Indexing the World

Ian Cooper (ihc@ukc.ac.uk)
Computing Laboratory
University of Kent at Canterbury
Canterbury, Kent CT2 7NF
United Kingdom

Telephone: +44 227 764000. Facsimile +44 227 762811

July 9, 1994

Abstract

The World Wide Web provides readers with a potentially effortless way to retrieve and view related materials from around the world. Although readers can reach many documents by stepping through explicit links, there is no convention to show what other related documents exist. Readers must know addresses of appropriate starting points for exploration.

Indices (such as Jumpstation and ALIWEB) and Web roaming applications (such as the Mosaic-fish 'spider') are currently used to try and alleviate these problems. In their current forms these tools will only be useful while the Web is relatively small, and the user base tolerant.

I propose that Uniform Resource Name resolution services provide an ideal location to serve indices. With reference to the work of Sheldon et al. [Sheldon94], I suggest that smaller indices linked to form a huge virtual index space provide a functional scalable method of indexing individual documents. The associative access provided by the system enables implicit links to be identified - essential in a dynamic hyperspace such as the Web.

1 Notes to the reader

This document has several differences from the original hypertext available at,

```
http://www.ukc.ac.uk/Html/Pubs/IHC10-94/
```

In particular, the section dealing with caching is now included within the main body of the text and may therefore appear 'out of sequence'. Some references may also appear to be missing – these were online references in the original hypertext.

2 Introduction

The ability to locate information is the single most important feature of the World Wide Web requiring research. If readers cannot locate information of interest, other issues are irrelevant - even navigation through the relatively few documents we currently have available can cause problems (why else have indices appeared?). If we do not rapidly address this issue, the Web must fail to meet its full potential.

Readers will be understandably reluctant to navigate through a hypertext looking for a particular piece of information, and may never find the indices that are currently available.

We need to help users stop getting trapped in the Web when they are looking for information. Hammond suggests that readers of hypertexts with indices look at more than those without, but that those without indices *feel* that they have seen more [Hammond91].

3 Why do we need to index the Web?

The Web is a hypertext of information, with links between related material. Surely, if links are provided, users will be able to follow them to find the information that they want? Why do we need to provide additional overheads in the form of indices?

In a small hypertext, many of the available threads (sets of connected documents) can be located by a short period of browsing through the available lines. In strong contrast, the Web is a large, anarchic, dynamic hypertext. It can can take a long time to find documents of interest just by following the links specified by different authors. Indices are needed to locate individual documents or suitable starting points.

I believe that there are three questions that users of the World Wide Web will ask:

1. I'm interested in ..., what is there about this?
2. I'm looking at something interesting. What else have you got on this subject?
3. What have you got information on?

3.1 What's out there?

Information that may be of use to a particular reader may be available, but the address of an appropriate starting point is not necessarily known. The default page presented to a reader may have links that have nothing to do with their interests, and would therefore seem unlikely to have any relation to the subject matter that they might be looking for. These users may be unwilling to spend time browsing through the documents hoping that they might find something useful.

Existing hypertexts, which tend to be located on a single filing system, are usually created by a single author or group of collaborating authors. Access to the various 'nodes' can be planned by providing embedded links, tables of contents, and indices. In contrast, by its very nature, the Web cannot have any such planning, since each document is an independent entity, maintained by an individual author.

3.2 What else exists?

The Web was developed to link related material from different information sources. While these links had been implied (such as the ftp address of a paper or application being quoted in a UseNet News article) the Web has provided a consistent way of explicitly specifying them so that readers are able to use tools to follow them.

The current use of explicit links implies that the Web is static. On the contrary, the Web is highly dynamic with new information constantly becoming available (and 'old' information being removed). While an author can include links to related documents that exist at the time of writing, new documents may become available in the future. How can we provide access to these?

While readers control which of the presented links they wish to follow - in order to retrieve more information on a topic in which they are interested - the *author* of the document controls which, of a possible multitude of links, the user is able to follow. Ideally it is the reader's decision which links should be followed, but this assumes that they are able to locate all possible destinations. While annotations may allow readers to create personal links to other documents that they have found, these cannot be added 'in-line' as traditional links are, and may therefore be presented out of context.

While authors are able to specify explicit links between documents, there may be other links of which the author may be unaware. The current environment provides no automatic method of identifying or following such implicit links.

