WISDOM: Web Intrapage Informative Structure Mining Based on Document Object Model

Hung-Yu Kao, Jan-Ming Ho, *Member*, *IEEE*, and Ming-Syan Chen, *Fellow*, *IEEE*

Abstract—To increase the commercial value and accessibility of pages, most content sites tend to publish their pages with intrasite redundant information, such as navigation panels, advertisements, and copyright announcements. Such redundant information increases the index size of general search engines and causes page topics to drift. In this paper, we study the problem of mining *intrapage informative structure* in news Web sites in order to find and eliminate redundant information. Note that intrapage informative structure is a subset of the original Web page and is composed of a set of fine-grained and informative blocks. The intrapage informative structures of pages in a news Web site contain only anchors linking to news pages or bodies of news articles. We propose an intrapage informative structure mining system called ***WISDOM*** (***W****eb* ***I****ntrapage* ***I****nformative* ***S****tructure Mining based on the* ***D****ocument* ***O****bject* ***M****odel*) which applies Information Theory to DOM tree knowledge in order to build the structure. WISDOM splits a DOM tree into many small subtrees and applies a top-down informative block searching algorithm to select a set of candidate informative blocks. The structure is built by expanding the set using proposed merging methods. Experiments on several real news Web sites show high precision and recall rates which validates WISDOM's practical applicability.

Index Terms—Intrapage informative structure, DOM, entropy, information extraction.

1 Introduction

Many Web pages are generated online for Web site maintenance, flexibility, and scalability purposes. They are usually generated by putting page content stored in back-end databases into predefined templates. The experimental results in [4] show that, on average, 43 percent of Web pages contain templates which indicates how pervasive template usage has become. Most commercial Web sites, such as search engines, portal sites, e-commerce stores, and news, apply a systematic technique to generate Web pages and to adapt various requests from numerous Web users. These sites are referred to as *systematic* Web sites [16]. The evolution of automatic Web page generation and the sharp increase of systematic Web sites have contributed to the explosive growth of Web page numbers. There exists much redundant and irrelevant information in these Web pages [1], [23], such as navigation panels, advertisements, catalogs of services, and announcements of copyright and privacy policies which are distributed over almost all pages of a systematic Web site. Such information is still crawled and indexed by search engines and information agents, thus significantly increasing corresponding storage and computing overhead.

We define specific regions of a page that users are interested in as *informative blocks* (or referred to as *IB*). Information within *IBs* manifests the main topic of the page and indicates related information. The set of these blocks and corresponding connecting structures form the *informative structure* (or referred to as *IS*) of the page. Fig. 1 shows the *IS* of an example news page and its corresponding parts of content. A Web page can be represented by a tree structure, i.e., **D**ocument **O**bject **M**odel (**DOM**) [27] and each content block in a page is a subtree of the original DOM tree. The *IS* can be defined as a reduced tree united by subtrees of *IB*s. The tree relation of united subtrees is also kept in *IS*. The *IS*, for example, in Fig. 1 is built by uniting subtrees of two news table of content blocks and is a tree reduced from the original DOM tree. As proposed in [16], which deals with the *IS* in a *site*, called *interpage informative structure*. The structure is composed of informative pages within a Web site and interconnecting links. In this paper, we work with *ISs* of *individual pages*, called *intrapage informative structure* (for simplicity, we use the same denotation *IS* in the last parts of the paper). Each page has its own *IS* and the structure is composed of *IB*s within the page.

Web informative content mining is an important task for search engines and Web agents [11]. Internet crawlers can use the *IS* to focus on crawling informative paths. Search engines can reduce the size of indices and make them more precise by removing the redundant and irrelevant page blocks. Intermedia information agents that search for specific information among Web sites with different presentation styles, page layouts, and site mapping can also benefit from the information preprocessed by the structure.

News search engines like Google News, Altavista News, and NSE[1] are typical examples of intermedia information agents. They crawl diverse news articles from thousands of news Web sites and extract and index article blocks. The *IS* of a page consists of sets of table of contents (abbreviated as TOC) blocks and news article blocks. The structure helps

1. Google News: http://news.google.com, Altavista News: http://www.altavista.com/news, and NSE: http://nse.iis.sinica.edu.tw.

- H.-Y. Kao is with the Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan, ROC. E-mail: hykao@mail.ncku.edu.tw.
- J.-M. Ho is with the Institute of Information Science, Academia Sinica, Taipei, Taiwan, ROC. E-mail: hoho@iis.sinica.edu.tw.
- M.-S. Chen is with the Graduate Institute of Communcation Engineering and the Electrical Engineering Department, National Taiwan University, Taipei, Taiwan, ROC. E-mail: mschen@cc.ee.ntu.edu.tw.

Manuscript received 8 Sept. 2003; revised 3 June 2004; accepted 22 Sept. 2004; published online 17 Mar. 2005.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0180-0903.

1041-4347/05/$20.00 © 2005 IEEE Published by the IEEE Computer Society

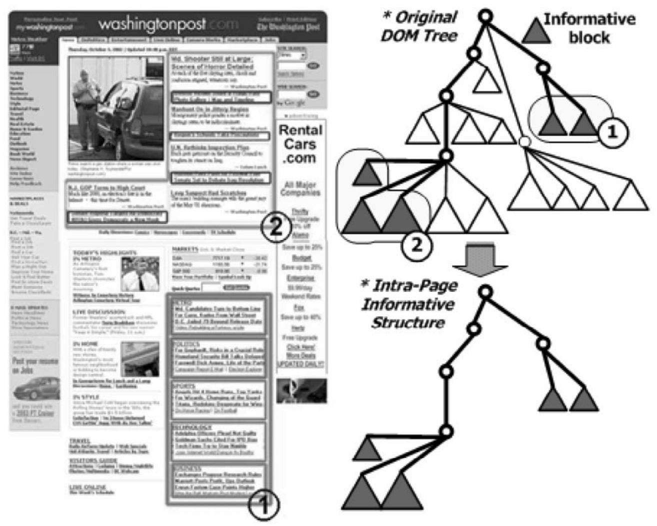


Fig. 1. The original document structure and its *IS* of a news page from WashingtonPost.

agents to automatize crawling and indexing. The *IS* of article pages usually consists of fine-grained and joined *IBs* and most of them contain only one HTML tag in its subtree, i.e., leaf nodes in the DOM tree. These blocks contain news article metadata, such as title, date, reporter, and place, which are very useful in categorizing pages for news information agents and metadata extraction. Agents can automatically extract article metadata by using this structure.

According to the definitions of hubs and authorities in [18], a good hub is a page linking to a good authority page that is relevant to some specific query. Analogously, we define a good *information hub* as a block linking to a good *information authority* block which will provide useful information. The *IS* of a page can then be considered as the set of blocks of good information hubs and good information authorities within that page. Note that Web pages in news Web sites usually contain the obvious and clear *ISs*, i.e., TOC and article blocks, in our observation. Fig. 2 shows the root page of the news Web site WashingtonPost (http://www.washingtonpost.com) and blocks 1 and 2 which provide anchors linking to hot news and selected news are the information hubs. We consider these two blocks as *IBs* as they are the crawling points for news information agents to collect daily news. Block 3 is merely a menu block which is appended ubiquitously to most pages in the WashingtonPost Web site, and is thus considered as redundant.

In an HTML document, tags are inserted for purposes of the page layout, content presentation, and for providing interactive functions, e.g., form filling and document linking. After being rendered by the browser, tags are invisible to users and are represented by means of visual appearances and functions. The layout and style of presentations provide hints to users for accessing and understanding information easily. The corresponding tagging structure therefore contains information about representation and semantics of Web pages. For example, a group of tight sibling anchor nodes with the short anchor-text, e.g., the tagging tree of block 3 in Fig. 2, is different from a group of sibling nodes with the long anchor-text interleaved with context nodes, e.g., the tagging tree of block 2 in Fig. 2. In Fig. 2, block 1 containing several tightly coupled anchor groups also provides different functionality and representation from block 2 and block 3. In news Web sites, a TOC block containing categorized news is usually similar in structure to that of block 1. Block 2 is also an informative TOC containing the abstracts of news and anchors linking to the news articles. Such useful evidences are more prominent in pages of the *systematic* Web sites in which *ISs* are usually generated automatically and dynamicallyby an iterative program from predefined templates.

