Computer Networks 33 (2000) 699–711

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

Mining the Web for relations

Neel Sundaresan [a,*], Jeonghee Yi [b,1]

[a] *IBM Almaden Research Center, San Jose, CA, USA*
[b] *University of California, Los Angeles, CA, USA*

Abstract

The Web is a vast source of information. However, due to the disparate authorship of Web pages, this information is buried in its amorphous and chaotic structure. At the same time, with the pervasiveness of Web access, an increasing number of users is relying on Web search engines for interesting information. We are interested in identifying how pieces of information are related as they are presented on the Web. One such problem is studying patterns of occurrences of related phrases in Web documents and in identifying relationships between these phrases. We call these the duality problems of the Web. Duality problems are materialized in trying to define and identify two sets of inter-related concepts, and are solved by iteratively refining mutually dependent coarse definitions of these concepts. In this paper we define and formalize the general duality problem of relations on the Web. Duality of patterns and relationships are of importance because they allow us to define the rules of patterns and relationships iteratively through the multitude of their occurrences. Our solution includes Web crawling to iteratively refine the definition of patterns and relations. As an example we solve the problem of identifying acronyms and their expansions through patterns of occurrences of (acronym, expansion) pairs as they occur in Web pages.

Keywords: Mining; Crawling; XML; Relation; Duality

1. Introduction

The World Wide Web is a vast source of information. However, the Web consists of an ever-growing set of pages authored by people with vastly differing cultures, interests, and educational levels, while the goal of the individual Web page author is to furnish information. Web crawlers visit these Web pages and index the crawled pages to serve search engines. As these crawlers analyze these Web pages they could look for and learn interesting pieces of information which remain buried in these pages. For instance, a crawler could analyze link information in Web pages to identify how many pages point to a Web page, and how many pages a Web page points to. Based upon this information, the crawler can identify pages that are authorities on certain topics and pages that are starting points (hubs) for such authorities [3,6]. This technique can be extended to identify communities on the Web which consist of pages that point to each other in particular ways [9].

Links are only one type of relations that link entities (Web pages in this case) of the Web. There could be other kinds of relationships of a semantic nature between entities. Identifying these relationships and the patterns of occurrences of these relationships can help provide valuable information buried in the Web.

* Corresponding author. E-mail: nsundare@hotmail.com
[1] E-mail: jeonghee@cs.ucla.edu

1389-1286/00/$ – see front matter
PII: S1389-1286(00)00085-2

This will be increasingly the case as access to the Web becomes pervasive and as end users rely on the Web to look for information with expectation of reliability. For instance, one may be interested in searching the Web to find out the author of a particular book, or to find all books written by a particular author [2]. Such information is typically not easily served by search engines of today.

The vision of a semantic Web [1] includes collaborations that extend to machines that are capable of analyzing all the data on the Web for content, links, and transactions. We are not anywhere near realizing this vision yet, but we do have some loose structure in the form of text, structure, and links in HTML. We would like to exploit what is available to find interesting information.

The rest of this paper is organized as follows. Section 2 discusses duality problems in the World Wide Web. Section 3 formalizes the duality problem of patterns and relations. Section 4 extends this to higher-level duality problems. In Section 5 we solve a *2-level* duality problem of finding acronyms and their expansions in detail. Section 6 discusses the issues in mining over structures and links. Section 7 further generalizes the duality problem and formulates how to discover new relationships. In Section 8 we discuss related research work and in Section 9 we draw conclusions and discuss work in progress and future work.

2. Duality problems in the World Wide Web

Duality problems are materialized in trying to identify two sets of inter-related concepts. Consider the problem of extracting a relation of books — (author, title) pairs from the Web [2]. Intuitively, the problem can be solved as follows.

(1) Begin with a small seed set of (author, title) pairs.
(2) Find all occurrences of those pairs on the Web.
(3) Identify patterns for the citations of the books from these occurrences.
(4) Then, search the Web for these patterns to recognize more new (author, title) pairs.
(5) Repeat the steps with the new (author, title) pairs.

Here, we try to solve the problem of extracting (author, title) relations for books, by iteratively refining mutually dependent coarse definitions of these concepts.

Note that, on the Web, the duality exists in two forms: (1) one induced by static link topology, and (2) the other occurring, in the text of Web documents, in the form of *relations* and *patterns*.

The first form of duality was identified as the notion of hubs and authorities [6,8]. The second form of duality is induced by a specific type of relation of information, such as (book, author) relation or (acronym, expansion) relation, and the patterns that signify the relations. Identifying these relations and patterns can help uncover valuable information in the Web. This will be increasingly more valuable as access to the Web becomes more common and as end users rely on the Web to look for more sophisticated information. For instance, one may be interested in studying business practices of companies as they do commerce on the Web or study hobbies of people and their geographic location through their Web pages for purposes of targeted marketing.

2.1. Some duality already explored

HITS (Hyperlink-Induced Topic Search) identifies authoritative Web pages by iteratively identifying *hub* pages and *authority* pages [6,8]. A *hub* page is a Web page that points to many authority pages. An *authority* page is one that is pointed to by many hub pages. In the world of research literature hubs can be identified with survey papers, and authorities with seminal papers. HITS starts with a small set of Web pages with their hub and authority scores set equal. At each iteration, it computes a list of hub pages by updating the hub score of each page on the basis of authority scores of its immediate neighbors. Likewise, it computes a list of authority pages on the basis of hub scores of its neighbors. The iteration continues until it converges.

DIPRE (Dual Iterative Pattern Relation Expansion) is applied to extract relations of (author, title) pairs for books from the Web, as described earlier [2]. The same framework can be applied to build a directory of people, a database of products, a bibliography of academic works, and many other useful resources.

