Template-Based Information Mining from HTML Documents[†]

Jane Yung-jen Hsu[‡] and **Wen-tau Yih**
{yjhsu, wtyih}@robot.csie.ntu.edu.tw
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan 106, R.O.C.

Abstract

Tools for mining information from data can create added value for the Internet. As the majority of electronic documents available over the network are in unstructured textual form, extracting useful information from a document usually involves information retrieval techniques or manual processing. This paper presents a novel approach to mining information from HTML documents using tree-structured templates. In addition to syntactic and semantic descriptions, each template is designed to capture the logical structure of a class of documents. Experiments have been conducted to extract FAQ information automatically from over one hundred HTML documents collected from the Web. Using two basic templates, the prototype FAQ Miner has accurately analyzed 65% of the collection of FAQ documents. With additional processing to handle "near-pass"es, the success rate is approximately 75%. The preliminary results have demonstrated the utility of structural templates for mining information from semi-structured text-based documents.

Introduction

With the rapidly expanding volume of unstructured natural language text available on the Internet, there is an increasing need for tools that help people retrieve the desired *information* from the web of documents. In recent years, several resource discovery tools have been introduced to help users locate information of interest from the Internet (Bowman *et al.* 1994a). Standard Web search services create and store an index of the Web as well as retrieve information from that index; meta-services provide higher quality search results by posting the queries to multiple search engines and collating the returned references (Selberg & Etzioni 1995). Relevant documents are identified and ranked in response to query posted as a set of keywords. A variety of Web agents have been designed to satisfy the informational needs of individual users (Wooldrige & Jennings 1995). Despite the dynamic nature of the Web, Etzioni has argued for the *structured Web hypothesis* that "information on the Web is sufficiently structured to facilitate effective Web mining" (Etzioni 1996).

This research explores the complementary task of extracting useful information from potentially interesting documents once they've been retrieved by the information discovery tools. Unlike structured information stored in databases that lend themselves to machine manipulations, text-based documents are primarily created for browsing or perusing by humans. The world wide web has encouraged the proliferation of a growing number of HTML documents, which are annotated with tags to format their display through a Web browser. Since the HTML specification (Raggett 1996) does not impose strict structural constraints, Web documents are only semi-structured at best. This paper argues for the *semi-structured document hypothesis*:

> *A semi-structured document with HTML tags is sufficiently structured to facilitate effective mining of semantically meaningful information.*

The objective of information mining agents is to transform documents from *data* composed of words into *information* presented by words. The extracted information has added values in supporting decision-making or deriving new information. As a result, electronic documents become not only machine *readable* but also machine *usable*.

As a first step toward achieving the above goal, this paper introduces the concept of *tree-structured templates*. Each template specifies the structural components of a class of documents to be captured. In what follows, we will first formulate the information extraction problem as template matching. The problem is then specialized to mining question-answer pairs from FAQ documents. Over one hundred HTML FAQ doc-

[‡]This research was supported in part by the National Science Council of ROC under grant NSC 86-2745-E-002-007.

uments were collected and tested. The experimental results are presented followed by the conclusion.

Template-based Information Extraction

Traditional information extraction techniques are designed to identify data entities, attributes, and relationships in full text. For example, a financial investor may be interested in extracting information on new product release by a specific company, its stock price and trading volume, as well as current market indices from news articles. Most existing tools are application-specific, and considerable efforts are required in order to build new applications.

Information extraction tasks vary widely in difficulty. For machine-generated documents with well-defined structures, e.g. searchable product catalogs, it is possible to extract useful information by learning their descriptions (Doorenbos, Etzioni, & Weld 1997). For manually edited documents with ill-defined structure, one often resorts to standard information retrieval techniques using keywords and terms (Salton 1989) or even manual processing. The Harvest system (Bowman *et al.* 1994b) relies on models of semi-structured documents to extract very specific information, e.g. the author and title from a LaTeXdocument. Document structure analysis has also been used to generate abstracts from documents (Sumita, Miike, & Chino 1995). While previous work has focused on *digitized* (e.g. scanned) images of hardcopy documents (Tang, Yan, & Suen 1994), the FAQ Miner aims to to extract information from text-based electronic documents.

