Unifying Browse and Search in Information Hierarchies

Ullas Gargi
Imaging Systems Laboratory
HP Laboratories Palo Alto
HPL-2003-180
August 25th , 2003*

information retrieval, MEERCAT, browsing, search, asset management

Information retrieval systems offer two main methods to access information: searching by submitting a query and browsing a constructed navigation hierarchy. In most systems, these two methods are accessed and operated by the information seeker in different ways and generate different types of result sets that allow different kinds of operations to be performed upon them.

This paper suggests two general principles and a set of techniques implementing them for combining the browsing and searching access methods and interfaces for a system into a unified information navigation access method. It illustrates some of these techniques with implementations in the MEERCAT multimedia information management system.

* Internal Accession Date Only
Approved for External Publication

Unifying Browse and Search in Information Hierarchies

Ullas Gargi
HP Labs

August 2003

1 Introduction

The American Heritage Dictionary of the English Language, Fourth Edition offers the following definitions for "browse": to inspect something leisurely and casually; to read something superficially by selecting passages at random; to look through or over (something) casually.

Roget's Interactive Thesaurus, First Edition (v 1.0.0) lists the following synonyms for the "browse" entry: check over, dip into, examine, feed, flip through, glance at, graze, inspect loosely, leaf through, nibble, once over, peruse, read, riff through, riffle through, run through, scan, skim, skip through, survey, thumb through, wander.

Browsing and searching are the two dominant modes for exploration offered by information management systems, corresponding to the two dominant information seeking behaviors of people. Browsing is commonly offered through a hierarchy imposed on the document space that the user navigates by following parent and child links.

Cove and Walsh [CW88] have suggested that there are actually three information exploration behaviors:

- Search browsing; directed search; where the goal is known
- General purpose browsing; consulting sources that have a high likelihood of items of interest
- Serendipitous browsing; purely random

From here on, we will use the word search to mean directed search; the qualification is used to differentiate it from generic information search which we refer to as information exploration. We do not consider serendipitous browsing. Our goal is to design a system that offers tightly–coupled directed search and general purpose browsing.

We assume that some sort of hierarchical organization of documents in the system, not necessarily evenly grained or complete, forms a browsing structure that is accessible to the user. Also we assume that there exists metadata for documents other than the hierarchical organization.

1.1 Definitions

Document A complete, cohesive intelligible data object that may be read or consumed by a person without necessary recourse to other objects. It may contain any combination of text, 2-D image, multidimensional image, audio, video, path trajectory, or other data type. Sometimes referred to as a multimedia document or an information asset.

Target document(s) The document(s) that the user wants or that fulfill the user's information needs.

Result set The set of documents returned by the system in response to a query or to a browse request.

Discriminator A common attribute of a set of documents that may be used to classify or divide them.

Browse Node A node in an information hierarchy that may contain other nodes or final documents. Also referred to as a category.

1.2 Tightly–coupled searching and browsing

Most information system separate searching and browsing unnecessarily widely. For example, there is often a separate search page that takes one back to the root node even if one has already spent time browsing to find the approximate target document location. Or else the manner in which a result set is presented or the things that can be done with one are different from what the user may do with static browse nodes. Since most information explorers use a combination of search and browse steps in non-strictly alternating sequence [CP95, DSG03], the system should allow whichever is the current information seeking behavior of the user to be performed, regardless of system state. Ideally it should also allow the same set of operations to be performed on a query result set that can be performed on a browse node.

Thus, one can state the two fundamental design principles for unifying browse and search. To tightly couple these two information exploration strategies a user must

1. Be able to perform either a hierarchy navigation (browse) or a query submission (search) operation at any step of an information exploration session without losing the current state and be presented with a new view that is consistent with the current state.
2. Be able to perform the same set of operations on the current result set regardless of whether that result set was obtained through a query submission or a hierarchy navigation step.

Whatever features are offered by the system for either browsing or searching must be accessible during any step in a sequence of search and browse steps, regardless of that sequence. This is what we mean by tightly–coupled searching and browsing. The example offered earlier of having the search function only available from the root node of the browsing hierarchy violates the first principle. As another example, the MEERCAT (Multimedia Enhanced Exploration, Retrieval and Categorization) system allows nodes corresponding to asset collections under a browse node to be downloaded to the client as Zip files. To satisfy the second design principle, the same functionality has to be allowed on result sets obtained from queries, possibly with a limit on the top N matching items.