3. Duality of relations and patterns on the Web

Let $\boldsymbol{W}$ be a large database of documents such as the Web. Let $R = \{r_i \mid i = 1, \ldots, n\}$ and $P = \{p_j \mid j = 1, \ldots, m\}$ be sets of relations and patterns, respectively. A relation is a pair of interrelated concepts, such as (acronym, expansion) pairs. A pattern is the way in which relations are marked up in Web pages. r_i occurs in $\boldsymbol{W}$ at least one time with one (or possibly more) pattern(s) p_j. A pattern p_j signifies at least one (or more) relation(s) r_i.

We iteratively identify two sets R and P, starting with R_0 and P_0 the initial definitions of R and P. R_i and P_i ($i > 0$) are computed as follows:

$$R_i = R_{i-1} \cup f(P_{i-1}, \boldsymbol{W}_i),$$

$$P_i = P_{i-1} \cup g(R_{i-1}, \boldsymbol{W}_i)$$

where $\boldsymbol{W}_i$ is a subset of $\boldsymbol{W}$ that was not seen until the current iteration i. f and g are functions extracting new relations and patterns, respectively. Patterns are applied to $\boldsymbol{W}_i$ in order to find relations, and so are relations to find patterns. The computation of R_i's and P_i's is repeated until they converge towards R and P. The ultimate convergence is achieved if the following conditions are satisfied:

$$R_{i+1} = R_i \text{ and } P_{i+1} = P_i$$

after iterating the process on the domain of Web pages. The iterations themselves may be defined by full or partial steps in a Web crawling exercise.

We can think of several examples of R and P. R can be the set of (author, book) pairs and P can be the set of patterns using which pairs in R are defined [2]. Or R can stand for a set of community pages (high quality Web pages that talk about a particular topic like 'tennis' or 'fishing') and P is a set of ways they can point to each other. In the HTML world, P is a simple set of 'anchor tag' relationships using which Web page may point to another Web page.

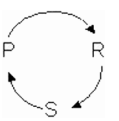
4. Higher level duality problems

It is possible to define higher-level dualities when the mutually recursive relation between R and P is through another set, say S. Here an approximation to R in a particular iteration may depend on an approximation to P in a previous iteration, which in turn may depend on an approximation to S in a previous iteration. The approximation of S itself may come from R. Thus we can define a *2-level* duality as:

$$R_i = R_{i-1} \cup f(P_{i-1}, \boldsymbol{W}_i)$$

$$P_i = P_{i-1} \cup g(S_{i-1}, \boldsymbol{W}_i)$$

$$S_i = S_{i-1} \cup h(R_{i-1}, \boldsymbol{W}_i)$$



The figure on the right side depicts the dependencies between entities.

This can be further generalized to an *n-level* duality problem. An example of a *2-level* duality problem can be found in discovering pairs of acronyms and their expansions.

5. Solving a 2-level duality problem: mining the Web for acronyms

Here we give an experiment we ran with solving the *2-level* duality problem. We apply the duality to identify acronyms and their expansions. We call the occurrences of (acronym, expansion) relations *AE-pairs*. An acronym comes from the space of words defined by the regular expression [A–Za–z0–9][A–Za–z0–9]*. An expansion is a string of words that stands for an acronym. Acronym *formation rule* is a rule which specifies how an acronym is formed from its expansion. The acronym identification problem involves two kinds of duality:

(1) one between *AE-pairs* and their patterns (1-level duality), and
(2) another between *AE-pairs*, acronym *formation rules*, and patterns (2-level duality).

The dualities are depicted in Fig. 1.

We start off with base sets of *AE-pairs*, patterns, and acronym formation rules. Using the base set of patterns, we crawl the Web to look for new *AE-pairs* that conform the patterns in the base set. From the set of *AE-pairs*, new formation rules are extracted. Moreover, we identify new patterns that associate the acronyms, in the *AE-pairs* set, with their expansions. With the extended sets of *AE-pairs*, patterns, and the rules, we continue crawling the Web in order to discover more of them.

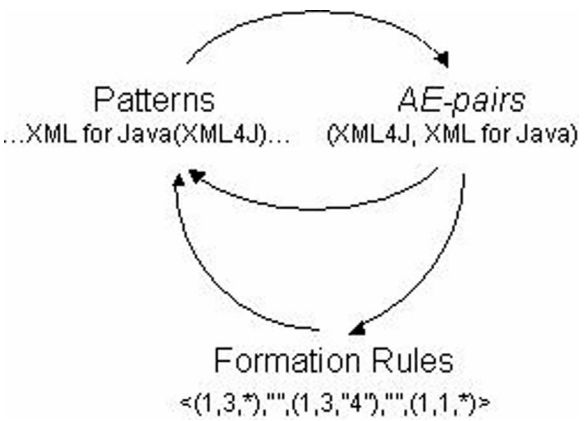


Fig. 1. Dualities for acronym mining: one between *AE-pairs* and their patterns, and another between *AE-pairs*, formation rules, and patterns.

5.1. Why acronyms?

Our work was motivated by the task of engineering a search engine targeted for XML (eXtensible Markup Language)-related information [5,14]. The search engine is available from http://www.ibm.com/xml. The **search engine** [1] is for developers and users of XML and its related technologies. It responds to queries for XML-related pages, documents, DTDs (Document Type Definition), and the like. XML is a meta-markup language and allows several domain-specific definitions of the language with their own definitions of element names and nesting structures. As a result, the crawler for the topic-specific search engine must include all Web pages that pertain to domain specific languages built on XML (e.g., MathML (Mathematical Markup Language) or CML (Chemistry Markup Language)) or XML instances. The automated and dynamic discovery of such language names and their relevant systems is a challenging problem. We notice that the domain of XML is acronym-driven. Thus automatic discovery of acronyms is a basis for discovering relevant concepts defined by the terms of acronyms. The discovered acronyms provide the domain (or scope) of mining the concepts relevant to the target topic.

