Chapter 8

The Conversion of a Web Site into a HyperContext Hypertext

8.1 Introduction

Constructing a HyperContext hypertext by converting a World Wide Web site proved to be an enormous undertaking, ultimately requiring significantly more time and effort than the actual implementation of the HyperContext prototype. The HyperContext hypertext is required to provide evidence that the HyperContext approach to adaptive hypertext is feasible and that it can provide benefits to its user community.

This chapter describes the process of converting a Web site into a HyperContext hypertext. A HyperContext Web Robot was designed to mirror a Web site onto a UNIX host called zeus.cs.um.edu.mt. The HTML files (documents) were processed to determine each document's in-links and out-links. Each of a document's in-links forms a context for which an interpretation of the document is required. The profile, interpretations, and links files for each document are automatically created. SWISH-E, a third-party indexing and retrieval system, briefly described in Section 8.4, is used to generate the interpretations files and merges these files to create a central index.

The experimentation on HyperContext requires that a control version is available against which results can be compared. Chapter 7.8 describes how modifications to the HyperContext client can reduce an adaptive HyperContext hypertext to a non-adaptive equivalent of a typical World Wide Web hypertext.

8.2 Preparing the mirror site

8.2.1 The HyperContext Web Robot

The HyperContext Web Robot HCTRobot downloads a Web site by following links from a nominated start document on the Web until there a no more links to follow or some limit has been reached. We limited the Web robot to downloading a maximum of 200 Web documents.

HCTRobot contains a simple HTML parser which identifies a document's out-links (HTML hypertext reference tags), converts the out-links into fully-qualified URLs, and extracts the out-links' source anchor text. HCTRobot traverses the Web breadth-first by maintaining and processing a download queue of unique URLs extracted from downloaded documents. The robot can also be configured to retrieve documents from specific Web servers only. The robot was restricted to downloading documents from www.w3.org to construct the experimental HyperContext hypertext.

HCTRobot maintains a record of each document's out-links as a destination URL and anchor text pair. Each record is keyed by a document URL which is followed by a list of its children. By subsequently inverting each record, the parents of each downloaded document can be identified.

The HyperContext Web Robot is configured to download only HTML and TEXT files, so that at the end of this stage, the mirror site will contain a number of documents, organised in the same way as the source site, but under a different root directory structure. The record of each documents' children is stored as a text file for later processing.

Finally, the children are consistency checked, to ensure that all the files referred to have been downloaded. Typically, there will be a number of files which have not been downloaded, perhaps because the robot has already reached its download limit, or because a link is stale (the URL points to a document which no longer exists), or because the robot has been instructed not to access the server hosting the document. The children file produced by the HyperContext Web Robot must only contain references to files which have actually been mirrored. All URLs which refer to files missing on the mirror site are removed from the children file.

8.2.2 Determining a document's parents

In order to create a document's interpretations it is necessary to determine the contexts in which the document exists. The list of each document's children produced by the HCTRobot can be inverted to produce a list of each document's parents (as parent URL and anchor text pairs). In turn, this process identifies each document's contexts. As the parent and children files can be quite large, hash files are used to provide rapid access to their entries.

8.2.3 HTML file preparation for HyperContext

The HTML files that have been downloaded to the mirror site require some changes before they can be used within the HyperContext prototype. URLs which refer to documents not resident on the mirror site are removed from the HTML files. The remaining URLs are replaced with numbered HyperContext <HCT> tags, and the source anchor text of the valid links is marked up to be displayed as underlined blue text, so that it can be easily identified by a user as being anchor text. The <HCT> tag number is stored with the corresponding URL in the children and parent files.

8.3 Generating the HyperContext hypertext

The process of mirroring a Web site in preparation for converting it into a HyperContext hypertext involves downloading HTML files, processing the HTML files to identify their children and parents, modifying the HTML files to remove redundant and invalid out-links, replacing valid out-links with an <HCT> tag, and modifying the source anchor text to underline it and colour it blue. The next major phase in the Web-HyperContext conversion process is the generation of the HyperContext Object and Structure Layers by automatically generating profile, interpretations, and links files for each downloaded HTML file.

8.3.1 Generating the Object Layer

The Object Layer is generated by creating a profile file for each HTML document. Each profile file contains the associated document's URL (which, in this case is the original URL, not the URL of the mirrored file); the document's title (which is extracted from the <TITLE> tag in the HTML file); its file type (HTML or TEXT) and access method (HTTP); and the HyperContext tags and the anchor text bound to URLs, extracted from the children file produced in Section 8.2.1 and modified in Section 8.2.3. The anchor text

will become the label on which an interpretation is linked to a destination document. The profile is stored in the same directory as the corresponding HTML file.