In the following sections we present some techniques to achieve a tightly–coupled unified searching and browsing framework.

2 Search after Browse: Hierarchy–constrained searching

Very often the user is able to narrow the desired set of documents very well using the static navigation hierarchy provided by the designer. In this case, they navigate down to the node where the system organization does not match their desired discriminator characteristic and then use search. Most good information hierarchies, such as Yahoo (http://www.yahoo.com) or the Open Directory project (http://dmoz.org), allow this. Figure 1 illustrates the process. The user implicitly selects the subset of the document space spanned by the subtree under the current browse node. This document space subset is then filtered through the query to present the result set. Ideally the system should make use of the users navigation to optimize the query execution process itself, not just to remove documents from the result set that are not in the browsing subtree.

Figure 2 shows the results of a global and hierarchy–constrained search in the MEERCAT system. The query "meta=crowd" (searching for images containing crowds of human faces) yields a 38–item result set with target documents from stock photo collections when executed on the entire document space (first screenshot), and a smaller 27–item result set when executed on the "User Collections" subtree.

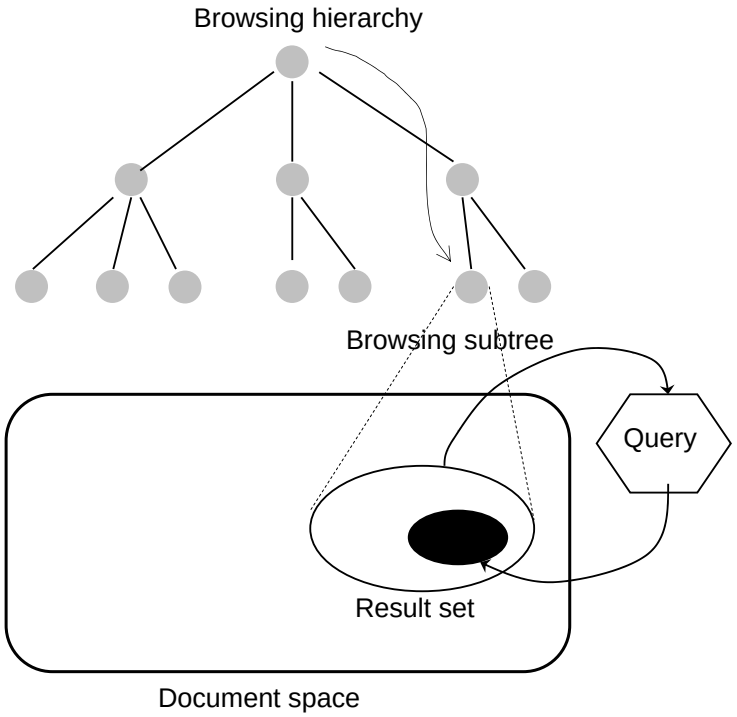


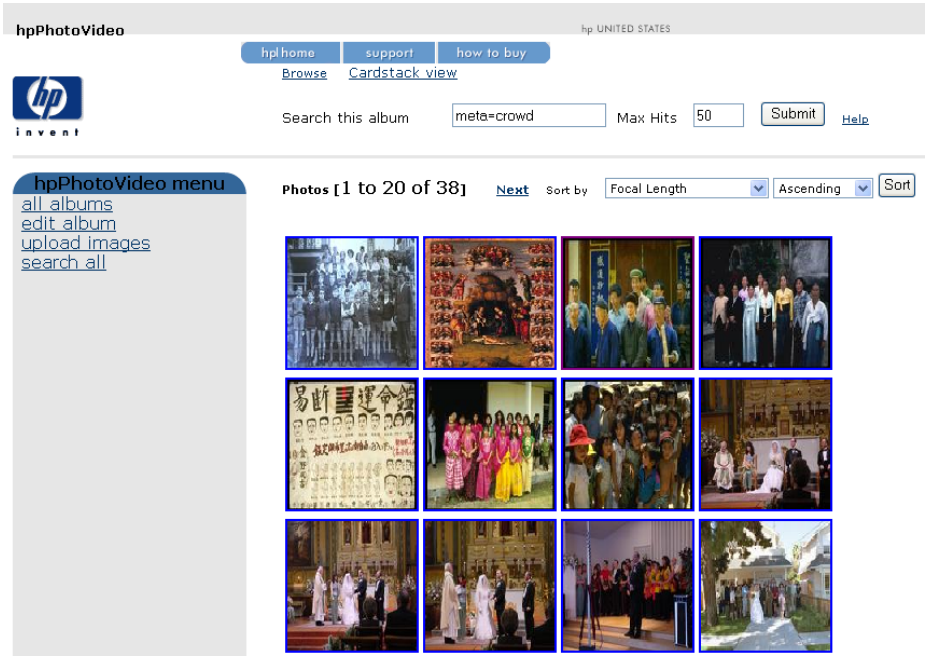
Figure 1: Searching after browsing: hierarchy–constrained search.