In this paper, we extract and use knowledge from the tagging tree structure of a Web page and apply the Information Theory to mine the *IS*. Considering the structure information and context in these nodes together, we are able to understand and extract the meaning of information contained in Web pages more clearly and precisely. Specifically, we propose in the paper an *IS* mining

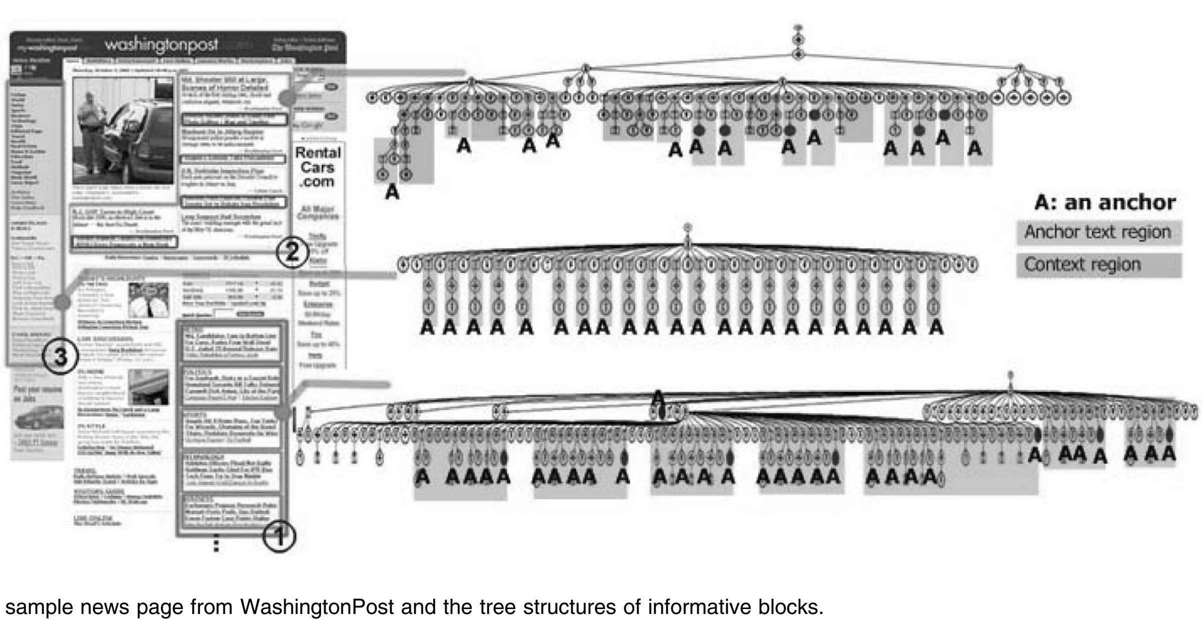


Fig. 2. A sample news page from WashingtonPost and the tree structures of informative blocks.

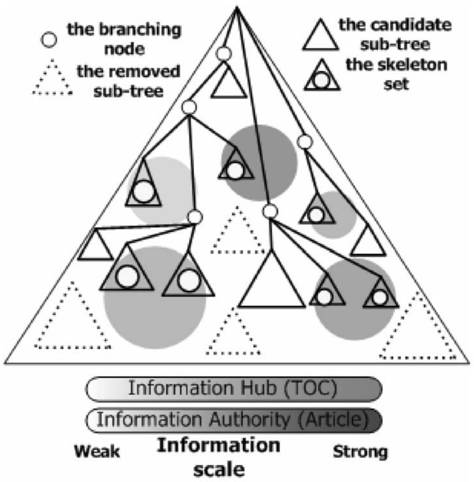


Fig. 3. The tree splitting and selection in WISDOM.

system called ***WISDOM***, standing for ***W****eb* ***I****ntrapage* ***I****nformative* ***S****tructure Mining based on the* ***D****ocument* ***O****bject* ***M****odel* to automatically extract and recognize the *IS* of each page in a Web site.

The main mining flow in WISDOM first uses the Information Theory to evaluate the information amount contained in each DOM tree node and then constructs the *IS* by applying the specific searching, filtering, and merging methods. The searching step finds *IB* candidates and its principle is based on the observation that the root node of an *IB uniformly* spreads its information around its children nodes in most cases. In view of this, WISDOM first splits the original DOM tree into several small and nonoverlapped subtrees as shown in Fig. 3 and selects some of them as the candidate subtrees, in accordance with the assigned threshold of the structure information. The threshold is applied for the judgment on the uniformity of information distribution. Some uninformative subtrees are removed in the step. Our system then applies a top-down *IB* searching algorithm to find the top-k most informative blocks and the corresponding filtering criteria to select a set of candidate *ISs* called **the skeleton set**. The skeleton set can be considered the core subtrees of the *IS* shown in the color shaded regions in Fig. 3. The *IS* is built by expanding the skeleton set using the proposed merging methods. The merging method works in a bottom-up manner to link the qualified sibling nodes in the skeleton set and other informative nodes.

The remainder of this paper is organized as follows: In Section 2, we describe related work. WISDOM is described in Section 3. In Section 4, we evaluate the performance of WISDOM by testing it on several real news Web sites, university, and commercial Web sites. The Section 5 gives our conclusion.

2 Related Work

Many works have been proposed that aim to extract the information of a page. Works on *wrappers* [9], [19], [22] provide learning mechanisms to mine the extraction rules of documents. The WebKB project in [6] automatically creates a computer understandable knowledge base from the textual content and hyperlink connectivity of Web pages. The work describes an inductive logic programming algorithm for learning wrappers and develops a trainable information extraction system. Works in [1], [15], [20] provide auxiliary systems to aid in the information extraction from semistructured documents. The clipping method proposed in [14] is based on a supervised learning to provide a practical tool to cut the news articles. However, they need either a premarked training set or a considerable amount of human involvement to perform information extraction. When we consider the whole World Wide Web as our problem domain, building a useful training set to represent the diversity of Web content and structure is very hard.

In a systematic Web site, *IBs* are usually generated by a loop program; the entities in blocks are therefore similar to one another in view of their tag patterns and information they carry. In Fig. 2, it can be seen that the tag patterns of micro blocks (the shaded regions) in *IBs* 1 and 2 look very similar to one another. Therefore, frequent substructure mining is a candidate solution for automatic extraction of *IB*s. The topic of mining frequent substructure on the DOM trees of semistructure pages has recently been studied in [2], [10], [24] where the frequent subtree was extracted by respective pattern mining and noise node concealment methods, such as the wildcard mechanism in [10] and node-skip and edge-skip pruning in [2]. Works also use the tree pattern mining to extract metadata information in Web pages [13], [28]. However, semantic information in mined blocks with the same tree structure may be different from one to another. We need other information measurement methods to filter out redundant information blocks from those blocks with a similar tree structure. Moreover, some *IB*s like article blocks are laid out with the unique structures and are indeed difficult to extract by the frequent structure mining.

Some techniques proposed in [12], [29] use the semantics and relationships of tags to extract the record boundaries of Web pages. Several heuristic rules of tag characteristics, such as the highest count-tags (HT), identifiable "separator" tag (IT) and repeating tag pattern (RP), are proposed in [12] and are applied to extract record boundaries on several ".com" Web sites. Research in [29] also categorized tags into several groups according to their tagging functionalities and discovered the major schemas between them to translate HTML documents to XML documents in a semantic view.

Research in [7] extends the definition of a hub by dividing a hub page into several fine-grained hub blocks with different hub values. This is accomplished by calculating and integrating the hub values of each anchor node in the DOM tree of a page. Entropy analysis proposed in [21] discriminates the informative authorities of pages by dividing a page into several authority blocks with different authority values weighted by the information of each block.

There are also works on mining informative structure [16], [21], which are different from our work in that they mainly deal with mining blocks delimited by <TABLE> tags. In contrast, we mine fine-grained blocks using the DOM tree. It is worth mentioning that in the problem of mining the fine-grained *IB*s in a page, a straightforward approach would be to divide the page into several unit blocks that contain only one tag and then to merge neighboring blocks that contain information together. This naive method, however, does not work well for real-world Web pages in our opinion because: 1) When an *IB* is divided into small blocks with the one-tag granularity,

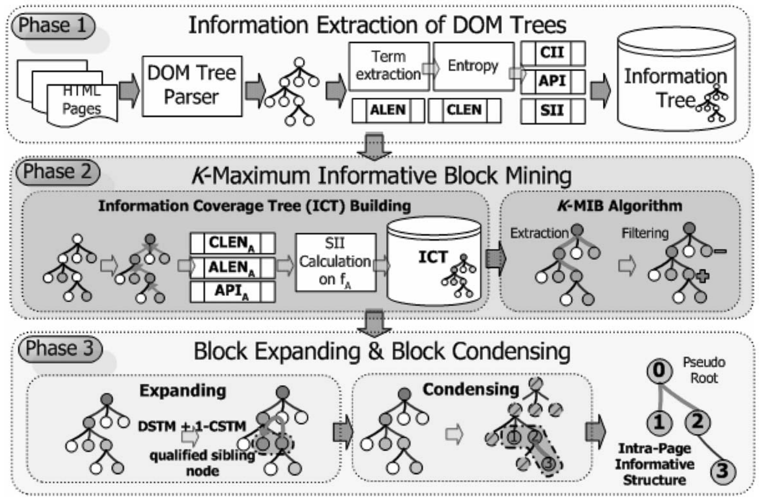


Fig. 4. WISDOM system flow.

the information contained is also divided into many small pieces which are difficult to discriminate from noises and redundant information and 2) there is no obvious method to merge such small blocks to form meaningful and integrated *IB*s. We therefore propose a top-down mining instead of bottom-up algorithm to extract fine-grained *IB*s.

3 WISDOM: A DOM-Based Mining System

WISDOM automatically extracts and recognizes *ISs* of each page in a Web site according to the knowledge in the tree structures of pages. As shown in Fig. 4, WISDOM consists of three phases: 1) information extraction from DOM trees, 2) *k*-maximum informative block mining, and 3) block expansion and condensation. In the first phase, we extract useful features from the information of the original DOM tree. These features can be classified into two types of information: node information and structure information. In the second phase, we aggregate the node information to build the ***I***nformation ***C***overage ***T***ree (***ICT***). According to the *ICT*, we devise a greedy algorithm, i.e., *k*-maximum informative block mining algorithm (*k*-MIB), to extract subtrees that contain richer information. The extracted subtrees are either better information hubs or better information authorities, depending on the criteria employed in the greedy algorithm. They form the skeleton set of the *IS* of a page. We then expand the skeleton set by assembling neighboring subtrees that contain similar features corresponding to the original skeleton subtrees. After condensing the expanded set by removing dummy nodes, the assembled forest (or tree), in essence the *IS* of a page, is constructed.