Every document has a *logical structure*, well-defined or not. A fundamental assumption underlying the proposed approach is that a class of "similar" documents share a common logical structure that can be specified as a *document template*. Although each HTML document corresponds to a *document tree* by definition, it is at a lower level than the template. As a result, effective information mining depends critically on the system's ability to match parts of a document with component structures in a document template.

Electronic Documents

There are three key ingredients in a document: *structure*, *content* and *format*. For example, the logical structure of this paper consists of the title, author information, an abstract, multiple sections, followed by a bibliography. The content of this paper consists of all the symbols (e.g. words) and their relative ordering as appeared in the paper. The format of this paper consists of the layout, font, and size specifications that define the visual presentation of the document.

In general, information contained in a document is primarily defined by its content. However, the three document ingredients interrelate in subtle ways. The same sequence of words appearing in different structural parts of a document may mean different things. For instance, the phrase "Template-Based Information Mining from HTML Documents" denotes *the title of this paper*. Had it appeared in the bibliography section, the same phrase could be a *reference* to a previously published document. It is often possible to alter the format of a document without changing either its structure or content, e.g. from single-column to two-column style. On the other hand, formatting information can often give clues for the document structure, e.g. a centered, bold-faced phrase may indicate the title of a paper, or the heading of a section.

Definition 1 *An* electronic document $\mathcal{D}$ *is specified as a 5-tuple* $\langle \mathcal{S}, \mathcal{C}, \mathcal{F}, \prec, \phi \rangle$, *where*

- $\mathcal{S}$ *is a set of structural components;*
- $\mathcal{C}$ *is the sequence of content symbols;*
- $\mathcal{F}$ *is the sequence of formatting symbols;*
- $\prec$ *is a partial order over* $\mathcal{S}$*;*
- ϕ *is a mapping from* $\mathcal{S}$ *to* $\pi(\mathcal{C})$*, which is the set of subsequences in* $\mathcal{C}$*.*

The structural components $\mathcal{S}$ with the partial order $\prec$ define the logical document structure. The sequence $\mathcal{C} = c_1 c_2 \ldots c_n$ preserves the ordering of content symbols c_i as appeared in the document. The sequence $\mathcal{F}$ can be obtained by replacing every content symbol in the document with a *dummy* symbol that does not appear anywhere in the original document. For an HTML document, the sequence $\mathcal{F}$ identifies the type and location of every HTML tag in the document. The *partial order* $\prec$ arranges elements in $\mathcal{S}$ into a tree structure. A structural component s' is said to be a child node of s if and only if $s \prec s'$. The mapping ϕ, which is usually not explicitly defined, should satisfy the following properties.

Property 1 *For any* $c_i, c_j, c_k \in \mathcal{C}$ *and some* $s \in \mathcal{S}$*, if* $i \leq j \leq k$ *and* $c_i, c_k \in \phi(s)$*, then*

$$c_j \in \phi(s).$$

That is, the function ϕ maps elements in $\mathcal{S}$ into *contiguous subsequences* in $\mathcal{C}$. In addition, the children of any node are ordered from left to right according to the ordering of their corresponding contents in $\mathcal{C}$.

Property 2 *Given structural elements* $s, s_l, s_m \in \mathcal{S}$*, such that* $s \prec s_l$ *and* $s \prec s_m$*. Node* s_l *is to the left of node* s_m *if and only if* $i \leq j$ *for any* $c_i \in s_l$ *and* $c_j \in s_m$*.*

The function ϕ maps a parent node into the union of contents defined by its child nodes.

Property 3 *Given any structural components $s, s' \in \mathcal{S}$ such that $s \prec s'$, we have*

$$\phi(s') \subseteq \phi(s).$$

It follows that there is a well-founded ordering $<$ on the elements in $\mathcal{S}$ from left to right and from top to down.