3 Browse after search: Dynamic browse–trees

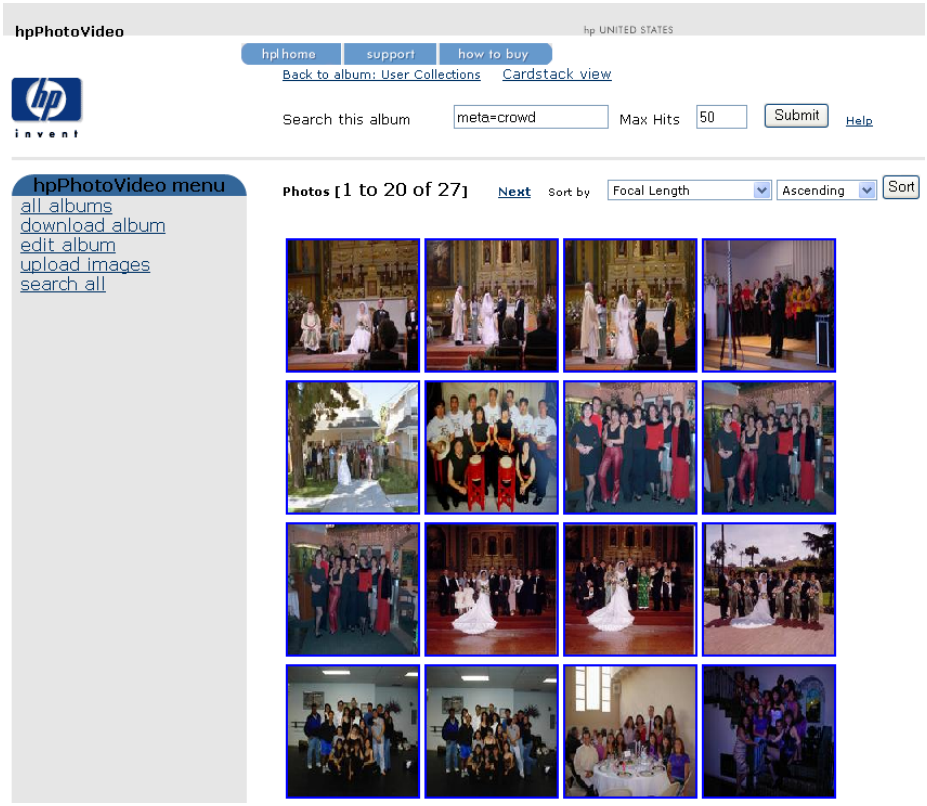
A result set is essentially a re-ordering of the entire document space with a dynamically specified ordering criterion (similarity to the query). If the document space has structure to begin with, the result set need not be a serial ordering of matching documents with possible relevance rankings and summaries. Instead an organization can be imposed on it. This organization can be achieved in the following ways.

3.1 Separating document from category matches

Yahoo and other information directories commonly return category matches in addition to individual item matches. These are categories which directly match the query terms, not just categories which contain some matching documents. Figure 3 shows category matches in MEERCAT. Search result sets are instantiated as (special) browsing nodes and matching categories are represented as links. The query "any=food" results in two category matches as well as individual image matches.



(a)



(b)

Figure 2: Screenshots of (a) non-constrained global and (b) hierarchy-constrained search for images with crowds in the MEERCAT system: (a) returns 38 matches in the entire collection while (b) returns 27 matches under the “User Collections” browsing node.