3.1 Phase 1: Information Extraction from DOM Trees

In the beginning, we crawl pages of a Web site in a specific crawling depth. When a page is crawled, we first extract the tree structure of a page based on DOM. Note that some HTML pages are not well-conformed, e.g., missing the ending tag for the <a> tag. We use HTMLTidy[2] to fix syntax mistakes in source documents. In tree *T*, each node represents a tag in the page and contains the tag name information, attributes in the tag statement, and its *innerText*, i.e., the context delimited by the tag. From the definition of DOM, the context of the *innerText* of node *N* includes all contexts of nodes in the subtree rooted by node *N*. We use *T(N)* to denote the subtree rooted by node *N*. The *innerText* of the root node in each page is the context of a page when all tags are removed. The text of a Web page can be classified into two types: 1) anchor texts and 2) contexts which are texts delimited by all other tags except <A> tags. We use *ALEN* to represent the length of the anchor text of a node and *CLEN* to represent the length of the contexts. A list of symbols used in this paper is given in Table 1.

We then parse the *innerText* of the root node to extract meaningful terms. A term corresponds to a meaningful keyword or phrase. Applying stemming algorithms and removing stop words based on a stop-list, English keywords (terms) can be extracted in a systematic manner [26]. Extracting terms in oriental languages is more difficult because of the lack of separators in these languages. In our system, we use an algorithm to extract keywords from Chinese sentences based on a Chinese term base. This base was generated by our search engine[3] by collecting hot queries and excluding stop words. After extracting terms in all crawled pages, we calculate the entropy value of each term according to its term frequency. From Shannon's information entropy [25], the entropy of term $term_i$ can be formulated as:

$$EN(term_i) = -\sum_{j=1}^{n} w_{ij} \log_n w_{ij},\ where\ w_{ij} > 0$$
$$and\ n = |D|,\ D\ is\ the\ set\ of\ pages,$$

where w_{ij} is the value of normalized term frequency in the page set. In the experiments on real Web sites containing a huge amount of pages, it is not practical to recalculate entropy values directly when a new page is crawled. In WISDOM, we use an incremental entropy calculation

2. HTMLTidy is an HTML fixing tool developed by Dave Raggett from the W3C team, http://www.w3.org/People/Raggett/tidy/.

3. The searching service is a project sponsored by Yam, a commercial search engine in Taiwan (http://www.yam.com/).

TABLE 1
The List of Symbols Used

Abbr.	Description	Abbr.	Description
ALEN	length of anchor text	**$\mathbf{ALEN_A}$**	aggregated ALEN
CLEN	length of contexts	**$\mathbf{CLEN_A}$**	aggregated CLEN
API	anchor precision index	**$\mathbf{API_A}$**	aggregated API
F	the set of tuple values, (ALEN, CLEN, API)	**$\mathbf{F_A}$**	the set of aggregated tuple values, ($ALEN_A$, $CLEN_A$, API_A)
T	a DOM tree	**ICT**	tree T with the aggregated set F_A
N	a node in the tree	**T(N)**	sub-tree rooted by N
innerText	contexts contained in T(N)	**$\mathbf{TLEN_A}$**	InnerText length
CII	content information index	**SII**	structure information index
ST	SII threshold	**TC**	type constraint
DSTM	direct sibling tree merging	**k-CSTM**	the k-th collateral sibling tree merging

process in the real Web site analysis. In the incremental calculation process, the new entropy value $E_{k+1}(f_j)$ is calculated only by the previous entropy value $E_k(f_j)$, total term frequency $TF_{j,k}$, and the new term frequency $tf_{(k+1)j}$ of the new included page for term f_j. The incremental calculation can be described as

$$E_{k+1}(f_j) = \begin{cases} \frac{E_k(f_j)}{\log_k(k+1)}, & when\ tf_{(k+1)j} = 0 \\ \Phi(E_k(f_j), TF_{j,k}, tf_{(k+1)j}), & otherwise. \end{cases}$$

The proof of the correctness of the process is given in Appendix A (which can be found on the Computer Society Digital Library at http://computer.org/tkde/archives.htm).

We define the weight of a term T_j as $W(T_j) = 1 - EN(T_j)$ to represent the importance of the term. The reason behind applying entropy calculation is that terms distributed in more pages in a Web site usually carry less information to users. In contrast, those appearing in fewer pages carry more information of interest. The weight of a term is similar to its inverse document frequency, IDF [3], which is defined as $\log_n \frac{n}{df_j}$, where df_j is the document frequency of T_j. IDF is usually applied to represent the discriminability of a term in a set of documents. According to the definition, we can conclude following relationships between $W(T_j)$ and IDF_j: 1) If T_j is uniformly distributed among some pages, $W(T_j) = IDF_j$. 2) If T_j is not uniformly distributed among the same pages in Item 1, then $W(T_j) > IDF_j$ and the more skewed the distribution of T_j is, the larger $W(T_j)$ is. The two relationships are proven in [17] and we include the detail of proofs in Appendix A which can be found on the Computer Society Digital Library at http://computer. org/tkde/archives.htm. Benefiting from these two relationships, the weight of a term attained from the entropy value is more representative for the importance of a term than from IDF. We use the example illustrated in Fig. 5 to explain these relationships. In this figure, $\mathrm{Term_A}$ is uniformly distributed among Page 1 to Page 3 and $\mathrm{Term_B}$ has the same term frequency and the document count with $\mathrm{Term_A}$, but most $\mathrm{Term_B}$s are located at Page 3. These two relationships are conformed by the following calculations:

$$\begin{cases} W(Term_A)=1-EN(Term_A)=1+3*\frac{2}{6}\log_4\frac{2}{6}=0.207519 \\ \quad =IDF(Term_A)=\log_4\frac{4}{3}=0.207519 \quad (1) \\ W(Term_B)=1-EN(Term_B)=1+2*\frac{1}{6}\log_4\frac{1}{6}+\frac{4}{6}\log_4\frac{4}{6} \\ \quad =0.374185 > IDF(Term_B)=\log_4\frac{4}{3}=0.207519 \quad (2). \end{cases}$$

According to the extracted information, we calculate three extended features to gain more implicit information from the tree, namely, 1) the content information index (***CII***) which indicates the amount of information contained in the block, 2) the anchor precision index (***API***) which represents the similarity between the anchor-text and the linked document, and 3) the structure information index (***SII***) which indicates the distribution of children's feature values of one node in the DOM tree. Each node in DOM tree T contains the tuple values of the feature set $F = \{ALEN, CLEN, API\}$. In the following sections, we will describe their respective calculations.

3.1.1 Content Information Indices (CII)

When entropy values of terms are calculated, we average the weight values of terms in an *innerText* of node N to get the content information index of N, i.e.,

$$CII(N) = \frac{\sum_{j=1}^{k} W(term_j)}{k},$$
$$where\ \forall_{j=1\sim k}\ term_j\ in\ innerText\ of\ N.$$

The *CII* value of node N represents the amount of information carried in a subtree rooted by N. The works in [16], [21] have shown that the entropy value corresponds to the recognition of the context parts of article pages. Consider the *CII* distribution of an article page in Fig. 6a. Note that the DOM tree is built by a depth-first traversal. The node ID of each node in the tree is generated according to the traversal order. Nodes with close node IDs are

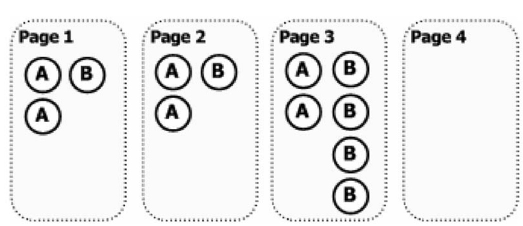
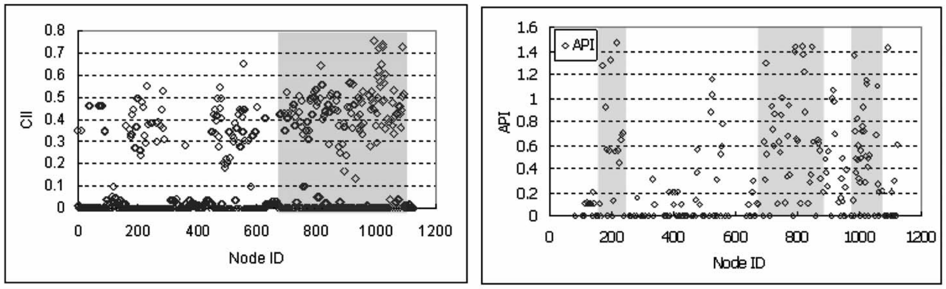


Fig. 5. An example of different term distributions.



(a) (b)

Fig. 6. The distributions of CII and API for two sample Web pages (shaded regions are IBs identified from the answer sets).

adjacent to each other in the physical layout of the page. The shaded region is an *IB* of the page, which is identified manually. Observe that 1) nodes with higher *CII* values in an *IB* are more than others and 2) nodes in an *IB* form a clear cluster in the *CII* distribution graph.

3.1.2 Anchor Precision Indices (API)

When browsing the Web, people use anchors to get information they want according to the semantics of anchors. The semantics of an anchor can be represented by the anchor text, text surrounding the anchor, the image, or other dynamic representations generated by scripts. The semantics of an anchor is expected to be relative to the page it links. Such relevance is, however, weak in some cases. We therefore define the value of the anchor precision index to indicate the correlation of the anchor and its linking page. We use the anchor text and the bounded text surrounding the anchor to evaluate the value of *API*. The correlation index *API* is defined as:

$$API(N) = \sum_{j=1}^{m} \frac{1}{EN(term_j)},$$

where $term_j$ is the term concurrently appearing in both the anchor text of N and the linked page and m is the number of matched terms.