[1] www.ibm.com/developer/xml

5.2. Problem definition

The acronym identification problem can be solved as a dual iterative problem where we start with a base set of *AE-pairs*, patterns, and the *formation rules* of the *AE-pairs*, in which the expansions occur in relation to the acronyms in Web pages.

We crawl the Web to look for new *AE-pairs* in the Web pages based upon the patterns described in the base set. We learn new *formation rules* from the new set of *AE-pairs*. In addition, we also identify additional patterns which associate the acronyms with their expansions in the base set of *AE-pairs*. Thus, after the first iteration, we would have identified new sets of *AE-pairs*, *formation rules*, and new sets of patterns in which the *AE-pairs* occur. Using the results in the first iteration, we continue crawling the Web to identify more *AE-pairs*, more *formation rules*, and more patterns. The goal is to identify as many *AE-pairs* and patterns of their occurrences while minimizing false identifications of acronyms, their expansions, *formation rules*, or patterns of their occurrences. We would also like to identify *good patterns* (patterns that identify the most *AE-pairs* with least false identifications) and identify *good relationships* (relationships that are identified by good patterns).

In this duality problem of identifying *AE-pairs*, it is possible to determine whether an expansion is a good expansion for an acronym. This can be automatically done through looking at the acronyms and their expansions. Since there is a reasonably well-known way how typical acronyms are built, this can be automatically done in our software system. In addition, since ours is a learning system, we define another duality problem, where we start with a base set of rules for how expansions for acronyms are provided. As we mine more acronyms and their expansions, we can also automatically refine the set of formation rules in our system.

5.3. The mining algorithm

Let W be a database of unstructured, or semi-structured documents such as the Web. Let R, P, and S be sets of target relations, patterns, and formation rules. Each relation, r (r in R), defines an *AE-pair*. An r can occur in one or more patterns. Each s in

1. initial set of *AE-pairs*: $R_0 = \{r_i \mid r_i \text{ is an } AE\text{-}pair, i = 1, \ldots, n_0\}$
   initial set of *patterns*: $P_0 = \{p_j \mid p_j \text{ signifies at least one tuple in } R_0, j = 1, \ldots m_0, m_0 \leq n_0\}$
   initial set of *formation rules*: $S_0 = \{s_k \mid k = 1, \ldots, l_0, l_0 \leq n_0\}$
2. Set $i = 1$
3. Let W_i be a new Web page (or a set of new Web pages) crawled.

   $R_i = R_{i-1} \cup f(P_{i-1}, W_i)$
   $S_i = S_{i-1} \cup h(R_{i-1})$
   $P_i = P_{i-1} \cup g(R_{i-1}, S_{i-1}, W_i)$

   where f, g, and h are relation, pattern, and formation rule extraction functions, respectively.
4. Set $i = i + 1$
5. If (steady state) stop, otherwise go to step 3.

Fig. 2. Acronym mining algorithm.

S specifies how an acronym of an expansion is determined. The acronym mining algorithm is given in Fig. 2. This is a specialized algorithm from the general duality algorithms discussed in Sections 3 and 4. The algorithm incorporates the double dualities, i.e., the 1-level duality between *AE-pairs* and their patterns, and the 2-level between *AE-pairs*, acronym *formation rules*, and patterns, into the mining.

New patterns have two ways of being identified: by either the *1-level* duality or by the *2-level* duality we identified at the beginning of this section. We combined the two cases into one-step as shown in step 3 of Fig. 2. For the mining of *formation rules*, a set of *AE-pairs* is sufficient without the lookup of Web pages.

The steady state is reached when W is repeatedly crawled and no new acronyms, patterns, or formation rules are discovered. In practice, we might not look for ultimate convergence. Alternatively, we may set a threshold on the rate of new knowledge discovered, on time, or on other resources, to determine the steady state. Since the set of Web pages is ever increasing, and typical crawlers do not terminate, this algorithm may be a constantly running process.

5.4. Acronym formation rules

An *acronym formation rule*, or, simply *formation rule*, is a rule by which an acronym is formed from its expansion. It consists of a list of *replacement rules*.

A *replacement rule* specifies how one or more characters in the acronym comes from a word in its expansion and can be encoded by a three-tuple

(*substr_bPos*, *substr_ePos*, *replacer*)

where *substr_bPos* (or *substr_ePos*) is the position of the leading (or ending) character of the substring of expansion word to be replaced. *Replacer* replaces the substring of the expansion word from *substr_bPos* to *substr_ePos* to form the acronym. If no replacement takes place, *replacer* is represented by '*'. For (*XML*, *eX*tensible *M*arkup *L*anguage) pair, for example, the replacement rule from '*Extensible*' to '*X*' is (1, 2, *X*), which indicates that the substring 'Ex' is retained in the acronym after being replaced with '*X*'.

A *formation rule* consists of a sequence of replacement rules interspersed with *intermediate*s. An *intermediate* is a substring of an expansion that is ignored in making its acronym. For the (*PICS*, Platform for Internet Content Selection) pair, for example, '*for*' is an intermediate. The formation rule of the pair is <(1,1,*), 'for', (1,1,*), "", (1,1,*), "", (1,1,*)>.