Property 4 *Given any structural components $s_1, s_2 \in \mathcal{S}$. If $s_1 < s_2$, then we have*

$$\forall c_i \in \phi(s_1) \ \nexists c_j \in \phi(s_2) \text{ such that } j > i.$$

In other words, the function ϕ preserves the ordering of content symbols in $\mathcal{C}$.

Problem Formulation

The problem of information extraction from tree-structured documents can be formulated as follows.

Definition 2 *Given a set $\mathcal{T}$ of information extraction targets and an electronic document $\mathcal{D} = \langle \mathcal{S}, \mathcal{C}, \mathcal{F}, \prec, \phi \rangle$, the problem of information extraction is to find a proper assignment*

$$\mathcal{E} : \mathcal{T} \rightarrow \pi(\mathcal{C})$$

such that the mapping is consistent with the document tree structure.

Each extraction target should map into a subsequence of text that corresponds to consecutive nodes or subtrees at the same level in the document tree. In particular, when $\mathcal{T} \subseteq \mathcal{S}$, it is desirable to find $\mathcal{E}$ such that

$$\forall s \in \mathcal{T}, \mathcal{E}(s) = \phi(s).$$

Without loss of generality, we define each extraction target to be a structural component with specific syntactic and semantic features.

Definition 3 *A* structural component *is specified as a 4-tuple (Σ, N, s_0, R), where*

- *Σ is a set of terminal symbols including*
  1. *structure tags*
  2. *format tags*
  3. *semantic terms*
- *N is a set of nonterminal symbols;*
- *$s_0 \in N$ is the start symbol;*
- *R is a finite set of rules of the form $N \rightarrow (\Sigma \cup N)^*$.*

For example, in an HTML document, the `<address>` tag specifies a logical structure while the `<i>` tag specifies the format for the succeeding text. Each *semantic term* can be further defined by a set of phrases with similar meanings, such as synonyms or broad/narrow terms defined by a thesaurus. The start symbol s_0 is referred to as the *name* of the structural component. A structural component s is said to *match* a (finite) sequence of content symbols if the text can be parsed successfully by the rules R defining s.

Document Templates

Intuitively, a document template defines the logical structure of a class of similar documents. Information can be extracted from a document by filtering it through a collection of candidate templates. By organizing the target structural components, a template can capture the expected structure as well as syntactic and semantic features. A formal definition is given below.

Definition 4 *A* tree-structured document template *is specified as a pair $\tau = (\sigma, e)$, where*

1. *σ is a set of structural components;*
2. *e is a context-free expression over $\sigma \cup \{\diamond\}$.*

The symbol $\diamond$ denotes a special structural component that is assumed to match arbitrary segments of text. The context-free expression e corresponds to a tree structure that defines the relationship between the structural components in the template. Such a document template can encode specific bias from its designer. Generally speaking, the extraction target is a subset of σ that are used to identify the *significant* contents of a matching document, while anything matching $\diamond$ will be filtered out. For example, The generic template $(\emptyset, \diamond)$ can match any document without extracting useful information.

The mapping from a document back to its defining ingredients is not unique. In other words, two identical-looking documents may be generated from very different structures and formats. It is therefore very difficult (if not impossible) to reconstruct the defining template of a document. For ease of implementation, the problem of deciding if a document matches a template has been solved by augmenting traditional parsing techniques with the ability to handle the semantic terms. Since both the document and the template are tree-structured, a more general approach is to transform the problem of *template matching* into the problem of *tree matching*. An approximate tree matching algorithm has been developed and further experiments are under way.

Information Extraction Algorithm

Given an electronic document, a collection of document templates, and a set of extraction targets, the problem of information extraction can be formulated as identifying the best-matched template for the document, and extracting "meaningful" information as defined by the targets. The algorithm is shown below.