The calculation of *API* stems from the similarity analysis between documents using the vector space model. We extend the model by using the inverse values of entropy to set the weights of terms. If the information amount in those matched terms is larger, we get a larger *API* value that indicates that the anchor carries more precise information. The usage of the inverse of entropy values in the API formulation is to emphasize and amplify the effect of matched terms. Moreover, the value of *API* is not normalized by the matched count because we want to show that the longer informative anchor text leads to more information. Note that $EN(term_i)$ is always larger than 0 because $term_i$ appears in at least two documents.

Consider the *API* distribution of a TOC page in Fig. 6b. It shows that the number of nodes with larger API values in the shaded region, i.e., regions of marked TOC blocks, are more than others on average. Anchors in the menu block have small *API* values because the anchor texts of these anchors are short and the entropies of terms they contain are almost one.

3.1.3 Structure Information Indices (SII)

The index *SII* of a node is calculated according to the distribution of the feature values of the node's children. However, some HTML tags either correspond to information that is not extractable or provide no useful information. Such tags, such as the comment tag <!>, the new line tag
, and the script program tag <script>, are called dummy tags and are removed from the following calculation of SII. We define the notion $f_i(N)$ as the value of feature f_i of node N, and $children(N)$ as the set of all nondummy children of the node N. For a simple tree structure of node N with children $n_0, n_1, \ldots, n_{m-1}$, we define the SII value of node N for feature f_i as:

$$SII(N,\ f_i) = -\sum_{j=0}^{m-1} w_{ij} \log_m w_{ij},$$

$$where\ w_{ij} = \frac{f_i(n_j)}{\sum_{k=0}^{m-1} f_i(n_k), \forall n_k \in children(N)}.$$

Note that $f_i(N)$ is larger or equal to the sum of $f_i(n_0), f_i(n_1), \ldots, f_i(n_{m-1})$. We apply entropy calculation here to represent the distribution of children's feature values of any node with more than one child. The value of *SII* indicates the degree that the feature values of the node are dispersed among its children. When the value of $SII(N, f_i)$ is higher, the values of all children's f_i tend to be equal.

In a systematic Web site, most context and anchors of TOC blocks are generated automatically. The styles, appearances, and information carried of entities in such a block are always similar from one to another. This phenomenon makes the *SII* values of these features become larger ones for the root nodes of such blocks.

3.2 Phase 2: The *k*-Maximum Informative Block Mining

In this phase, we first build the information coverage tree for features extracted during the phase one to obtain corresponding aggregated feature values. The proposed *k*-MIB algorithm is then applied to extract and filter out the candidate *IB*s. In Section 3.2.1, we describe the construction of *ICT* and the aggregated features. Extracting and filtering processes of the proposed algorithm are described in Section 3.2.2.

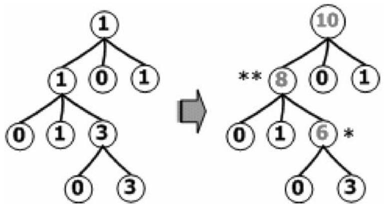


Fig. 7. An example of feature aggregation.

3.2.1 Information Coverage Tree Building

We define a tree with bottom-up aggregated features as an information coverage tree (abbreviated as *ICT*). In an *ICT*, any feature in the aggregated feature set F_A is obtained from the corresponding features in set F. Each node in an *ICT* contains all feature information of nodes in the subtree rooted by this node. The feature aggregation is a bottom-up process from the leaf nodes to the root node. The process of level k of the tree is shown below:

$$f_{Ai}(N) = f_i(N) + \sum f_{Ai}(n_j),$$
$$\forall n_j \in children(N) \text{ } and \text{ } level(N) = k.$$

We aggregate features from the lowest level of the tree to the level one. The complexity of the process is O(|N|). Fig. 7 shows an example aggregation process where the node marked by $*$ is labeled with $6 = 3 + (3 + 0)$ and the one marked by $**$ is labeled with $8 = 1 + (1 + 6)$.

The aggregated features in *ICT* for each node N are subject to the constraint where $f_{Ai}(n_j)$ is the aggregated value of feature f_i of node n_j:

$$f_{Ai}(N) \geq \sum_{j=0}^{m-1} f_{Ai}(n_j), \ \forall n_j \in children(N).$$

The length of *innerText* of each node is a typical aggregated feature because the *innerText* of a parent node contains all the *innerText* of its child nodes. We use $TLEN_A$ to represent the length of *innerText*. In WISDOM, we also aggregate node information $ALEN$ and API to get the corresponding aggregated features, denoted by $ALEN_A$ and API_A. Note that $TLEN_A(N)$ is composed of the length of contexts in $T(N)$, i.e., $CLEN_A(N)$, and the length of anchor texts in $T(N)$, i.e., $ALEN_A(N)$. The value of $TLEN_A$ is thus equal to $CLEN_A + ALEN_A$. We then apply the SII calculation on these three aggregated features to get corresponding structure information of aggregated features for each node.

3.2.2 Block Extracting and Block Filtering

The proposed maximum informative block mining algorithm $MIB(\boldsymbol{k}, \boldsymbol{f_A}, \boldsymbol{ST})$ is a greedy and top-down tree traversal process. For input value *k*, the algorithm outputs at most k *IB*s, i.e., TOC blocks or article blocks. The aim of the algorithm is to find the top-*k* nodes with maximal aggregated feature f_A values under the given *SII* constraint, i.e., ***S****II* ***T****hreshold* (***ST***). When the value of *ST* is larger, the structure constraint is tighter and the children of each extracted node in the resulting candidate set will have more similar values of aggregated features in accordance with the

```
Algorithm MIB (k, f_A, ST) begin
/* Cheap is a sorted stack */
InfoBlock = 0
Push root node into CHeap(f_A)
While (InfoBlock < k and CHeap is not empty) begin
        Pop Node N with max(f_A) from CHeap(f_A)
        If (SII(N, f_A) > ST or N is a leaf) then
                find = true
                If (N matches the type constrain) then
                        insert N into CandidateSet
                        InfoBlock = InfoBlock + 1
                end if
        else
                push children(N) into CHeap(f_A)
        end if
end
End
```

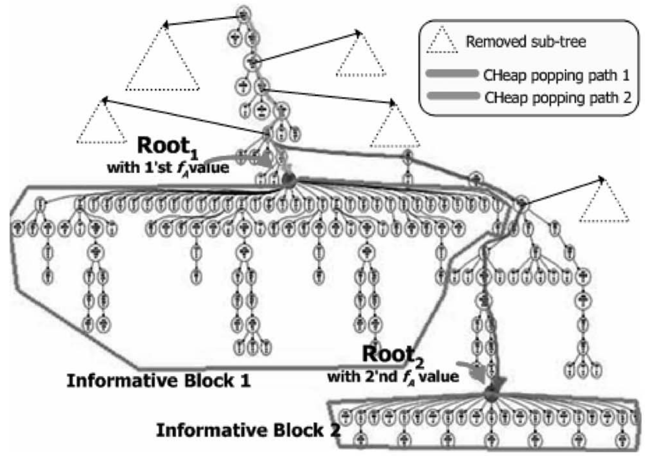


Fig. 8. An example of k-maximum informative block mining on the tree of a TOC page, $k = 2$.

definition of *SII*. The searching path of the algorithm is shown, for example, in Fig. 8. The original tree is extracted from a real TOC page by eliminating those subtrees removed by *MIB*.

When extracting the top-*k* candidate nodes, we apply type constraints to eliminate pseudoinformative nodes. Type constraints (*TC*) are dependent on the type of blocks described as:

$$\begin{cases} if \ type = \ "Article," \ CII(N) \geq 1 - TC_{article} \\ if \ type = \ "TOC," \ \frac{API_A(N)}{\#anchors \ in \ T(N)} \geq TC_{TOC}, \end{cases}$$
$$\forall N \in CandidateSet.$$

Type constraints are motivated from heuristic observations that 1) article blocks contain informative context and, hence, their entropy values must be bounded and 2) TOC blocks contain highly semantic relevant anchors linking to information authorities and the average *API* value should be more than others in blocks with redundant and irrelevant anchors. These heuristic constraints are useful in removing pseudoinformative blocks.

Due to the tree traversal characteristic of the MIB algorithm, each node in the filtered candidate set is not an ancestor of any other nodes. The subtrees rooted by these nodes are therefore isolated and nonoverlapped. The set of these selected subtrees is called the ***skeleton*** of the *IS* of a page and the root nodes of these subtrees ***skeleton nodes***.

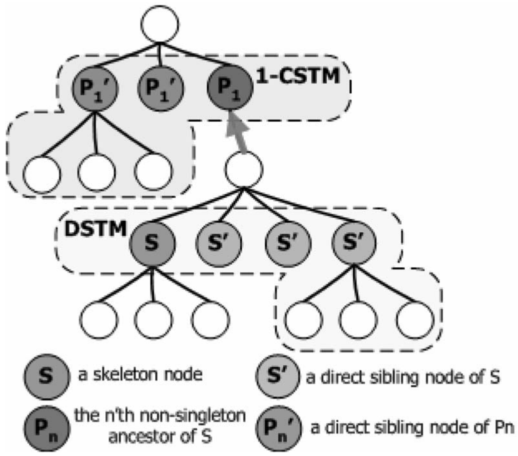


Fig. 9. Two sibling tree merging methods.