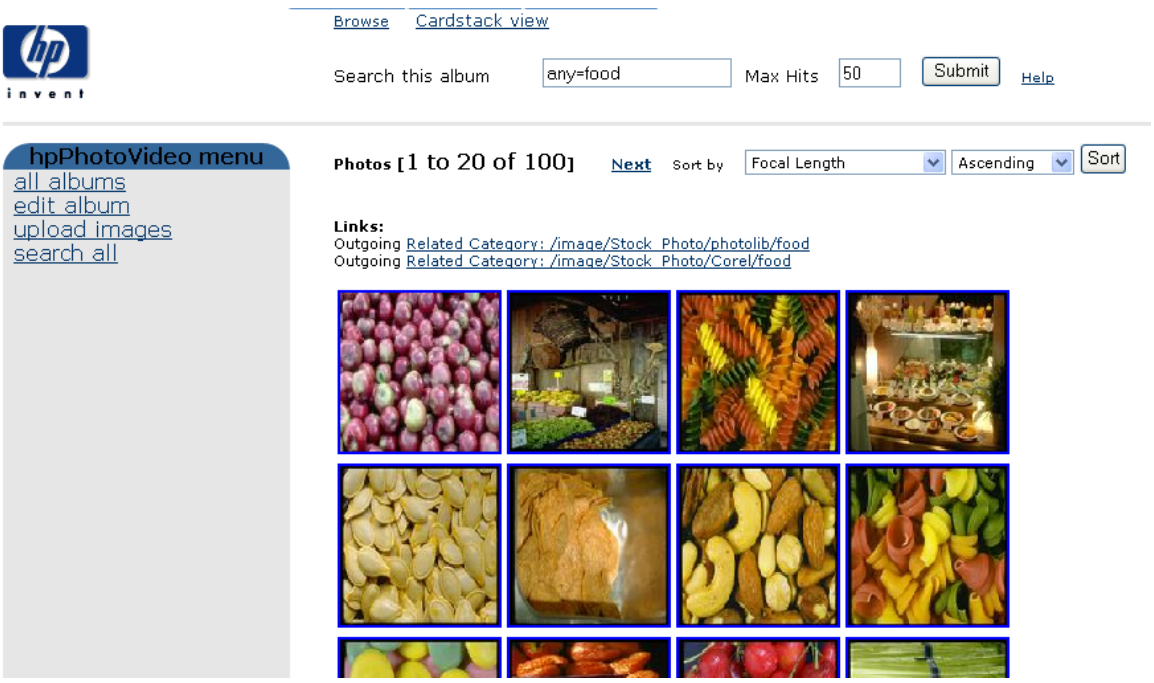


Figure 3: Screenshots of search result set in the MEERCAT system showing separate category and document matches.