5.5. Patterns

A *pattern* is a three-tuple

(*a_pattern*, *e_pattern*, *formation_rule*)

where *a_pattern* and *e_pattern* are acronym and expansion patterns, respectively, and are composed of two types; *text patterns* and *structure patterns*. *Formation_rule* is the formation rule of its acronym.

Patterns in Web document appear in text, or are embedded in structure tags. Patterns with HTML tags require more information than those with plain text only.

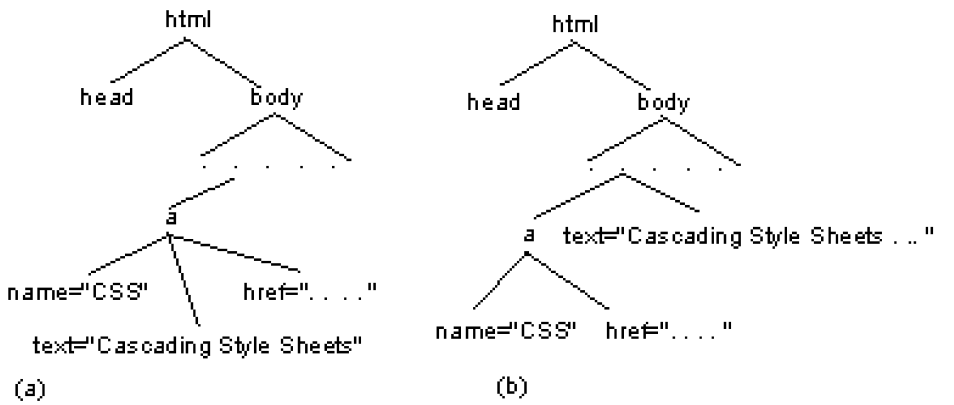


Fig. 3. HTML document hierarchies containing *AE-pair* definitions. In (a), expansion is defined at the *same* level as the acronym, such as '... Cascading Style Sheet...'; in (b), expansion is defined in the *parent* level of the acronym, such as '... Cascading Style Sheet...'.

5.5.1. Text patterns

a_pattern and *e_pattern* for flat text are defined as follows:

(1) *a_pattern* is a pair *(a_prefix, a_suffix)*, where *a_prefix* and *a_suffix* are characters surrounding an acronym;

(2) *e_pattern* is a pair *(e_prefix, e_suffix)*, where *e_prefix* and *e_suffix* are characters surrounding the expansion.

5.5.2. Structure patterns

In Web documents many *AE-pairs* are embedded in the structure of HTML, as in:

Cascading Style Sheet

This pattern is a sub-hierarchy of the entire document hierarchy. In this case only the sub-hierarchy, not the full hierarchy, needs to be stored as the pattern.

- *a_pattern* is a four-tuple: (*a_tag*, *a_attr*, *a_prefix*, *a_suffix*), where *a_tag* and *a_attr* are an HTML tag and its attribute, respectively, in which the acronym occurred. *a_prefix* and *a_suffix* are the surrounding characters of the acronym.
- *e_pattern* is a five-tuple: (*e_hier*, *e_tag*, *e_attr*, *e_prefix*, *e_suffix*), where *e_tag* and *e_attr* correspond to an HTML tag and its attribute in which the expansion occurred. *e_hier* is the relative position of the occurrence of the expansion, in the document hierarchy, in comparison to that of the acronym. *e_hier* has one of the four values [same | sibling | parent | child].

Fig. 3 shows patterns where *e_tag* is *same* and *parent*, respectively.

Note that the text pattern is the special case of the structure pattern where the values of *a_tag*, *a_attr*, *e_hier*, *e_tag*, and *e_attr* are all null.

5.6. Experiments on pattern learning

We conducted experiments on acronym learning in the context of a topic-specific search engine for XML-related information. (The search engine is available from http://www.ibm.com/xml.) The Web pages used for acronym mining were gathered by a targeted crawler [20] that crawls the Web for information related to XML. The crawler is started off with the *AE-pairs* given as seeds in Table 1.

Table 1
Initial set of *AE-pairs* and patterns provided for duality mining

(DCD, Document Content Description)	'Document Content Description (DCD)'
(CSS, Cascading Style Sheets)	'Cascading Style Sheets (CSS)'
(XML, eXtensible Markup Language)	'eXtensible Markup Language (XML)'

Unlike the conventional Web crawlers that visit all hyperlinks contained in the previously downloaded Web pages, the goal of a targeted crawler is to crawl ideally only and all the Web pages that are relevant to the given topic. In order to achieve the goal, our crawler system (1) incrementally defines the target topic, and (2) identifies and crawls the Web pages that qualify under each level of this topic definition. Typically a target topic consists of many sub-topics. In addition, it includes many other topics that are relevant because they share some property with the target topic. The system discovers the sub-topics and relevant topics using data mining techniques. It applies association mining on hyperlink metadata of the Web pages and various filtering techniques on the mined data. The system starts off with initial definition of target topic and a small set of seed pages. With the mining, the system enhances the definition incrementally by adding newly discovered topic terms, such as sub-topics and relevant topics. For example, the system is initially provided with a coarse definition of topic with a few basic terms, such as {XML, DTD}. As the crawling proceeds, the system discovers and adds new relevant topic terms to the topic definition like XSL (XML Speech Language) or JSML (Java Speech Markup Language).

The targeted crawler needs to be directed in order to avoid drifting away from the target topic. The crawler is guided on the basis of hyperlink metadata of the URLs. The system utilizes various prediction algorithms that compute the likelihood of the relevance of a Web page to the topic, without actually visiting the page, on the basis of its hyperlink metadata. That enables the system to prioritize the visiting order of URLs and thus prune irrelevant links without page lookup.