3.3 Phase 3: Block Expanding and Condensing

When investigating on the skeleton set, we find the selected skeleton nodes are often the subtrees of the *IB*s. The observation is more obvious when the structure threshold (ST) is larger and tighter. This is because the selected subtree is smaller when ST becomes larger in the k-MIB process. According to the skeleton structure, we therefore apply two sibling tree merging methods, i.e., direct sibling tree merging (*DSTM*) and collateral sibling tree merging (*CSTM*), to expand the skeleton set. The *DSTM* method merges the subtree rooted by qualified sibling nodes of each skeleton node S as shown in Fig. 9. Note that node S' may be one of skeleton nodes. Any qualified sibling node S' needs to match the type constraints and $f_A(S')$ must also be smaller than $f_A(S)$. We do not need to merge sibling nodes with the larger f_A values because they are checked in the previous searching paths of the *k-MIB* algorithm and have been either selected into the skeleton set or removed from the *IS*. After *DSTM*, we then select the nonsingleton ancestors, i.e., $P_1, P_2, \ldots, P_n$, of S for the process of *CSTM*. The *i*th nonsingleton ancestor P_i is the *i*th ancestor of S which has more than one nondummy sibling node. The dummy node is defined as a node whose value of f_A is zero, e.g., the node with $CLEN_A = 0$ for the article block and the node with $ALEN_A = 0$ for the TOC block. The method of *k-CSTM* is equal to applying *DSTM* on P_k. In WISDOM, we apply *DSTM* and *1-CSTM* to proceed the default block expanding. For example, in Fig. 9, we merge subtrees rooted by three S' into the skeleton set in the *DSTM* process. In the *1-CSTM* process, we first traverse the tree from the node S up to the root node to find the first nonsingleton ancestor P_1 and we then apply *DSTM* on P_1 to merge its qualified sibling nodes, i.e., two P_1' nodes.

The intention of *DSTM* is to merge small *IB*s surrounding the skeleton blocks together. The nonuniform distribution between f_A values of the skeleton node and corresponding sibling nodes leads to the node separation in the *k-MIB* phase. In our experiments, DSTM can merge the metadata blocks, i.e., the article title, date, reporter, etc., into the main body of the article news. They are all *IB*s, but the distribution of their context length is skewed.

The block condensing process removes the subtrees rooted by nodes that cannot match the type constraints from the expanding trees as these dummy subtrees are mainly tags for the page layout. This process is used to remove the uninformative subblocks from the merged trees obtained from the previous processes.

4 Experiments and Results

In this section, we describe several experiments conducted on some real news Web sites in order to evaluate the performance of WISDOM. Data sets used and employed evaluation criteria are described in Section 4.1. We evaluate the performance of selection and filtering in the *k*-MIB algorithm in Section 4.2. The performance of block expanding and condensing is assessed in Section 4.3. Finally, Section 4.4 provides the overall performance evaluation of WISDOM.

4.1 Data Sets

We conduct our experiments on the data sets[4] used in [16]. In addition to these news Web sites, for evaluating WISDOM on other domains, a good example is the data set used in the WebKB project [8] and the data set used in the page segmentation research [9]. These data sets contain several university sites and commercial Web sites as described in Table 2. To assess WISDOM, we add two new answer sets, i.e., TOC blocks and article blocks. These blocks are extracted manually by news domain experts according to their experience in issuing real-world newspapers. We select most TOC pages and some candidate pages among all article pages with different tagging structures to mark. Unmarked TOC pages are pages which cannot be correctly parsed, or those containing many outsite anchors linking to uncrawled pages, e.g., TOC pages in CNET and TTV. The latter case will cause the accuracy of *API* calculation to decrease suddenly and blur the evaluation results.

As shown in Table 2, the percentages of information coverage of the *IS* over the original page vary among data sets. Values are dependent on the styles and page layouts of news sites. The more redundant information added, the less information the *IS* carries.

To attain a quantitative evaluation, we employ two different evaluating methods to measure the values of precision and recall of article and TOC blocks. The TOC evaluation method is called significant node coverage (SNC). In SNC, we count the matched anchor nodes in subtrees rooted by nodes in the answer set and our output. For evaluating article blocks, we calculate the ratio of the matched context contained in each subtree by length to indicate the performance. The method is called information coverage (IC). The selection is made because only the context and anchors in the *IS* need to be indexed and extracted for crawling. In our experiments, we use the rates of precision (P) and recall (R) to indicate the similarity of these two sets. We also use F-measure [3] which combines recall and precision in a single efficiency measure. The value is the harmonic mean of precision and recall, and is formulated as $\frac{2^*(R^*P)}{R+P}$. With the example in Fig. 10, we show the evaluation results of four methods in Table 3. The answer sets are two sets of root nodes, i.e., the TOC answer set $\mathrm{A}_T = \{a_{T1}, a_{T2}, \ldots, a_{Tn}\}$ and the article answer set

4. Pages of Web sites in data sets were crawled on 2001/12/27, 2002/4/11, and 2004/3/29. The data sets can be retrieved at our research site: http://kp06.iis.sinica.edu.tw/isd/index.html.

TABLE 2
Data Sets and Their Informative Structure Distributions

Site Abbr.	URL	Total pages	TOC pages	Marked TOC pages	Marked article pages	Marked Toc blocks	Marked article blocks	Answer Coverage	
								TOC (S-NC)	Article (IC)
CDN	www.cdn.com.tw	261	25	22*	60#	38	63	46.30%	98.40%
CTIMES	news.chinatimes.com	3747	79	69	66	313	68	32.10%	82.50%
CNA	www.cna.com.tw	1400	33	29	50	106	50	21.90%	80.10%
CNET	taiwan.cnet.com	4331	78	38	37	84	86	17.50%	63.60%
CTS	www.cts.com.tw	1316	31	19	53	21	80	54.80%	52.10%
TVBS	www.tvbs.com.tw	740	13	12	50	25	50	73.70%	56.90%
TTV	www.ttv.com.tw	861	22	18	42	20	75	20.10%	54.50%
UDN	udnnews.com	4676	252	243	52	674	106	28.00%	67.80%
CORN	www.cs.cornell.edu	1346	N/A$	14	14	24	18	45.42%	80.80%
UTEX	www.cs.utexas.edu	2935	N/A	11	10	11	10	45.02%	84.90%
WASH	www.cs.washington.edu	1526	N/A	16	10	23	10	79.04%	69.98%
WISC	www.cs.wisc.edu	2973	N/A	10	15	11	15	41.73%	77.08%
ABOUT	compnetworking.about.com	498	N/A	10	10	11	48	18.65%	43.54%
ECNET	reviews.cnet.com	500	N/A	10	10	10	10	38.89%	57.72%
ESPN	sports.espn.go.com	494	N/A	8	10	16	12	28.32%	58.39%
MONET	www.mo.net	261	N/A	9	10	9	13	35.78%	54.91%
XML	www.xml.com	807	N/A	10	10	10	26	43.46%	80.89%

*: Unmarked TOC pages are removed from the TOC answer set due to the error occurring when parsing their DOM trees.
#: Domain experts selected the article pages with different and distinctive tagging styles to be the article answer set.
$: We only select some TOC and Article pages containing different structures for performance evaluation. We do not find all answers in English Web sites.

$\mathrm{A}_A = \{a_{A1}, a_{A2}, \ldots, a_{An}\}$. The extracted results are the set of TOC blocks $W_T = \{w_{T1}, w_{T2}, \ldots, w_{Tn}\}$ and the set of article blocks $W_A = \{w_{A1}, w_{A2}, \ldots, w_{An}\}$.

We show the result of the incremental entropy calculation in Fig. 11. In the figure, the value of the Y-axis means the ratio of the resulting entropy and the final entropy value calculated from the whole page set. We can find that the ratio difference is smaller than 0.1 when the corresponding document count is larger than 200. Therefore, in the practical usage, WISDOM can achieve a stable performance when the crawled page set is smaller than the whole page set of a Web site.

4.2 Evaluation of *k*-MIB

After the *ICT* of a page is built, we have to determine the searching (f_A) and branching (*ST*) criteria before applying *k-MIB* to the *ICT*. These selection criteria of *k-MIB* will affect the performance of the algorithm. In Fig. 12, we first conduct experiments to show the effects of different selection criteria for TOC blocks. We select $ALEN_A$ and API_A for the searching criteria and corresponding *SII* values for the branching criteria. The result in Fig. 12 shows that using $SII(API_A)$ for the branching criterion outperforms the one using $SII(ALEN_A)$ when the selection criterion is to use a threshold of being equal to or smaller than 0.8. This is because API_A contains more information for discriminating the informative and redundant links than the length of anchor texts does.

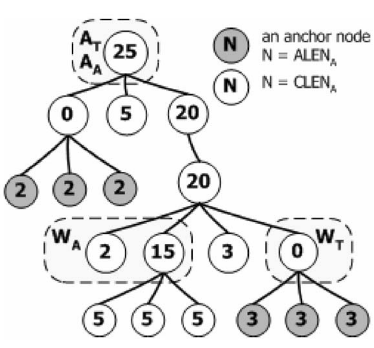


Fig. 10. A simple tree with an answer node and two results marked.

We then apply the *k*-MIB algorithm to the *ICT* with the parameter pair (k, f_A, ST). We use different *ST* values to control the number and granularity of the *IB*s. When the *ST* value is larger, more tighter and smaller blocks will be induced as shown in Fig. 13. Note that the average size of *IB*s in CTIMES is about three times as others and is out of the boundary of Fig. 13. This is due to the existence of big TOC blocks with entries of all categories of news in CTIMES. The sizes of these blocks are about 800 tags (nodes).