3.3 Requirements

In order to allow readers to use the Web as a serious source of information, I believe that there are three requirements which need to be met.

1. Ability to locate information by finding any document available on the Web.
2. Ability to locate documents that are related to the current one, irrespective of any explicit links.
3. Facilities to help users look for information without having to return periodically to an index.

4 What tools are currently available?

What tools are available for information location on the Web? What facilities do they provide and what are their shortcomings?

This section considers current technology. While in the future some of the shortcomings may not be so relevant (super-fast networks, immense backing storage capabilities), we cannot assume that technology will solve all of the problems.

4.1 Web roaming applications

Web roaming applications (or ’spiders’) such as Mosaic-fish use the link information embedded in hypertext documents to locate, retrieve, and locally scan as many documents as possible for keywords entered by the reader. Using the embedded link information, a high proportion of available hypertext documents are likely to be searched. However, since links are usually only specified when the destination is believed to exist, links to very new documents may not exist and new information may therefore be missed. Further, since some servers may be unreachable when the spider is run (due to network or server downtime) whole sections of the hypertext may be missed.

While such applications may be used infrequently by individuals in order to locate up-to-date references, they waste resources by transferring redundant information. If two users from the same site perform individual searches there is a possibility of the search spaces overlapping and the same information being returned to each user - wasting system resources. While use of these applications was not have considered harmful at the time of their inception, if there were relatively few users reading a small number of documents, they will cause network congestion and high machine load if many readers use them simultaneously. Caches (such as Lagoon) may help to partially alleviate this problem, by allowing subsequent users to load files from a local store.

Spiders may be used by sites in order to build indices to a large proportion of the documents available on the Web (Jumpstation). In order to keep indices up-to-date, sites will run spiders more regularly than individual users.

Spiders can aid readers in respect of questions 1 and 2 depending on whether links to documents can be located during the search. Given an appropriate search term, requirements 1 and 2 may be met. Requirement 3 is not met.

4.2 Indices

The World Wide Web has a number of indices which readers may use to locate information. Current indices fall into two categories:

1. the index stores details on as many documents as possible, striving to index every available document, e.g. w3catalog, Jumpstation;
2. the index stores information on various services, (such as a ’Table of Contents’), providing the reader with a starting place to look for more information, e.g. ALIWEB.

With various indices available, readers must choose which one to use. While interfaces (such as SUSI, and the CUI meta-index) can help by removing the need to locate an index in the first place (the user still needs to find the interface itself!), they do not provide readers with information about which services the indices offer. For example, should a query be sent to ALIWEB or the w3catalog (which contains the ALIWEB data)? What is the query searched against - titles, headers, body text or some ’third party’ abstract?

4.2.1 Indices to everything

Indices to everything (such as Jumpstation, w3catalog) can happily exist when there are relatively few hypertext documents being indexed (a recent scan located approx. 100,000 multimedia documents (World Wide Web Worm, Jumpstation statistics)). However, this model does not scale well and is limited in functionality.

Users are dependent on a single site to provide access to the index. Should that one site be unavailable, the whole index becomes unavailable. As the Web grows it is unlikely that a single site will be able to maintain an index to all documents available on the Web. Further, with such a large index it may be difficult for users to formulate queries narrow enough to return a suitable set of references [Sheldon94].

Global indices can generally be built automatically by web roaming applications and therefore require few human resources.

Indices to everything may help readers answer all questions. However, in order to answer question 2, readers would need to query the index manually. Answering question 3 might be impractical.

Requirement 1 is met if the file is included in the index, requirement 2 is partially met but readers are required to perform the search manually. Requirement 3 is not met.

4.2.2 Indices to services

Indices to services and further sources of information (such as ALIWEB) can provide useful starting points for browsing or further searches (by providing an index to indices). However, once a source of further information has been identified, the user must again interrogate the system (by browsing or using a search engine) in an attempt to locate the documents which hold the information they are actually interested in.

Such indices tend to require human intervention to build an index to the services offered on individual machines. This may be inappropriate if the file needs to be regularly modified. Automatic generation of index files based on *transducers*[Gifford91], eg. the `site-index` tools, are currently under development.

All questions may be answered, depending on whether particular documents or topics have been submitted for inclusion in the index. Requirement 1 is not met (by design), requirement 2 is partially met but readers are required to perform the search manually. Requirement 3 is not met.