Table 2 demonstrates the progressive learning of *AE-pairs* and patterns, iteration-by-iteration, for the first five iterations. The new *AE-pairs* and patterns are listed for each iteration (columns 4 and 5). Due to space limitation, we omit listing of formation rules identified. For readability, the patterns are given as they appear in documents, rather than as in their formal definition. In our experiments we saw that learning through duality has far outperformed our initial expectation with respect to both quantity and quality of the discovery.

5.7. Good patterns and noise patterns

In Section 5.2 we defined *good patterns* as those that identify the most (acronym, expansion) pairs with least false identification, and *good relations* as the relations identified by good patterns. In our experiments, we hardly found any false identification of (acronym, expansion) relationships by any given pattern, mainly because our patterns require strict formation rules as well as well-defined prefixes and suffixes. Almost all acronyms conforming to any pattern indeed turned out to be acronyms. Therefore, we measure only the number of (acronym, expansion) relations to measure the goodness of patterns and relations.

5.7.1. Effectiveness of the acronym mining by duality

Table 3 summarizes the mining results in a nutshell. A single-threaded crawler implemented in JAVA ran on a PC with a 399 MHz Pentium II processor and 128 M RAM. The crawler downloaded and analyzed 13,628 Web pages, from which 2694 unique *AE-pairs* and 948 unique patterns were identified. A new *AE-pair* was discovered for every 5 pages, and a new pattern was discovered for every 14.5 Web pages.

In the experiment, no false identification of *AE-pairs* has occurred, except one case, achieving virtually zero error-rate, thanks to the strict pattern specification that requires strict formation rules, structure pattern, as well as well-defined prefixes and suffixes. The incorporation of formation rules specific to acronym and expansion into the general framework of patterns lifted the accuracy of the learning significantly. In general, in order to achieve high-quality learning, it is important to refine the definition of patterns specific to the target of the learning.

However, the system does discover some patterns that are not so useful to discover new relations. One reason is that our pattern is too strict, a trade-off for low error-rate of relation discovery. It is also due to the ambiguity in Web documents. For instance, from the text *XML*: *Extensible Markup Language*, the system extracts two patterns: one from only the anchor text, and another from the entire anchor tag. The latter is an unnecessary duplicate. Though the unnecessary patterns may increase computational complexity, this is not a

Table 2
New *AE-pairs* and new patterns identified by first five iterative mining process [a]

Iteration	Number of AE-Pairs used	Number of Patterns Used	New Acronyms	New patterns discovered
seeds			DCD CSS	XML Document Content Description (DCD) Cascading Style Sheets (CSS) Extensible Markup Language (XML)
1	3	1	ACSS	Cascading Style Sheets, CSS, XML: Extensible Markup Language Cascading Style Sheets W3C:Cascading Style Sheets
2	4	5	DOM, DTE, RDF, SAX, SDML, SMIL, VML, XSL	The Cascading Style Sheets (CSS) XSL - the eXtensible Stylesheet Language <cite>Document Content Description for XML</cite> Document Content Description <cite>Document Content Description</cite>
3	12	10	ANSI, ATM, BAWG, BRAN, CTM, DECT, DVB, ECBS, ECMA, EWG, FPLMTS, HLSQ, ICC, IEC, IEV, ISSS, ITPWG, PICS, PTS, TIPHON, TMWG	
4	33	10	DDML, ICE, ATHML, WIDL	<area ... href="...dcd" alt="Document Content Description...>
5	38	11	AIML, AML, CBL, CML, ICE, JSML, RDF, OMG, TML, XBEL, XLF, XLL	Document Content Description for XML (DCD) Extensible Markup Language [XML] DOM, Document Object Model CGM (Computer Graphics Metafile) "XML" The Extensible Markup Language <a ...>DOMDocument Object Model

[a] The 2nd and 3rd columns list the number of *AE-pairs* and patterns, respectively, known a-priori. The 4th column lists the new acronyms discovered in the iteration. The 5th column lists the occurrences of new patterns discovered. Starting off with 2 *AE-pairs* and 3 patterns, the duality mining discovered 1, 8, 21, 4, and 12 new *AE-pairs*, and 4, 5, 0, 1, and 6 new patterns from iteration 1 to 5, respectively.

major concern as the process is mostly communication-bound rather than computation-bound.

The next experiment compares the result sets of *AE-pairs* discovered with and without duality-based mining. For the comparison, we recrawled the same set of URLs by applying 10 a-priori acronym patterns and some variations of the conventions without duality-based mining process. Table 4 lists the result. Note that the a-priori patterns in the table are more general than our earlier definition of *pattern* in the sense that one pattern in the table may correspond to many patterns discovered by the system. Note that the sum of all *AE-pairs* discovered individually by the patterns is more than the entire *AE-pairs* discovered because of duplicates.

The result shows that the mining by duality extracts twice as many *AE-pairs* as the extraction

Table 3
AE-pairs and patterns extracted from Web documents

Number of URL downloaded	13628
Number of unique AE-pairs	2694
Number of unique patterns	948

AE-pairs are discovered at the rate of one every 5 pages, and new patterns are discovered at the rate of one every 14.5 pages.