In the second phase of WISDOM, type constraint filtering plays an important role to remove the false-positive nodes. The selection of TC_{TOC} and $TC_{Article}$ is made as follows: The distributions of the average *API* values, i.e., the criterion of the TOC type constraint, of top-k *IB*s in UDN are shown in

TABLE 3
The Evaluating Calculation of the Example in Fig. 10

Method	(A_T, W_T)						(A_A, W_A)					
	AW	AO	WO	P	R	F	AW	AO	WO	P	R	F
SNC	3	3	0	1	0.50	0.67	5	5	0	1	0.50	0.67
IC	9	6	0	1	0.60	0.75	17	8	0	1	0.68	0.81

*AW=the number of answer of the intersection of A and W
*AO=the number of answer in A but not in W
*WO=the number of answer in W but not in A
* P = AW/(AW+WO), R=AW/(AW+AO)

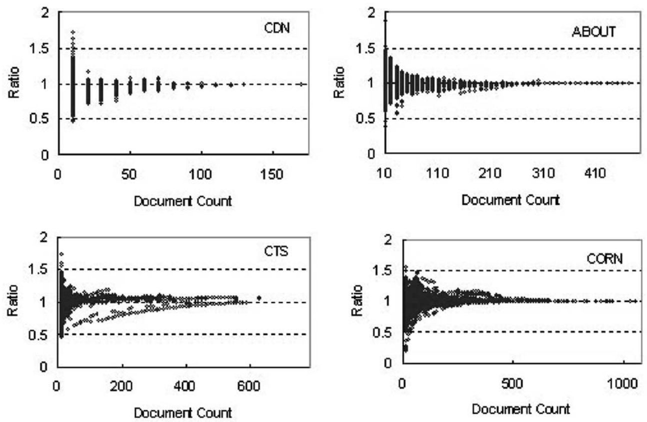


Fig. 11. Incremental entropy distribution for data sets CDN, CTS, ABOUT, and CORN.

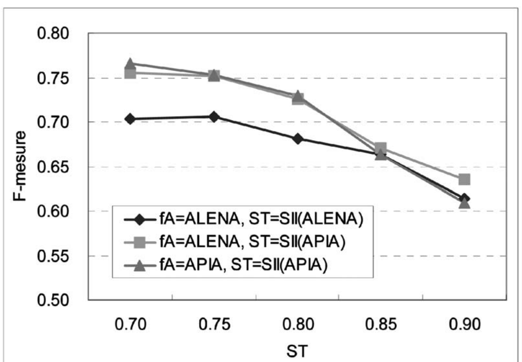


Fig. 12. The effect of different criteria of *k-mib* for TOC blocks.

Fig. 14 where it can be seen that there are two obvious noise groups of values in this figure, i.e., 1.1 and 2.2, and they are reasonably chosen to be the TC_{TOC}. The selection of $TC_{Article}$ is not so straightforward as the selection of TC_{TOC}. This is because when the size of *IB*s is divided into smaller ones, the number of extracted terms in each small block decreases, so does the accuracy of corresponding *CII*. Moreover, the index *CII* is not an aggregated value. We choose the uninformative link threshold described in [16], i.e., 0.8, to be $TC_{Article}$. Consequently, we use (1.25, 0.8) as default values for $(TC_{TOC}, TC_{Article})$ for all data sets in WISDOM. The choice of TC_{TOC} value, 1.25, is simply motivated from the *API* formulation. We assume that each basic informative link contains one matching term with entropy 0.8, and its API value is 1.25 by the formulation. The value conforms to our observation on the real data shown in Fig. 14.

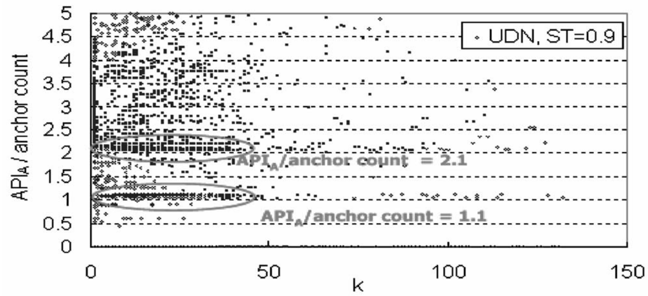


Fig. 14. The selection of the type constraint of TOC blocks [UDN, ST = 0.9].

We show the average precision and recall values of k-MIB under the different selections of *ST* and k in Fig. 15. The results of TOC and article blocks both show the phenomena incurred by *ST*. When the value of ST increases, the sizes of split *IB*s decrease and the granularities of these blocks become finer. The selection of more fine-grained *IB*s increases the precision, but reduces the coverage of *IB*s, i.e., the recall. The same observation can be made in the same figure when the value of k becomes smaller.

From the results in Fig. 15, WISDOM is good at mining the informative article blocks rather than TOC blocks. First, there exists only one informative article block in most marked article pages. The information of an article page is thus more concentrated than information of a TOC page. This helps WISDOM discriminate informative article blocks easily. Second, noises affect and blur the *API* value. Using entropy to indicate the amount of information does not work well when few terms are extracted from the anchor

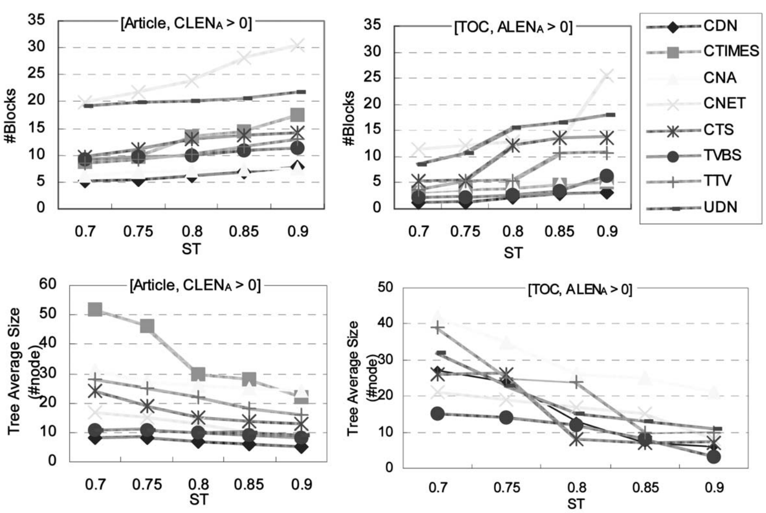


Fig. 13. The average number and size of total *IB*s in a page selected by k-MIB without filtering.

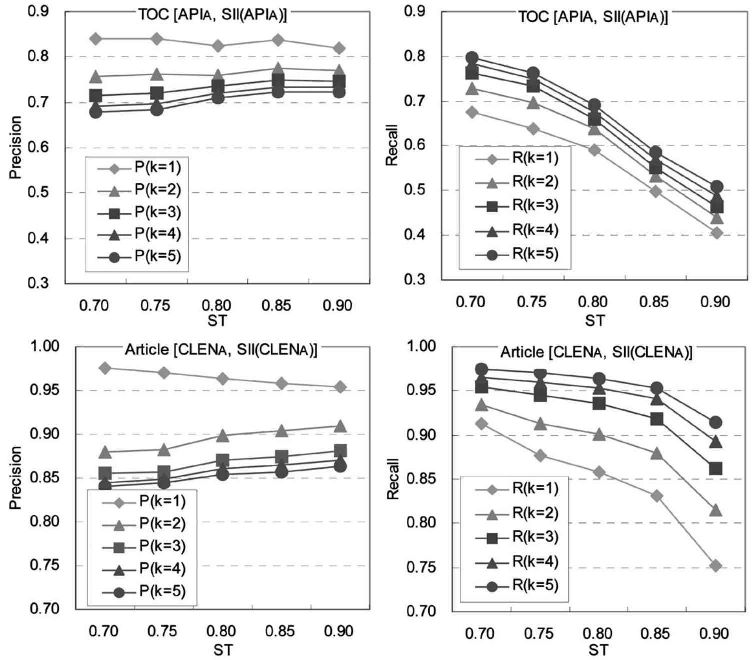


Fig. 15. The average values of precision and recall before phase 3 (caption: block type [f_A, *SII*]).

text. The local menu effect mentioned in [16] can also decrease the entropy values of anchors in the menu blocks. The noise effects are prominent in CNET and UDN and the discriminability of the *API* value in these Web sites decreases suddenly. In CNET, more than 68 percent of all *IB*s have average *API* values of less than 2 if $\mathrm{k} <= 5$. Third, some informative TOC blocks mined by WISDOM are not "news" TOC blocks. These blocks are not selected in the answer set. The effects of applying different TCs are shown in Fig. 16. Filtering constraints can be used to remove pseudoinformative blocks.

4.3 Evaluation of Block Expanding and Condensing

Fig. 17 shows that the average improvement of different merging methods. The performances of experiments with different STs become similar after block expansion. This is because most blocks extracted by a high ST value are real *IB*s, though the sizes of these blocks are smaller than blocks extracted by a low ST value. The sizes of these smaller blocks can be expanded to the sizes of larger blocks by merging sibling subtrees which are also real *IB*s. Merging methods do not work well if a skeleton set contains many pseudoinformative blocks, such as TOC blocks in CNET. Expansion of the skeleton set will incur more false-positive results. This is also the reason that the results with $\mathrm{k} = 1$ are better than those with $\mathrm{k} = 3$.