5 Caching

Caching servers (such as that at the Unix Hensa archive) act as intermediaries between readers and the rest of the world. As a document is requested, the server checks for a cached copy. If a copy exists, it is sent to the reader, otherwise it is retrieved from the document’s original server. Having retrieved and stored a copy of the document, it is forwarded to the reader. As there is now a copy of the document in the cache, the next request for it will return the cached version which (should) be returned faster than if the same request was made to the original site.

This mechanism can reduce the load on networks and servers when several readers frequently request the same document from remote sites, especially when using web roaming applications.

Caches can only help if they retain a high proportion of the documents that are being requested, since the caching process carries an overhead that would otherwise increase the time to load individual documents. As the number of documents in the Web grows, caches will have to grow or will retain fewer of the documents.

While undoubtedly useful, caches work against the principle of the Web - to provide access to *distributed* information

- since they attempt to store a copy of the world in a single location.

6 Proposal

Current indices require readers to have some knowledge of their location. There is no consistent way of jumping to these services from the current variety of Web clients. Users should not need to know the locations of indices.

The future implementation of Uniform Resource Names (URN), is intended to provide location independent, persistent, pointers to objects. URN to URL resolutions services (Registrars) must be provided in order for client applications to retrieve copies of the objects. To be functional these services would need to store details on each registered document from a variety of sites. For example, one Registrar might provide resolution for the documents from computing laboratories of all Universities in the United Kingdom. Users should not need to know the location of such services.

These Registrars are the ideal location to also provide query services:

- To be functional, both need high proportions of documents to be registered;
- Both need to be provided anonymously (as far as the user is concerned);
- Both need to store at least a limited amount of information on each document to enable their identification to the user. A duplication of some of this information would occur if they were separate services.

6.1 Indexing individual documents

If the URN Registrars are to be able to offer indices to registered documents, keywords and other related information needs to be collected at the time of registration. There are two functional options for the submission of such material, which is obtained by using *transducers*[Gifford91] (filters). In both cases documents are initially POSTed to a local server:

1. the local server sends the whole document to the Registrar. This would allow 'global' filters to be used;
2. the local server filters the document and then sends the keywords (and other information) to the Registrar.

This area needs to be discussed in order to determine what information should be indexed. Do we want to allow readers to search for any word in the document (requiring huge indices), or more limited terms from titles, headers, and meta-information?

6.2 Indexing huge numbers of documents

Earlier, I stated that indices to everything are neither functional nor scalable. We must provide some way of linking many small indices into a huge virtual index. Sheldon et al. [Sheldon94] suggest a method of connecting index servers and *content routers* in order to provide a scalable means of accessing large amounts of information

6.2.1 Distributed, specialised, connected indices

In their work, Sheldon et al. propose that files are grouped into *collections*, each having a *content label* which provides an abstract of that collection. This is comparable to ALIWEB entries. Users formulate queries by supplying attribute (field)-value pairs corresponding to the fields in the content labels.

In a prototype content routing system, queries may be formulated by browsing through a virtual directory structure of the fields and their relevant values contained within the collection, or by specifying the field-value pairs directly. This *high level* query is intended to constrain the search to a limited number of collections so that only those index servers and content routers containing potentially useful information are involved in subsequent, detailed, queries.

When a suitably constrained query has been defined it is forwarded to the relevant index servers and content routers. The user may then expand the query to include further fields that are available within the defined collections.

With such an interface, it is possible that index servers could be maintained to manage the documents for quite distinct domains (such as that mentioned earlier).

The browsing scheme of selecting fields and values forces users to choose keywords which are known to exist in the index, rather than generating a set of keywords that they *hope* has been used. Suitable field names that could be used by all transducers in order to create a consistent virtual index structure need to be defined.

The virtual directory structure used to present fields and their values could be modelled in current hypertext documents by returning either a directory structure (similar to the ftp interface through the Web) or a dynamic hypertext document with links to appropriate fields and values. The final level of the query would present a list of documents that match the query.

6.3 Associative access

Detailed content labels give an abstract view to documents they describe. If Web clients were to retrieve and store the content label of a displayed document, by issuing queries based on sub-sets of the label, related documents might also be located and could subsequently be retrieved. While some of these other documents may have been linked explicitly, there is a possibility of locating implicitly linked information as well.