Algorithm 1 *Template-Based Extraction*
Input:
document $\mathcal{D}$
template set Γ
target set $\mathcal{T}$
Procedure:
for each $\tau = (\sigma, e) \in \Gamma$ **do**
if *Match*($\mathcal{D}$,τ)
then for each $s \in \sigma$
if $s \in \mathcal{T}$ **then**
$\phi(s) \leftarrow Assign(s, \tau, \mathcal{D})$.
return ϕ.

There are several issues concerning good template design.

- A good template should be *effective*. The effectiveness of a template τ is defined as $\frac{N_\tau}{N}$, where N is the number of documents matching the template, and N_τ is the number of documents from which information can be accurately extracted.
- A good template should be *flexible* to support easy *reuse* with minor modifications.
- A good template should facilitate *efficient* matching.

Case Study: FAQ Documents

To verify the proposed ideas about semi-structured document processing, we have performed information extraction from a constrained set of documents called Tagged FAQ (or TFAQ), which will be defined below.

FAQ Documents

The rapid development of the Internet and WWW has attracted an increasing number of novice computer and network users. When a problem occurs, it is often possible to get answers or help by searching through the various resources available on the network. Over the years, many FAQ documents have been compiled for a wide variety of topics. Such documents provide an important source of valuable information. Unfortunately, most FAQ documents consist of natural language texts that were created and edited manually, and they do not follow any standard format or structure. Consider the body of Usenet documents called FAQs, each of which is compiled and maintained by some expert on that subject. The documents vary significantly in both content and format.

An efficient and effective way to extract answers from FAQ documents will be a time-saving tool for many computer users. There have been research projects for building tools to retrieve FAQ information, most notably the FAQ Finder (Hammond *et al.* 1995; Burke, Hammond, & Kozlovsky 1996) and Auto-FAQ (Whitehead 1994). Without manual preprocessing, each FAQ document is treated as a single piece of text indexed by keywords. Our experiments were designed to extract title and Q/A information from HTML documents automatically so that each question can be indexed and processed with a higher degree of accuracy.

In general, an FAQ document consists of a collection of questions and their corresponding answers on one specific topic of interest.

Definition 5 *A tagged frequently-asked-questions document (TFAQ) is a document that satisfies the following properties:*

1. *It belongs to the class of documents generally known as "Frequently Asked Questions" (FAQs).*
2. *It is marked up with valid HTML tags.*
3. *It consists of three meaningful structural components in sequence: the title, a table of contents (TOC), and the body of questions and answers (Q/A).*

FAQ Document Template

A well-structured FAQ document usually identifies the main idea of the document as its title, and a clearly listed table of contents. The body of Q/A is usually a list of pairs, each of which is a question followed by the corresponding answer. For an HTML document, the three parts can be defined by HTML tags as follows:

- The *title* is defined by the ⟨TITLE⟩ tag.
- The *TOC* is usually an ordered or unordered list of all the questions contained in the Q/A body. The list is tagged with ⟨OL⟩ or ⟨UL⟩, while each item is tagged by ⟨LI⟩. The pointer to the corresponding question and answer is tagged by a hyper link tag ⟨A HREF="..."⟩.
- The *question-answer pairs*, each of which contains a question statement and an answer paragraph, constitute the major contents of an FAQ document. There is one entry in the Q/A body for each question in the TOC, and it is marked with a ⟨A NAME=⟩ tag.

Formatting information in the TOC can provide useful information for segmenting a TFAQ. For example, a typical TOC begins with an indicator like "Table of Contents", "Contents", "Questions List" or "Overview", and ends with a separating line generated by ⟨HR⟩. Assuming the TOC is always between the TITLE and the Q/A body, the three parts can be easily separated once the TOC is identified.