Table 4
The coverage of a-priori patterns for AE-pairs

Pattern	Number of AE-pairs extracted by each pattern
expansion (acronym)	896
acronym (expansion)	332
acronym-expansion	98
acronym: expansion	31
expansion [acronym]	9
(acronym) expansion	6
acronym [expansion]	5
acronym, expansion,	5
(expansion) acronym	2
[acronym] expansion	1
	1385*

	Number of new AE-pairs
without duality-based mining	1033**
with duality-based mining (from Table 3)	2694***

The sum of all AE-pairs discovered (*) is greater than the entire AE-pairs discovered (**) because the same AE-pair is discovered by multiple patterns. The number of AE-pairs identified with duality-based mining (***) is more than 2.5 times of that without the mining (**).

without the mining. Moreover, we found that many acronyms are defined in unusual ways. Sometimes, they are used without explicit intention of defining it. For example,

```
<CENTER>
    <H1><font size=16>Center for Image
        Processing Research</font></H1>
    <IMG ALIGN=CENTER
        SRC="images/cipr_logo_200t.gif">
</CENTER>
```

Our system would mine the pair (CIPR, Center for Image Processing Research) from these data. This would not easily be recognizable without our mining technique.

6. Mining over structures, duality-links, and metadata

In the acronym-expansion formation we saw two kinds of patterns: text patterns and HTML structure patterns. In the text patterns we described the occurrence of acronyms and their expansions in terms of tuples of regular expressions representing character strings. In HTML structure patterns, we described patterns using element/attribute names and values and parent/child/sibling relationships. For generalized relationships between two entities we need a description language that describes how two entities are related over arbitrary tree or graph structures. The simplest way to describe such a pattern is through a pair of path expressions starting at the root or at a key node in the tree. A syntax as seen in WebL [7], XPath [17,19] or a proposed XML Query language [13] may be used for this purpose.

Other relationships could occur over link structures. These relationships may be defined by the annotations around the anchor tags in HTML. The anchor tag has attributes like ALT, HREF, NAME, ONMOUSEOVER, and TITLE. In addition there are annotations like the anchor text and the surrounding text. One can define a pattern language for describing how two entities are related over an anchor link.

Finally, text patterns, structure patterns, and link patterns have to be combined to be able to describe the relationship between two entities that are in two document fragments and, within their individual fragments, are embedded in some structure and, within that structure there is a text pattern that is required to define the relationship.

As the Web becomes increasingly XML-enabled these relationships would be over XLinks [16] and XPointers [18]. As we realize the vision of the semantic Web, metadata graphs would be used for mining.

7. Proposal for discovering new relations

So far in our discussion we have kept the notion of relation fixed in any instance of mining. For example, the relationship between acronyms and their expansions or between acronyms and theirs formation rules is fixed. These two relationships together define the overall relationship of the acronym problem that we are solving. We mine Web pages iteratively to find entities that substantiate that relationship. However, if we were to treat relations themselves as variables that can be mined and defined iteratively, we can come up with new relations.

Here we formalize this as follows. Let D be the

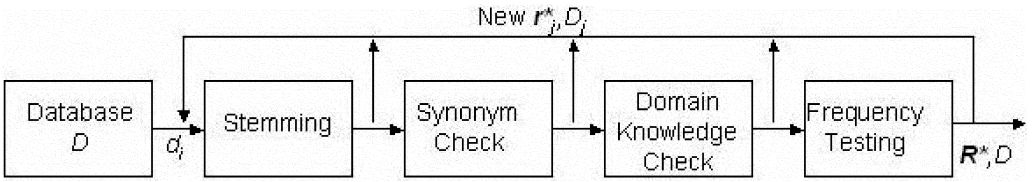


Fig. 4. Block diagram of the relationship identification process.

set of the pairs, (r, p), where relation r is a pair of entities, such as (person, company) or (book, author), and p is a phrase in which the relation is defined, such as 'John is an employee of IBM'. Let R be the set of relations and P be the set of phrases. Let $\boldsymbol{R}^*$ denote the set of relationships, r, in D. Formally, the terms are defined as follows:

(1) Database of Relations: $R = \{r \mid r = (e_1, e_2)$, where e_1 and e_2 are entities$\}$,
(2) Database of Phrases: $P = \{p\}$,
(3) Database of Relationships: $\boldsymbol{R}^* = \{\boldsymbol{r}^*\}$,
(4) Database of (relation, phrases): $D = \{d \mid d = (r, p), r$ in R, p in $P\}$

The problem is

(1) to identify $\boldsymbol{R}^* = \{\boldsymbol{r}_1^*, \boldsymbol{r}_2^*, \ldots, \boldsymbol{r}_n^*\}$, from D, and
(2) to partition D into classes $D_1, D_2, \ldots, D_n$, such that all pairs in D_i pertain to the same relationship and D is the union of all D_is.

Informally, the problem can be solved as follows. We start with D and an empty set R. For the first pair of D, d_1, we

(1) create a new relationship of d_1, and
(2) create D_1 and add d_1 to D_1. Suppose $d_1 =$ ((John, IBM), 'is employed by').

We create a new relationship, $\boldsymbol{r}_1^* =$ 'employment', and $D_1 = \{d_1\}$.

For a new d_i,

(1) if its relationship is already identified as $\boldsymbol{r}_j^*$, add it to the corresponding class D_j, or
(2) if it does not belong to any known relationship, create a new relationship, $\boldsymbol{r}_k^*$, and a new class D_k, and add d_i to D_k.

For example, suppose $d_i =$ ((Mary, IBM), 'works for'). Since 'works for' represents employment relationship, d_i is added to D_1. If $d_i =$ ((Smith, Smith Software), 'owns') and suppose an ownership relationship has not been identified, we create $\boldsymbol{r}_j^* =$ 'ownership', and place d_i into D_j.