A standalone hypertext system, StrathTutor [Kibby89][Mayes88], uses a similar method of access to provide a dynamic hypertext. Links are generated, dynamically, on the basis of the similarity between attributes required by a link button, and the attributes the author specifies as being dealt with by an individual node. In such a system it is not necessary to explicitly define links between documents.

6.4 How does this help?

We have a consistent index space to a large proportion of the documents, enabling solutions to all questions and requirement 1 to be met.

The system is scalable since each index server is concerned with a particular domain. These domains could be split and served by a content router at a later date.

The use of content labels to locate implicit links provides a solution to question 2 and meets requirement 2.

The above proposal fails to meet requirement 3. I believe that this would require applications such as "user agents" which would scan records being added to the index. Users would be informed of new information which they may be interested in. (See CERN discussion on notification of new material).

7 Conclusions

I have discussed the need for indices based on the need to locate documents of interest without needing to browse through large amounts of irrelevant material.

Current resource discovery tools work, but are only functional while the Web is relatively small. We must strive to provide better tools in the near future in order to generate continued interest in the use of the Web.

I have proposed a functional, scalable solution to the problem of indexing individual documents which provides a browsable interface. The model provides a means of gaining associative access to documents which may not be explicitly linked.

The need for further work to determine how to automate the generation of suitable keywords is identified. Further, a need to provide additional resource discovery tools for dynamic hypertexts is also identified.

8 References

8.1 Paper based references

[Gifford91] Gifford, DK., Jouvelot, P., Sheldon, MA., O'Toole, JW. Jr. (1991) Semantic File Systems. Operating Systems Review, **25**(5), 1991.

[Hammond91] Hammond, N. (1991) Teaching with Hypermedia: Problems and Prospects. In *Hypermedia/Hypertext and object-oriented database*, ed. Brown, H. London: Chapman & Hall.

[Kibby89] Kibby, MR., & Mayes, JT. (1989) Towards Intelligent Hypertext. In *Hypertext Theory into Practice*, ed. McAleese, R. Ablex.

[Mayes88] Mayes, JT., Kibby, MR., & Watson, H. (1988) StrathTutor©: The Development and Evaluation of a Learning-by-Browsing System on the Macintosh. In *Computers in Education*, **12**(1), pp.221-229.

[Sheldon94] Sheldon, MA., Duda, A., Weiss, R., O'Toole, JW. Jr., Gifford, DK. (1994) Content Routing for Distributed Information Servers, Extending Database Technology, LNCS, March, 1994, Cambridge, England.

8.2 On-line references

The following on-line references have been used throughout the paper.

- Burners-Lee, T. World Wide Web Initiative: The Project — `http://info.cern.ch/hypertext/WWW/TheProject`
- What Is HyperText? — `http://info.cern.ch/hypertext/WWW/WhatIs.html`
- Burners-Lee, T., & Cailliau, R. WorldWideWeb: Proposal for a HyperText Project — `http://info.cern.ch/hypertext/WWW/Proposal.html`
- Post, R. Lagoon: a WWW cache — `http://www.win.tue.nl/lagoon/`
- Fletcher, J. 1993. Jumpstation — `http://www.stir.ac.uk/jsbin/js`
- CUI W3 Catalog — `http://cui_www.unige.ch/w3catalog`
- Koster, M. ALIWEB (Archie Like Indexing the WEB) — `http://web.nexor.co.uk/aliweb/doc/aliweb.html`
- Koster, M. Simple Unified Search Interface (SUSI) — `http://web.nexor.co.uk/susi/susi.html`
- CUI Meta-Index — `http://cui_www.unige.ch/meta-index.html`
- World Wide Web Worm — `http://www.cs.colorado.edu/home/mcbryan/WWWW.html`
- Thau, R. Site Index Transducer — `http://www.ai.mit.edu/tools/site-index.html`
- Unix Hensa Cache — `http://www.hensa.ac.uk/hensa.unix.html`
- Burners-Lee, T. Uniform Resource Names, in *UR* and The Names and Addresses of WWW objects* — `http://info.cern.ch/hypertext/WWW/Addressing/Addressing.html`
- Notification of new material — `http://info.cern.ch/hypertext/WWW/DesignIssues/Notification.html`
- De Bra, P., Houben, G-J., & Kornatzky, Y. Navigational Search in the World-Wide Web — `http://www.win.tue.nl/help/doc/demo.ps`