4.4 Overall Performance

In Fig. 18, we use the system default setting, i.e., $\mathrm{k} = 1$, $\mathrm{ST} = 0.8$, $\mathrm{TC} = (0.8, 1.25)$, and merging methods DSTM and 1-CSTM, to show the overall performance of WISDOM on each data set. This figure shows that WISDOM is very good at the article blocks mining of all data sets and exhibits excellent performance on TOC blocks mining of CDN, CTIMES, CNA, CTS, and TVBS. The low values of precision and recall on CNET, TTV, and UDN are caused by the low accuracy of API values. Another low precision value on UDN is affected by the merging method 1-CSTM. Many pseudoinformative blocks are merged in the 1-CSTM step, even though WISDOM has reached the high recall rate after DSTM merging. The high average values of precision and recall also represent the robustness of WISDOM. We also compare WISDOM with two straightforward extracting methods in Fig. 19 to show the improvement. The method

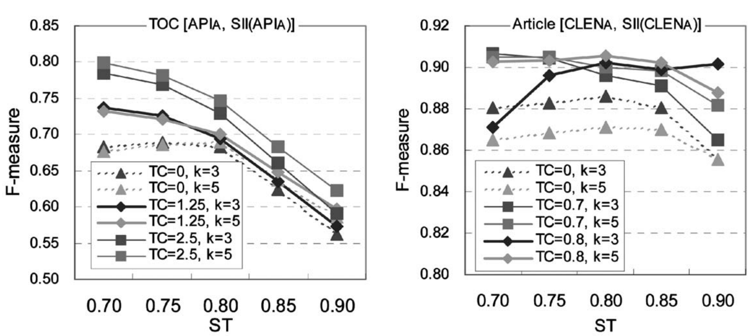


Fig. 16. The effects of different type constraints.

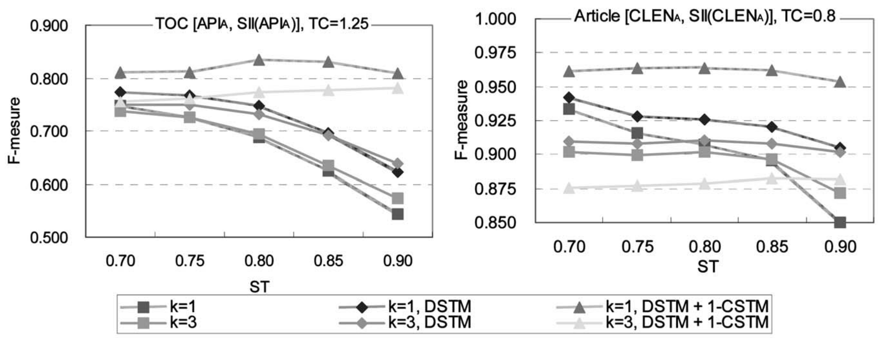


Fig. 17. The effects of DSTM and 1-CSTM.

M1 selects and merges leaf nodes of a DOM tree to a set of subtrees and is similar to the straightforward method described in Section 2. These merged leaf nodes must satisfy the same information constraint of WISDOM. The method M1 can be treated as simplified WISDOM that selects all leaf nodes into the skeleton set in the k-MIB phase. The method M2 works like M1 as well. The difference is that the method M2 uses the length constraint to filter the merged leaf nodes, i.e., the TLEN of a node must be larger than 5. Fig. 19 shows that WISDOM with the default setting leads these two methods and gives the prominent performance for the article pages.

The result in Fig. 20 shows the overall performance for English Web sites which consist of university and commercial domains by using the default setting same as in news Web sites. The performance for the article block extraction is also good as that in news Web sites. However, the performance of the TOC block extraction is worse than that in news Web sites. We found three reasons to cause the negative effect, which are 1) some informative anchors contains short anchor-text and common terms between anchors and linking pages are few. We cannot extract the anchor information in these cases and the feature APIs of these informative anchors therefore cannot be discriminated from redundant ones. 2) Some important terms are considered as stop-words and ignored, e.g., course-id in the university course pages. WISDOM ignores the numerical terms to reduce the noise effect caused by their high weights, which are obtained from the entropy calculation. However, course-id is an important clue for users to choice which course they feel interesting. This evidence shows that the stop-words selection must be dependent on the domain characteristics, otherwise, some important words will be ignored. 3) Many commercial anchors are generated by the script language embedded in the pages. WISDOM cannot extract the anchor texts from these dynamic links.

To remedy these issues, we conduct an experiment to use different features instead of API to extract TOC blocks. The result in Fig. 21 shows the improvement when the feature ALEN and corresponding ST threshold are applied on some data sets in which API does not work well. This can be explained by that the TOC structure characteristics are retained when features ALEN and SII(ALEN) are applied and blurred when API is not correctly calculated or hard to be evaluated due to the lack of matched terms. API values are always zero even though corresponding ALEN values are larger than zero in these situations. We can evaluate the ratio of numbers of zero-API anchors over all meaningful anchors, i.e., their ALEN values are more than some threshold, in a page to be our criterion on the selection of appropriate features. The average ratios of data sets in Fig. 21 are obviously higher than others from our experimental observations.

5 Conclusion

We propose WISDOM to mine the *ISs* of a page. Given an entrance URL, WISDOM is able to crawl the site, parse pages into DOM trees, and analyze node and structure information in order to build information coverage trees. The system uses the Information Theory to split the DOM tree of a Web page into a set of *IB*s and uses the proposed searching (*k-MIB*), filtering, and merging (*DSTM*, *1-CSTM*) methods to mine the *IS* of a page.

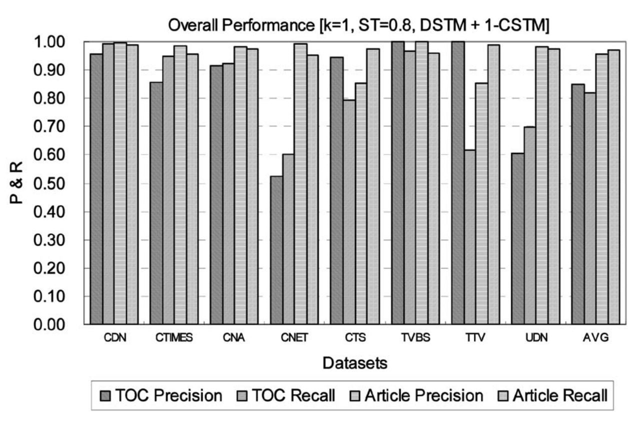


Fig. 18. Overall performance of WISDOM.

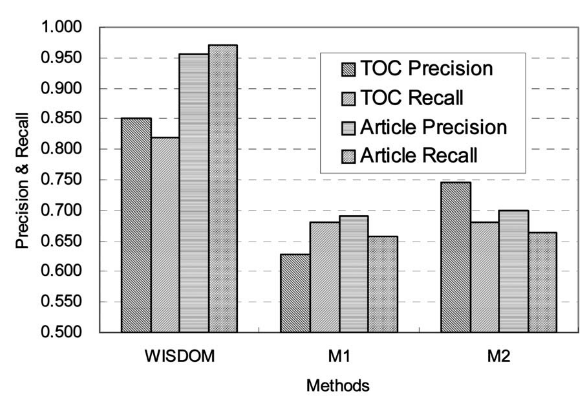


Fig. 19. Comparison of WISDOM to two straightforward methods.

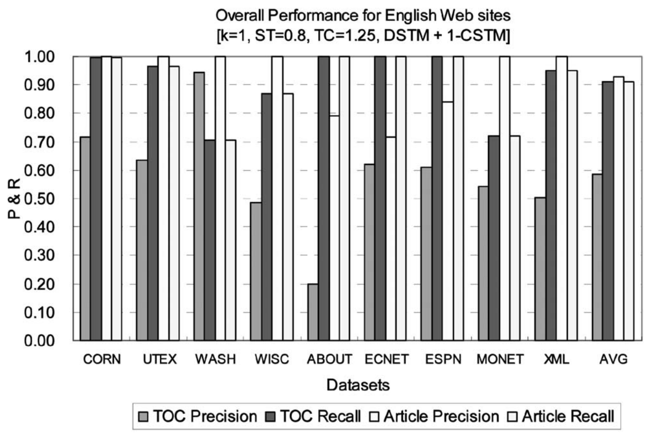


Fig. 20. Overall performance of WISDOM on other domain Web sites.

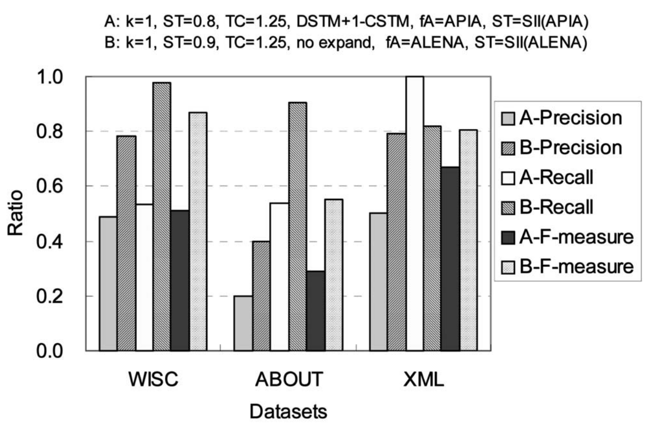


Fig. 21. Different feature selection for the TOC extraction.

For search engines, intermedia information agents, and crawlers, the *IS* mined by WISDOM is useful for indexing, extracting, and navigating significant information from a Web site. Experiments on several real news Web sites show high precision and recall rates attained by WISDOM which validates its practical applicability on news Web sites. We are integrating WISDOM into our news search engine (NSE) to help system managers speed up their work flow and reduce the labor of maintaining the site-dependent rule based extraction. For Web sites in other domains, even for nonsystematic Web sites, we are conducting some augmented feature to remedy the noise effects and improve the applicability.