To extract meaningful structural components and their relationship from TFAQ documents, we have defined a template called `Standard_TFAQ`. For simplicity, Figure 1 presents the template in a tree-structure description, which is a straightforward abstraction of the actual context-free grammar rules.

```
Standard_TFAQ
        ...
        title
                <TITLE>
                TERM_faq_title
                </TITLE>
        ...
        toc
                index_indicator
                        TERM_TOC_indicator
                index_body
                        (ordered_list
                                <OL>
                                list_item*
                                </OL>)
                        | (unordered_list
                                  <UL>
                                  list_item*
                                  </UL>)
        ...
        q_a_pairs
                (question_answer_paragraph)*

list_item
  <LI> Hyperlink_Anchor TERM_question </A> </LI>
```

Figure 1: The Basic Template: `Standard_TFAQ`

Following the standard notations for regular expressions, the symbol * denotes the Kleene closure and the symbol | denotes disjunction. The template specifies that the internal node 'title' has three child nodes `<TITLE>`, `TERM_faq_title`, and `</TITLE>` in sequence. The template divides TOC into 2 components: the index indicator and the index body. The former is a semantic term that identifies the beginning of the TOC, and the latter is a list pointing to the Q/A pairs. Of course, an FAQ document can include additional information such as its purpose or history of development. In the sample template, symbol "..." plays the role of ⋄ in **Definition** 4.

An Example

Given the `Standard_TFAQ` template, one can easily extract the following target information:

- `TERM_faq_title`: the title,
- `TERM_question`: each individual question, and
- `Hyperlink_Anchor`: the location of the answer for the corresponding `TERM_question`.

Figure 2 shows a sample document entitled "Apple IIGS Accelerator FAQ" [4]. Applying the `Standard_TFAQ` template, the FAQ miner will produce the output shown in Figure 3.

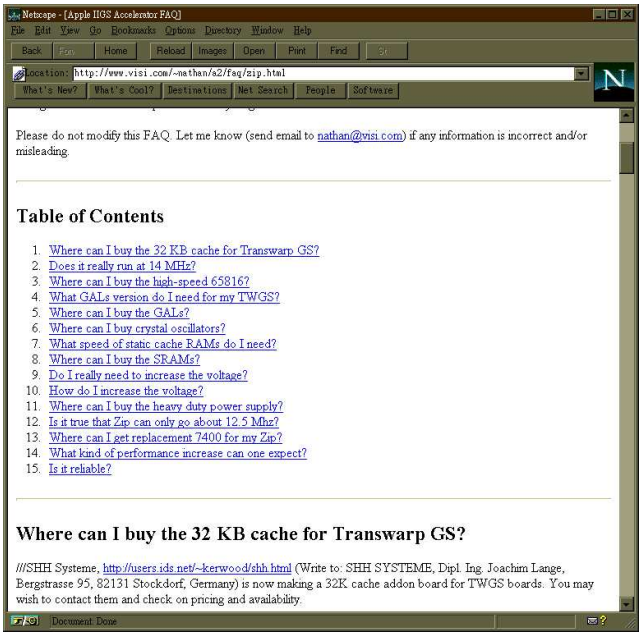

Figure 2: A Sample HTML Document

The hyper links in this sample document are named references within the same document. Once the list of questions is extracted, each question can be indexed using standard information retrieval techniques based on the contents in the corresponding Q/A pair.

Experimental Results and Discussion

Figure 4 shows the architecture of FAQ Miner, a template-based information extraction system.

In the experiment, a relatively small sample of 110 HTML FAQ documents were used as the test data. The collection of documents were assembled by first querying Infoseek using 'FAQ' as a keyword. Many of the "hits" but were not FAQ documents, and a large