8.3.2 Generating the Structure Layer

This step is the most time consuming part of the automatic Web to HyperContext conversion process. The parent file is processed to extract the contexts of each document. Each document will have a single interpretations file containing all the interpretations of the document, including an interpretation in the context bottom. In addition, each document will have an associated links file which contains a set of out-links for each interpretation. The out-links for an interpretation is a set of <HCT> tag number, anchor text (label), and destination document triples. The links file also contains the document name (URL) and title (copied from the document's profile). As specified in Chapter 6.4, a link is represented by the triple node-label-document. In the links file, node is the value of the name (URL) of the document containing the link source, label is a value representing the link source anchor text, and document is the destination document's URL. An interpretation is a vector of weighted terms (labels), which in the prototype is externally stored as an index file generated by SWISH-E.

The first interpretation of a document is created for the context bottom. This interpretation will contain all of the links which survived the validity checking process described in Section 8.2.1, and the labels selected to represent the document are the terms extracted from the document after they have been stemmed, using the Porter stemming algorithm [69]. The individual words in the source anchor text on which the destination documents are linked are stemmed, and the stop words are removed. The labels are indexed into the interpretations file.

Once an interpretation for the document in the context bottom has been created, the document's other contexts are extracted from the parent file. To generate an interpretation for a given context, the documents representing the source and destination of the link are compared to determine which terms in the destination document should be used to construct its interpretation. Both documents are divided into *blocks* of text (figure 8.1). In HTML, blocks can be identified through the location of the following tag types: <HTML>, <BODY>, <H1>, <H2>, ..., <H7>. These tags can be used to identify progressively smaller blocks within a document. Once the individual blocks in the two documents have been identified they are processed to remove all HTML tags and stop-words, and to stem all remaining words. The source anchor text is similarly processed. The blocks of the source document are then checked to identify those which contain any of the terms present in the source anchor text. Each block which contains at least one term

present in the anchor text is added to a *context block*. Once the source document blocks have all been processed, the resulting context block is compared with each of the blocks of the destination document, to identify those which share terms. All relevant blocks in the destination document are combined and indexed to form the document's interpretation for that context. When the destination document is segmented into blocks, a record is kept all the out-links present in each block. The out-links present in the blocks used to form a document's interpretation are stored in the links file for that interpretation.

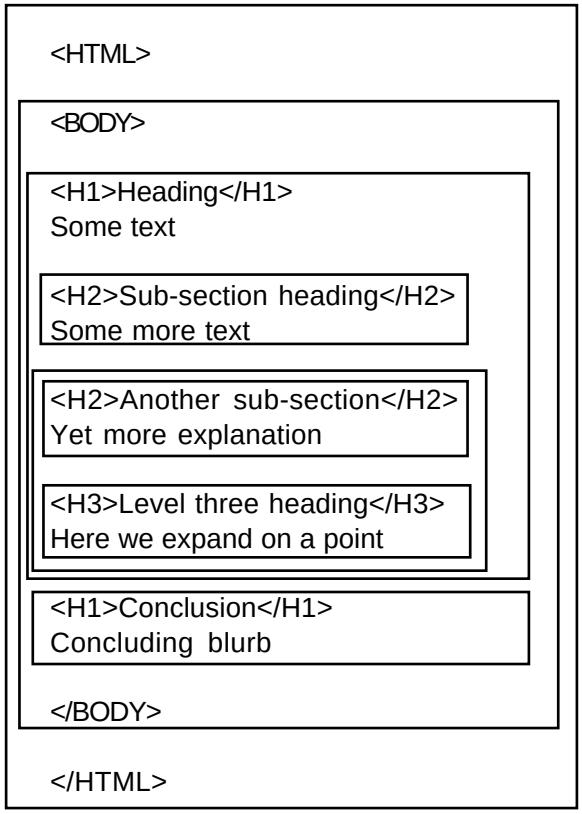

Figure 8.1: Dividing an HTML file into blocks

8.4 SWISH-E: Simple Web Indexing System for Humans - Enhanced

SWISH-E [48] is an enhanced Boolean information indexing and retrieval system designed to index HTML documents. Simple Boolean models of information retrieval cannot rank documents according to relevance, because they usually record only the presence or absence of terms in documents. SWISH-E, however, derives term weights which not only take into account term frequency with a document, but also the structural importance of a term. For example, if a term occurs in an HTML document's title or one or more of its headings, then it will be given a higher weight than a term which occurs only in paragraphs.

SWISH-E was chosen to provide the HyperContext with information retrieval services for the following reasons:

- SWISH-E is provided with C source code, which is freely modifiable;
- it provides a merge mechanism for combining indices to form a larger index;
- and, it provides a index decoding function which converts a hashed, encoded index file into a text file.