Acknowledgments

The work was supported in part by the National Science Council of Taiwan, R.O.C., under Contracts NSC93-2752-E-002-006-PAE. H.-Y. Kao was with the Department of Electrical Engineering, National Taiwan University when this work was done.

References

[1] B. Adelberg, "NoDoSE—A Tool for Semi-Automatically Extracting Structured and Semistructured Data from Text Documents," *Proc. 1998 ACM SIGMOD Int'l Conf. Management of Data (SIGMOD),* 1998.

[2] T. Asai, K. Abe, S. Kawasoe, H. Arimura, H. Sakamoto, and S. Arikawa, "Efficient Substructure Discovery from Large Semi-structured Data," *Proc. SIAM Int'l Conf. Data Mining (SDM),* 2002.

[3] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval.* Addision Wesley, 1999.

[4] Z. Bar-Yossef and S. Rajagopalan, "Template Detection via Data Mining and Its Applications," *Proc. 11th World Wide Web Conf. (WWW),* 2002.

[5] A. Broder, S. Glassman, M. Manasse, and G. Zweig, "Syntactic Clustering of the Web," *Proc. Sixth World Wide Web Conf. (WWW),* 1997.

[6] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery, "Learning to Construct Knowledge Bases from the World Wide Web," *Artificial Intelligence,* vol. 118, nos. 1-2, pp. 69-113, 2000.

[7] S. Chakrabarti, "Integrating the Document Object Model with Hyperlinks for Enhanced Topic Distillation and Information Extraction," *Proc. 10th World Wide Web Conf. (WWW),* 2001.

[8] Y. Chen, W.-Y. Ma, and H.-J. Zhang, "Detecting Web Page Structure for Adaptive Viewing on Small Form Factor Devices," *Proc. 12th World Wide Web Conf. (WWW),* 2003.

[9] W. Cohen, "Recognizing Structure in Web Pages Using Similarity Queries," *Proc. Nat'l Conf. Artificial Intelligence (AAAI),* 1999.

[10] G. Cong, L. Yi, B. Liu, and K. Wang, "Discovering Frequent Substructures from Hierarchical Semi-Structured Data," *Proc. SIAM Int'l Conf. Data Mining (SIAM SDM),* 2002.

[11] R. Cooley and J. Srivastava, "Web Mining: Information and Pattern Discovery on the World Wide Web," *Proc. Ninth IEEE Int'l Conf. Tools with Artificial Intelligence (ICTAI),* 1997.

[12] D.W. Embley, Y. Jiang, and Y.K. Ng, "Record-Boundary Discovery in Web Documents," *Proc. 1999 ACM SIGMOD Int'l Conf. Management of Data (SIGMOD),* 1999.

[13] K. Furukawa, T. Uchida, K. Yamada, T. Miyahara, T. Shoudai, and Y. Nakamura, "Extracting Characteristic Structures among Words in Semistructured Documents," *Proc. Sixth Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD),* 2002.

[14] H. Grundel, T. Naphtali, C. Wiech, J.-M. Gluba, M. Rohdenburg, and T. Scheffer, "Clipping and Analyzing News Using Machine Learning Techniques," *Proc. Int'l Conf. Discovery Science,* 2001.

[15] C.N. Hsu and M.T. Dung, "Generating Finite-State Transducers for Semi-Structured Data Extraction from the Web," *Information Systems,* vol. 23, no. 8, pp. 521-538, 1998.

[16] H.-Y. Kao, S.H. Lin, J.M. Ho, and M.-S. Chen, "Entropy-Based Link Analysis for Mining Web Informative Structures," *Proc. ACM 11th Int'l Conf. Information and Knowledge Management (CIKM),* 2002.

[17] H.-Y. Kao, S.-H. Lin, J.-M. Ho, and M.-S. Chen, "Mining Web Information Structures and Contents Based on Entropy Analysis," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 1, Jan. 2004.

[18] J.M. Kleinberg, "Authoritative Sources in a Hyperlinked Environment," *Proc. ACM-SIAM Symp. Discrete Algorithms (SODA),* 1998.

[19] N. Kushmerick, D. Weld, and R. Doorenbos, "Wrapper Induction for Information Extraction," *Proc. 15th Int'l Joint Conf. Artificial Intelligence (IJCAI),* 1997.

[20] A. Laender, B. Ribeiro-Neto, A. Silva, and J. Teixeira, "A Brief Survey of Web Data Extraction Tools," *SIGMOD Record,* vol. 31, no. 2, June 2002.

[21] S.H. Lin and J.M. Ho, "Discovering Informative Content Blocks from Web Documents," *Proc. Eighth ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD),* 2002.

[22] W.Y. Lin and W. Lam, "Learning to Extract Hierarchical Information from Semi-Structured Documents," *Proc. ACM Ninth Int'l Conf. Information and Knowledge Management (CIKM),* 2000.

[23] X. Li, B. Liu, T.-H. Phang, and M. Hu, "Using Micro Information Units for Internet Search," *Proc. ACM 11th Int'l Conf. Information and Knowledge Management (CIKM),* 2002.

[24] T. Miyahara, Y. Suzuki, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda, "Discovery of Frequent Tag Tree Patterns in Semistructured Web Documents," *Proc. Sixth Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD),* 2002.

[25] C.E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical J.,* vol. 27, pp. 398-403, 1948.

[26] G. Salton, *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer.* Addison Wesley, 1989.

[27] W3C DOM, Document Object Model (DOM), http://www.w3.org/DOM/, 2005.

[28] K. Wang and H. Liu, "Discovering Structural Association of Semistructured Data," *IEEE Trans. Knowledge and Eng.*, vol. 12, no. 3, May/June 2000.
[29] C. Yip, C. Gertz, and N. Sundaresan, "Reverse Engineering for Web Data: From Visual to Semantic Structures," *Proc. 19th IEEE Int'l Conf. Data Eng. (ICDE)*, 2002.

Hung-Yu Kao received the BS and MS degrees in the Department of Computer Science from National Tsing Hua University (NTHU), Hsinchu, Taiwan, in 1994 and 1996, respectively. In July 2003, he received the PhD degree from the Electrical Engineering Department, National Taiwan University, Taipei, Taiwan. From 1996 to 2003, he was a research assistant in the Institute of Information Science (IIS), Academia Sinica. He was a postdoctoral fellow of IIS from 2003 to 2004. Dr. Kao is currently an assistant professor of computer science and information engineering at National Cheng Kung University (NCKU). His research interests include Web information retrieval/extraction, knowledge management, data mining, bioinformatics, and content network.

Jan-Ming Ho received the BS degree in electrical engineering from National Cheng Kung University in 1978, the MS degree from the Institute of Electronics at National Chiao Tung University in 1980, and the PhD degree in electrical engineering and computer science from Northwestern University in 1989. He joined the Institute of Information Science, Academia Sinica, Taiwan, R.O.C., as a associate research fellow in 1989, and was promoted to research fellow in 1994. He visited the IBM T.J. Watson Research Center in the summer of 1987 and 1988, the Leonardo Fibonacci Institute for the Foundations of Computer Science, Italy, in the summer of 1992, and the Dagstuhl-Seminar on combinatorial methods for integrated circuit design, in October 1993. He is a member of the IEEE and the ACM. His research interests are targeted at the integration of theoretical and application-oriented research, including mobile computing, environment for management and presentation of digital archive, management, retrieval, and classification of Web documents, continuous video streaming and distribution, video conferencing, real-time operating systems with applications to continuous media systems, computational geometry, combinatorial optimization, VLSI design algorithms, and implementation and testing of VLSI algorithms on real designs. He is an associate editor of *IEEE Transaction on Multimedia*.

Ming-Syan Chen received the BS degree in electrical engineering from National Taiwan University, Taipei, Taiwan, and the MS and PhD degrees in computer, information, and control engineering from The University of Michigan, Ann Arbor, Michigan, in 1985 and 1988, respectively. Dr. Chen is currently a professor in the Electrical Engineering Department, a professor in the Computer Science and Information Engineering Department, National Taiwan University, Taipei, Taiwan, and the chairman of the Graduate Institute of Communication Engineering. He was a research staff member at IBM Thomas J. Watson Research Center, Yorktown Heights, New York, from 1988 to 1996. His research interests include database systems, data mining, mobile computing systems, and multimedia networking, and he has published more than 180 papers in his research areas. In addition to serving as program committee members in many conferences, Dr. Chen served as an associate editor of the *IEEE Transactions on Knowledge and Data Engineering* from 1997 to 2001, is currently on the editorial board of several journals, and is a distinguished visitor of IEEE Computer Society for Asia-Pacific from 1998 to 2000. He served as the program chair, program vice-chair, and general chair of several conferences. He was a keynote and tutorial speaker at several conferences, and also a guest coeditor for the *IEEE Transactions on Knowledge and Data Engineering* special issue on data mining in December 1996. He holds, or has applied for, 18 US patents and seven ROC patents in the areas of data mining, Web applications, interactive video playout, video server design, and concurrency and coherency control protocols. He is a recipient of the NSC (National Science Council) Distinguished Research Award and K.-T. Li Research Penetration Award for his research work, and also the Outstanding Innovation Award from IBM Corporate for his contribution to a major database product. He also received numerous awards for his research, teaching, inventions, and patent applications. Dr. Chen is a fellow of the IEEE and a member of the ACM.