[4] http://www.visi.com/∼nathan/a2/faq/zip.html

```
TITLE: Apple IIGS Accelerator FAQ

QUESTIONS:
Where can I buy the 32 KB cache for Transwarp GS?    #TWGS32
Does it really run at 14 MHz?                        #DoesRun14
Where can I buy the high-speed 65816?                #HS65816
What GALs version do I need for my TWGS?             #GALVer
Where can I buy the GALs?                            #BuyGALs
Where can I buy crystal oscillators?                 #BuyOscil
What speed of static cache RAMs do I need?           #SRAMSpeed
Where can I buy the SRAMs?                           #BuySRAM
Do I really need to increase the voltage?            #NeedVolts
How do I increase the voltage?                       #UpVolts
Where can I buy the heavy duty power supply?         #BuyPwr
Is it true that Zip can only go about 12.5 MHz?      #MaxZipSpd
Where can I get replacement 7400 for my Zip?         #Buy7400
What kind of performance increase can one expect?    #PerformUp
Is it reliable?                                      #Reliable
```

Figure 3: Sample Output

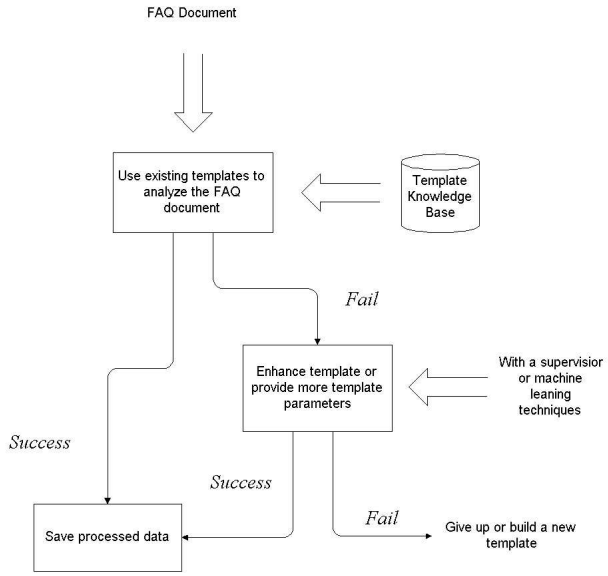

Figure 4: FAQ Miner Architecture

portion of the Usenet FAQs are in plain text. From the first 1010 pages returned from Infoseek, only 110 documents were judged to be valid TFAQs. However, as the web evolves, we expect the number of TFAQs to grow considerably in the near future.

In addition to the `Standard_TFAQ` template, a second template `No_TOC_Indicator` was designed for the experiment. It is a minor modification of `Standard_TFAQ` by removing the index indicator. Applying the two basic templates to the data we collected, the question-answer information in 72 documents (65.5%) was successfully extracted. Template `Standard_TFAQ` extracted information from 62 documents and Template `No_TOC_Indicator` extracted information from 10 documents. Table 1 summarizes the experiment.

Table 1: Summary of Results from 110 documents

Success Template/Fail	#doc	Ratio
`Standard_TFAQ`	62	56.4%
`No_TOC_Indicator`	10	9.1%
Near Pass	13	11.8%
Difficult	25	22.7%

The prototype FAQ Miner has accurately analyzed 65% of the test documents. The 38 documents that were unsuccessful can be further categorized as follows.

1. **multi-segment:** Some FAQ documents organize the table of contents into multiple levels. For example, the "World Wide Web FAQ"[5] has a table of contents for several sections, each of which points to the table of contents for that section. The context-free grammar cannot specifies templates with dynamic multi-level structures.

2. **unusual-format:** Some creative FAQ authors decorate their FAQ documents with unusual formats. For example, in the document "FAQ: Status of Recently Posted Stories"[6], the table of contents is organized using ⟨TABLE⟩, which is not predicted by the standard templates.

3. **only-qa-pair:** Not every FAQ document has a table of contents. A few FAQ documents list the question and answer pairs in a variety of formats. For example, the "The Talk.Origins Archive: Q&A List"[7] uses graphic files `q.gif` and `a.gif` to denote the locations for the questions and answers. Similarly,

[5] http://www-iso8859-5.stack.net/pages/faqs/www/

[6] http://www.cs.runet.edu/∼sratliff/FAQs/StatusOfStories.html

[7] http://earth.ics.uci.edu:8080/ origins/faqs-qa.html

"Q" and "A" or "Question" and "Answer" were used in others.

