002). The web as a resource for question answering: Perspectives and challenges. In *Proc. of the 3rd. International Conference on Language Resources and Evaluation*, Canary Islands, Spain.
- Lin, J. J. and Katz, B. (2003). Question answering from the web using knowledge annotation and knowledge mining techniques. In *Proc. of the 2003 ACM CIKM International Conference on Information and Knowledge Management*, pages 116–123, New Orleans, USA.
- Liu, L., Pu, C., and Han, W. (2000). XWRAP: An XML-enabled wrapper construction system for Web information sources. In *Proceeding of the Sixteenth International Conference on Data Engineering*, pages 611–621, San Diego, USA.
- Muslea, I., Minton, S., and Knoblock, C. (2001). Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems*, 4(1/2):93–114.