The following paragraphs describe the techniques for identifying the relationship of d. The techniques are applied to each d_i. Fig. 4 illustrates the block diagram of the relationship identification process.

Stemming. This process strips content terms of common suffixes (such as -s or -ing or -ed) that indicate plurality, verb tense, conjugation, and so on, and leaves only the root of the term. The content terms after stemming is used to determine the relationship. For example, 'employee', 'employer', and 'employed' all become 'employ' after stemming and belong to 'employment' relationship.

Synonym check. This process identifies synonyms of a relationship word on the basis of the definition in a thesaurus, such as [12] and [15]. Phrases with synonyms are classified into the same relationship. For example, 'works for' is a synonym of 'employed by'. Both are classified in 'employment' relationship.

Domain knowledge checking. Some phrases convey multiple relationships. For example, a phrase 'by' can convey many relationships, such as '(a book) by (author)' for authorship, '(a house) by (the lake)' for locational proximity. For those phrases, we can apply available domain knowledge that specifies the qualification of entities. For example, the author entity in the authorship relationship has to be a person. Therefore, the (A,B) relation in the 'A by B' phrase cannot be an authorship relationship, unless B is a person.

Frequency testing. When a phrase implies multiple relationships, and is not resolved by the domain knowledge checking, we make a statistical judgment of the relationship on the basis of the frequency. That is, for a new instance of (relation, phrases) with the phrase with multiple relationships, such as 'by', with no other domain knowledge, we classify it as the most frequent relationship among those signified by the phrase. For example, suppose we have an (A,B) relation in an 'A by B' phrase, i.e., $d_i =$ ((A,B), 'by'), and we have no knowledge about A or B. Further we assume that there are two relationships

identified by the phrase 'by', $\boldsymbol{r}_j^*$ and $\boldsymbol{r}_k^*$ (such as *authorship* and *locational proximity*) where the number of relations of the form 'X by Y' that belongs to $\boldsymbol{r}_j^*$ is greater than that of $\boldsymbol{r}_k^*$. Then, d_i is classified as relationship $\boldsymbol{r}_j^*$.

7.1. The relationship mining algorithm

(1) Let R be a database of Relations, $R = \{r \mid r = (e_1, e_2)$, where e_1 and e_2 are entities$\}$.
Let $\boldsymbol{R}^*$ be a database of Relationships. Set $\boldsymbol{R}^*$ to empty set.
Let P be a database of Phrases, $P = \{p\}$.
Let D be a given database of (relation, phrases), $D = \{d \mid d = (r, p), r \in R, p \in P\}$
(2) Set $i = 1$.
(3) Stem the content word of phrase (p_i) of d_i, and determine the relationship of the stemmed term. If the relationship, $\boldsymbol{r}_j^*$ $(j \leq i)$, is already identified, add d_i to D_j and go to step (8).
(4) Look up the synonyms of terms in p_i, and determine the relationship, $\boldsymbol{r}_j^*$ $(j \leq i)$, of the synonyms. If the relationship is already identified, add d_i to D_j, and go to step (8).
(5) Look up domain knowledge of d_i, if available. If the domain knowledge uniquely determines the relationship, and the relationship, $\boldsymbol{r}_j^*$ $(j \leq i)$, is already identified, add d_i to D_j, and go to step (8).
(6) If the relationship of d_i is ambiguous, but identified by other pairs, choose a relationship $\boldsymbol{r}_j^*$ $(j \leq i)$ with highest frequency, and add d_i to D_j,
(7) Create a new relationship for d_i, called $\boldsymbol{r}_j^*$ $(j \leq i)$, and a new class D_j, and add d_i to D_j.
(8) $i = i + 1$.
(9) If $(i \leq |D|)$ go to (3); otherwise exit.

8. Related work

Bibliometrics [10] studies the world of authorships and citations through measurement. Bibliometric coupling measures similarity of two technical papers based upon their common citations. Co-citation strength is a measure of the number of times two papers are cited together. Statistical techniques are used to compute these and other related measures [11]. In typical bibliometric situations the citations and authorships are explicit and do not have to be learned or derived as in our system.

HITS (Hyperlink-Induced Topic Search) [6,8] is a system that identifies authoritative Web pages based upon the link structure of Web pages. It iteratively identifies hub pages (pages which point to authorities) and authority pages (pages which are pointed to by hub pages). The difference between HITS and our system is that in HITS the 'pattern' space is the links (anchor tags) in a Web page. Also, the 'hub' pages and the 'authority' pages are of the same kind — they are all Web pages. In our formulation of a general duality problem the related entities are not restricted to Web pages. For instance, neither acronyms nor their expansions stand for Web pages. They just occur close to each other in Web pages and form a tuple for our measure. In an interesting extension of our problem we could define the notion of 'goodness' as defined in HITS. For instance, we can formulate good acronyms and their expansions, good patterns of their occurrences, and good rules of acronym formation as follows.

Good (acronym, expansion) pairs are those that are identified by a large number of good patterns. Good patterns are those that identify large number of good (acronym, expansion) pairs. Similarly, good acronym formation rules are those that identify a large number of good (acronym, expansion) pairs; and good (acronym, expansion) pairs are those that are identified by a large number of good formations.

DIPRE (Dual Iterative Pattern Relation Expansion) [2] addresses the problem of extracting (author, book) relations. Brin [2] observed that given a set of pattern P with high coverage and low error rate, a very good approximation to R can be constructed simply by finding all matches to all the patterns. This system mines just the text in the pages to identify the relations. Unlike our system, this system does not involve strict formation rules and double duality which enhances the quality of our results.