4. **toc-not-clear:** In the prototype FAQ Miner, heuristics were used in identifying the TOC. Lists with the same formats can be used for different purposes within a single document. For FAQ documents without TOC indicators, it is extremely difficult to decide if a list is a TOC without semantic understanding of its contents. Applying the standard templates on such documents often yields inaccurate information.

Although it is always possible to design very specific templates for documents in the unsuccessful categories, it is undesirable. Each template may successfully extract information from a small number of documents, so its utility will be very limited.

Conclusion

This paper presents the *semi-structured document hypothesis* that documents with HTML tags are sufficiently structured to facilitate effective mining of semantically meaningful information. A template-based information extraction approach is proposed. Our preliminary experiments have shown very promising results which supports that mining information from semi-structured Web documents is indeed feasible. One limitation of the current implementation is our use of existing parsing tools, e.g. lex and yacc, for template matching. It reduced the development efforts while putting severe limitations on semantic-based matching. Further improvements to the template specification and an approximate tree matching algorithm are under development. In addition, the ability to learn and generalize document templates will greatly enhance the utility of such a system. Similar experiments on mining information from other classes of documents, e.g. on-line news, have been planned.

References

Bowman, C. M.; Danzig, P. B.; Manber, U.; and Schwartz, M. F. 1994a. Scalable internet resource discovery: Research problems and approaches. *Communications of the ACM* 37(8):98–107.

Bowman, C. M.; Danzig, P. B.; Manver, U.; and Schwartz, M. F. 1994b. The harvest information discovery and access system. In *Proceedings of the 2nd International World Wide Web Conference*, 763–771.

Burke, R.; Hammond, K.; and Kozlovsky, J. 1996. Knowledge-based information retrieval from semi-structured text. Technical report, The Artificial Intelligence Laboratory, The University of Chicago. file://cs.uchicago.edu/pub/users/burke/bhk_fss.ps.Z.

Doorenbos, R. B.; Etzioni, O.; and Weld, D. S. 1997. A scalable comparison-shopping agent for the world-wide web. In *Proceedings of Autonomous Agents'97*.

Etzioni, O. 1996. The world-wide web: Quagmire or gold mine? *Communication of The ACM* 39(11):65–68.

Hammond, K.; Burke, R.; Martin, C.; and Lytinen, S. 1995. Faq finder: A case-based approach to knowledge navigation. In *AAAI Spring Symposium on Information Gathering in Heterogeneous, Distributed Environments*, 69–73. AAAI. file://cs.uchicago.edu/pub/users/burke/.

Raggett, D. 1996. Html 3.2 reference specification. Proposed Recommendation PR-html32-961105, W3C. http://www.w3.org/pub/WWW/TR/PR-html32-961105.

Salton, G. 1989. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Infomation by Computer*. Addison-Wesley.

Selberg, E., and Etzioni, O. 1995. Multi-service search and comparison using the metacrawler. In *Proceedings of the 1995 World Wide Web Conference*. http://www.cs.washington.edu/research/projects/softbots/www/metacrawler.ps.

Sumita, K.; Miike, S.; and Chino, T. 1995. Automatic abstract generation based on document structure analysis and its evaluation as a document retrieval presentation function. *Systems and Computers in Japan* 26(13):32–43.

Tang, Y. Y.; Yan, D. C.; and Suen, C. Y. 1994. Document processing for automatic knowledge acquisition. *IEEE Transactions on Knowledge and Data Engineering* 6(1):1–21.

Whitehead, S. D. 1994. Auto-faq: an experiment in cyberspace leveraging. In *Proceedings of the Second International WWW Conference*, 25–38. http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/Agents/whitehead/whitehead.html.

Wooldrige, M., and Jennings, N. R. 1995. Intelligent agents: Theory and practice. *Knowledge Engineering Review* 10(2):115–152.