There were a number of bugs present in SWISH-E, especially in the index merging function. The declaration of some variables as static meant that they kept their values between separate calls from the HyperContext Java code. They, and other code segments, were modified to make SWISH-E re-entrant. The decoding mechanism displayed a decoded index to the process environment's output device (normally the screen). The source code was modified to output a decoded index to file so that it could be subsequently read and processed by HyperContext.

The ability to merge separate indices and decode indices is of great importance to the HyperContext prototype. The prototype externally stores interpretations as individual index files which contain all of the interpretations of a particular document. The central index used by all the search paradigms in the prototype is created by merging all of the individual indices representing the interpretations. An interpretation is extracted by decoding the appropriate interpretations file for a document and then generating a vector of all terms which occur in the given context, together with the terms' weights. The structure of a decoded interpretations file is described in Section 8.5. The same section also describes how the interpretations file is created.

SWISH-E is called from HyperContext by invoking a separate child process which HyperContext then waits for.

8.5 The structure of the interpretations file

The structure of a typical decoded SWISH-E index file is shown in figure 8.2. An encoded index file is compressed using a hash function to conserve space. The first few lines of the decoded index file identify the index file, the indexing system, the person responsible for the index, and some accounting information.

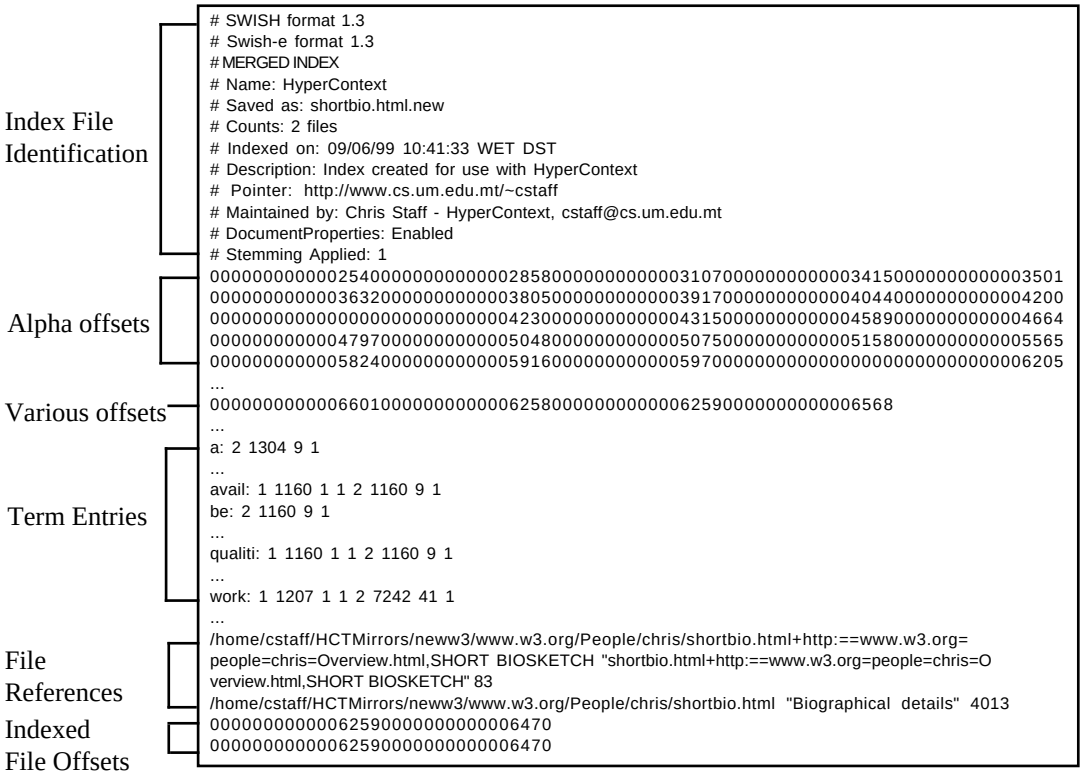

Figure 8.2: A SWISH-E decoded index file structure (edited for brevity)

A typical index term entry contains the index term followed by sets of four values. The first two values represent the file containing the term and the term weight. The last two values represent the HTML structures within which the term occurs in the file, and whether a meta-name is associated with the term. In SWISH-E it is possible to restrict the relevance of terms to particular structural locations within an HTML document. For example, a user may decide that a term is relevant only if it occurs in the document's title (which means it must occur between the <TITLE> </TITLE> HTML tags). The third value in the set represents all the structures within which a term occurs using the following values: IN_FILE = 1, IN_TITLE = 2, IN_HEAD = 4, IN_BODY = 8, IN_COMMENTS = 16, IN_HEADER = 32, IN_EMPHASIZED = 64, IN_ALL = 127. The value 41 would represent a term which occurred at least once within a section heading (32), within the <BODY> of an HTML document (8), as well as anywhere within the file (1). HyperContext does not make use of a term's structure value or meta-name.