Collins and Singer [4] use unlabeled examples on inducing lexicons or other knowledge sources from large corpora. The task is to learn a function which classifies an input string to one of the following categories: *Person*, *Organization*, or *Location*, with only small seed rules. They leverage natural redundancy in the unlabeled data either in *spelling* or *contextual*

rules. For example, in

..., says **Mr. Cooper**, a vice president of ...

both a spelling feature (that the string contains *Mr*.) and a contextual feature (that *president* modifies the string) are strong indications that **Mr. Cooper** is of the type *Person*.

9. Conclusions and future work

In this paper we studied the duality problem of how entities are related on the Web. Given that the Web is a great source of information where the information itself is buried under the visual markups, texts, and links of the Web pages, discovering relationships between entities is an interesting problem. The repeated occurrences of loosely defined structures and relationships help us define these entities with increased confidence. In this paper we formalized the iterative process of mining for patterns and relations over text, structures, and links. We defined and solved the (acronym, expansion) two-level duality problem. We also proposed ideas on mining for new relations. Currently we are working on generalizing our implementation to work for arbitrary user-defined relations and on improving our pattern language over structures and links. We anticipate the seamless adoption of XML in the future and are working on support for XML-style well-formed structures and links [16,18]. In the future, as the Web evolves from a structural Web to a semantic Web, we envision such duality mining over data and metadata to identify higher-level metadata as a key area of research.

References

[1] T. Berners-Lee, Weaving the Web, Harpers, San Francisco, CA, 1999.
[2] S. Brin, Extracting patterns and relations from the World Wide Web, in: Proc. WebDB '98, Valencia, 1998.
[3] S. Chakrabarti, M. van de Berg and B. Dom, Focused crawling: a new approach to topic-specific Web resource discovery, in: Proc. 8th World Wide Web Conference '99 (WWW8), Toronto, 1999.
[4] M. Collins and Y. Singer, Unsupervised models for named entity classification, in: EMNLP 99, 1999.
[5] Extensible Markup Language (XML) 1.0, W3C Recommendation, T. Bray, J. Paoli and C.M. Sperberg-McQueen (Eds.), Feb. 1998, available from http://www.w3.org/TR/1998/REC-xml-19980210.
[6] D. Gibson, J. Kleinberg and P. Raghavan, Inferring Web communities from link topology, in: HyperText '98, Pittsburgh, PA, 1998, pp. 225–234.
[7] T. Kistler and H. Marais, WebL: a programming language for the Web, in: Proc 7th World Wide Web Conference '98 (WWW7), Brisbane, 1998.
[8] J. Kleinberg, Authoritative sources in a hyperlinked environment, in: Proc. 9th ACM–SIAM Symposium on Discrete Algorithms, May 1997.
[9] R. Kumar, P. Raghavan, S. Rajagopalan and A. Tomkins, Trawling the Web for emerging cyber-communities, in: Proc. 8th World Wide Web Conference '99 (WWW8), Toronto, 1999.
[10] R. Larson, Bibliometrics of the World Wide Web: an exploratory analysis of the intellectual structure of cyberspace, Technical Report, School of Information Management and Systems, University of California, Berkeley, 1996, http://sherlock.sims.berkeley.edu/docs/asis96/asis96.html.
[11] K. McCain, Mapping authors in intellectual space: a technical overview. Journal of the American Society for Information Science 41 (1990) 433–443.
[12] Miller, Introduction to WordNet: an on-line lexical database, ftp://ftp.cogsci.princeton.edu/pub/wordnet/5papers.ps.
[13] QL'98 — The Query Languages Workshop, available from http://www.w3.org/TandS/QL/QL99.
[14] W3C Technical Reports and Publications, available from http://www.w3.org/TR/.
[15] Webster Online, available from http://work.ucsd.edu:5141/cgi-bin/http_webster.
[16] XML Link Requirements 1.0, W3C Working Draft, S. De Rose (Ed.), Feb. 1999, available from http://www.w3.org/TR/NOTE-xlink-req/.
[17] XML Path Language Version 1.0, W3C Recommendation, Nov. 1999, available from http://www.w3.org/TR/xpath.
[18] XML XPointer Requirements 1.0, W3C Working Draft, S. De Rose (Ed.), Dec. 1999, available from http://www.w3.org/TR/xptr.
[19] XSL Transformations Version 1.0., W3C Recommendation, Nov. 1999, available from http://www.w3.org/TR/xslt.
[20] J. Yi, N. Sundaresan and A. Huang, Using data mining techniques for building a topic-specific web search engine (submitted for publication).

Neel Sundaresan is a research manager of the eMerging Internet Technologies Department at the IBM Almaden Research Center. He has been with IBM since December 1995 and has pioneered several XML and internet related research projects. He was one of the chief architects of the Grand Central Station project at IBM Research for building XML-based search engines. He received his Ph.D. in computer science in 1995. He has done research and advanced technology work in the area of compilers and programming languages, parallel and distributed systems and algorithms, information theory, data mining and semi-structured data, speech synthesis, agent systems, and internet tools and technologies. He has over 30 research publications and has given several invited and refereed talks and tutorials at national and international conferences. He has been a member of the W3C standards effort.

Jeonghee Yi is a Ph.D. candidate in computer science at the University of California, Los Angeles. She is a researcher at IBM Almaden Research Center, San Jose, California since July 1998. Her current research interests include data mining, Web mining, internet technologies, semi-structured data, and database systems. She received a BS and a MS degrees in Computer Science from Ewha Woman's University, Korea, in 1986 and 1988, respectively, and a MS degree in computer science from the University of California, Los Angeles in 1994. The work described here was partially supported through an IBM Graduate Fellowship.