Following the file information, the character offsets of the first term for each letter of the alphabet are represented as a stream of 16 digit integers. The next entry, the various offsets, shows the locations of the end of file marker, the end of the term list, the start of the file references, and the location of the indexed files offsets. This is followed by the list of indexed terms. This index file contains the interpretations of the document

www.w3.org/People/chris/shortbio.html in the contexts of bottom and www.w3.org/people/chris/Overview.html,SHORT BIOSKETCH, which are identified by the file references. The delimiter in the path names of the parent files providing the contexts is "=", rather than "/" for reasons explained in Section 8.6. A file reference also includes the name of the file (or the title of the document, if available) enclosed by quotation marks, and the file size in bytes. These entries are space delimited in the file reference. Finally, the indexed file offsets gives the offset of the name of each indexed file in the file references.

8.6 Creating the interpretations files

As described in Section 8.3.2, the interpretation of a document in the context bottom is created by requesting that SWISH-E indexes the specified HTML file. Each subsequent interpretation of the document is created by extracting terms that describe the document in that specific context, and indexing the terms. The index file is saved to a file named by the name of the HTML file which has the name of the context concatenated to it separated by a "+". In order to create a file with this name, it is necessary to replace the path delimiter "/" in the name of the parent file with "=", which explains the presence of "=" in the name of the context. After an index file representing the interpretation of a document in a single context is created, it is merged with the interpretations file for that document, and then deleted.

8.7 HyperContext hypertext statistics

The HyperContext hypertext used for experimentation has 170 nodes. Although this is quite a small hypertext, the conversion process took several weeks, as conversion is dependent not only on the number of nodes, but also on the number of interpretations that must be created. The 170 nodes are highly connected. On average a node has 85 in-links and 86 out-links, which in HyperContext means that each document has on average 86 interpretations (one for each in-link and one for the context bottom), and each document provides on average 86 contexts. The total number of interpretations in the hypertext is 4187. The spread of contexts (excluding the context bottom) is presented in table 8.1.

From table 8.1 it can be seen that the largest number of nodes have been 1 and 10 contexts each. 32 nodes have in excess of 100 contexts. These nodes are typically the root node, and some documents which provide copyright information. Many of the contexts are also self-referential, in that an HTML document may contain several navigational within-document links for rapid relocation within the document. As the conversion process is so time consuming, we decided to create separate interpretations of

documents only if the document had less than 31 contexts. In the event that the document has more than 30 contexts, the document will be interpreted in the context bottom whenever it is accessed. 50 of the 170 documents have an interpretation in the context bottom only.

Number of contexts	Number of nodes
1 - 10	93
11 - 20	17
21 - 30	10
31 - 40	3
41 - 50	1
51 - 60	4
61 - 70	1
71 - 80	1
81 - 90	7
91 - 100	1
more than 100	32

Table 8.1: Number of contexts per node

Table 8.2 shows the number of out-links from each node. Only eight nodes (two of which are TEXT documents) have no out-links. Once again, many of the out-links are navigational within-document links.

Number of out-links	Number of nodes
0	8
1 - 10	64
11 - 20	32
21 - 30	11
31 - 40	7
41 - 50	6
51 - 60	0
61 - 70	1
71 - 80	3
81 - 90	0
91 - 100	2
more than 100	34

Table 8.2: Number of out-links per node

In hindsight, a Web site with different characteristics would have been a better hypertext to use for experimentation. It would have been more suitable to select a Web site which had a lower average number of in- and out-links so that more nodes could have been downloaded and converted within the same time-frame. However, the greatest advantage that the selected Web site had in its favour is that a standard version of HTML was used consistently and correctly. Previous attempts to convert other Web sites were

unsuccessful because of the incorrect use of HTML, the use of HTML extensions which are browser dependent, or extensive reliance on technologies such as JavaScript. Of course, the automatic conversion of a Web site into a HyperContext site does not yield a HyperContext site which is as expressive as a HyperContext hypertext constructed by human users over time. The experimental site, for example, does not contain examples of labels which lead to one destination document in one interpretation, and another document in another interpretation. In the experimental hypertext, if a linked label is included in an interpretation of a document it will always lead to the same destination document. Interpretations are created by selecting those terms in a document which co-occur with terms in the link source anchor's context block (Section 8.3.2). In the HyperContext framework, a user creates an interpretation by selecting terms which describe how the document is relevant to him (Chapter 5.2). A user-created interpretation is likely to be significantly more useful (both to other users and to Adaptive Information Discovery) than an interpretation which is derived by comparison with the document's parent.

Despite these shortcomings, the experimental HyperContext hypertext is still very useful as a first step process. With an implementation of the HyperContext framework that enables users to create and modify interpretations, the hypertext would eventually reflect the needs and requirements of its community of users.