3.2 Dynamic clustering of the result set

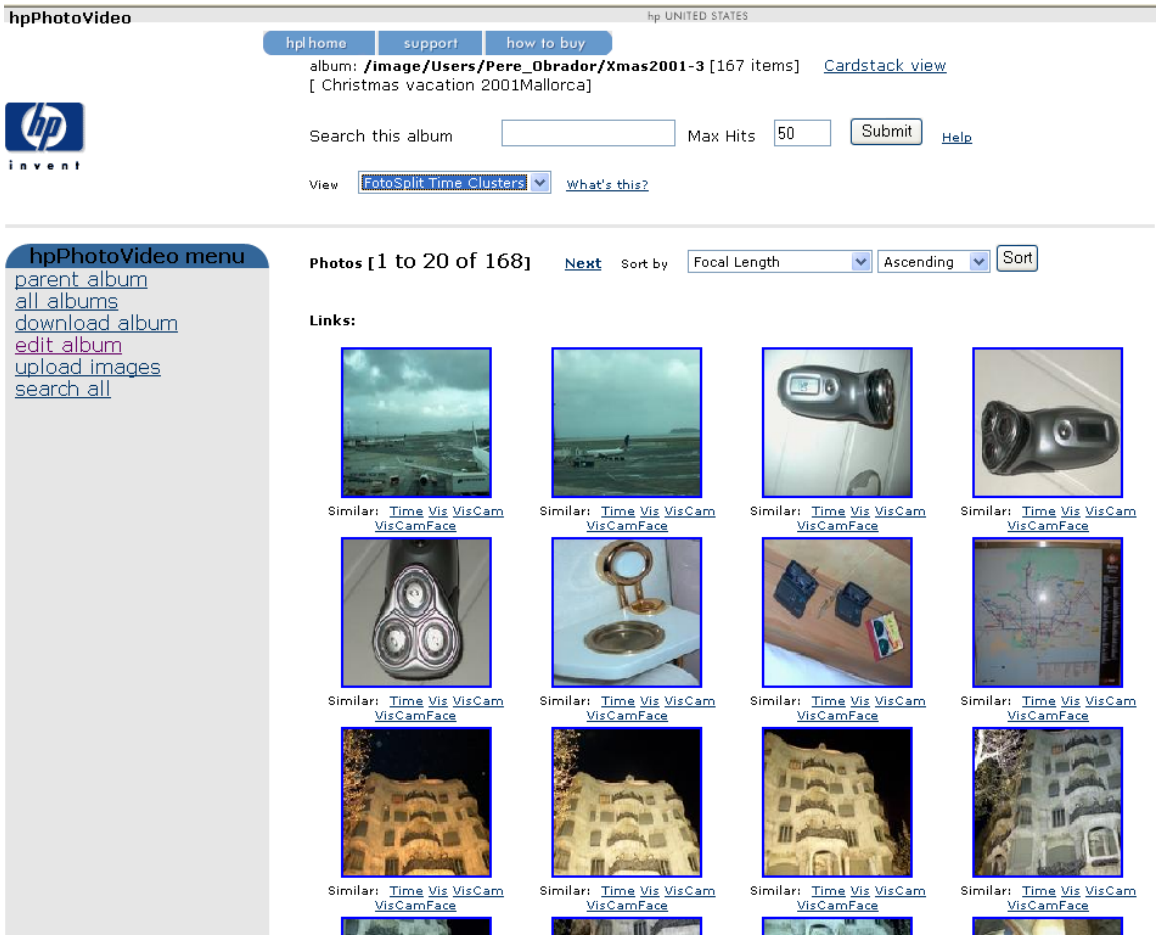
Other researchers have demonstrated the utility of clustering search results in real-time [CKPT92, CGNS99]. The Northern Light (http://www.northernlight.com) search engine's distinguishing feature during its heyday was sorting query results into "Custom Search Folders (TM)" presented in a left panel. Four types of attributes were used for clustering, either subject topic, document type (e.g. recipes, resumes, press release), source (e.g. web sites, magazines) or language (e.g. English, French) of document. Choosing a folder caused its contents to be recursively clustered.

The MEERCAT system allows a photograph album's contents to be dynamically clustered into time clusters using the timestamp metadata as described in an HPL Technical Report [Gar03]. Figure 4 shows the contents of an album with the dynamically generated time clusters.

3.3 Dynamic browse–trees

Another alternative is to use the existing hierarchy (possibly in addition to, or in place of the results of dynamic clustering) to organize the search results. After all, the static hierarchy was presumably created with some care, probably by hand, and hence is likely to be of high quality and valuable in organizing the result set. Note that this is different from merely listing matching categories.

The process is illustrated in Figure 5. The document space is searched with the users query and the set of documents that do not match the user's query is (conceptually) removed from the sub-tree under the node being searched (in this case the root node). If each document has metadata about categories it is a member of, then a dynamic browse-tree may be generated by retrieving this value for each member of a search result set and generating a minimum spanning tree from the resulting set of browse nodes. Empty nodes may be pruned, skinny paths (a skinny path is a chain of



(a)

(b)

Figure 4: Screenshots of (a) default organization and (b) dynamic time cluster organization of a user photo album within the MEERCAT system: (a) is just a paginated listing of images while (b) has organized the set into clusters.

browse nodes with only one child each) collapsed and the resulting hierarchy presented to the user to browse. The item relevance is retained by assigning a weight to each remaining path to leaf nodes. The relevance for a node is the maximum relevance of any node or item under it (alternatives such as average are possible).

This technique is more applicable when query terms are hard, allowing documents to be definitively removed from consideration. It can be thought of as a document-tree-level version of query-sensitive summaries (where the query terms are shown in the context of a hit document).

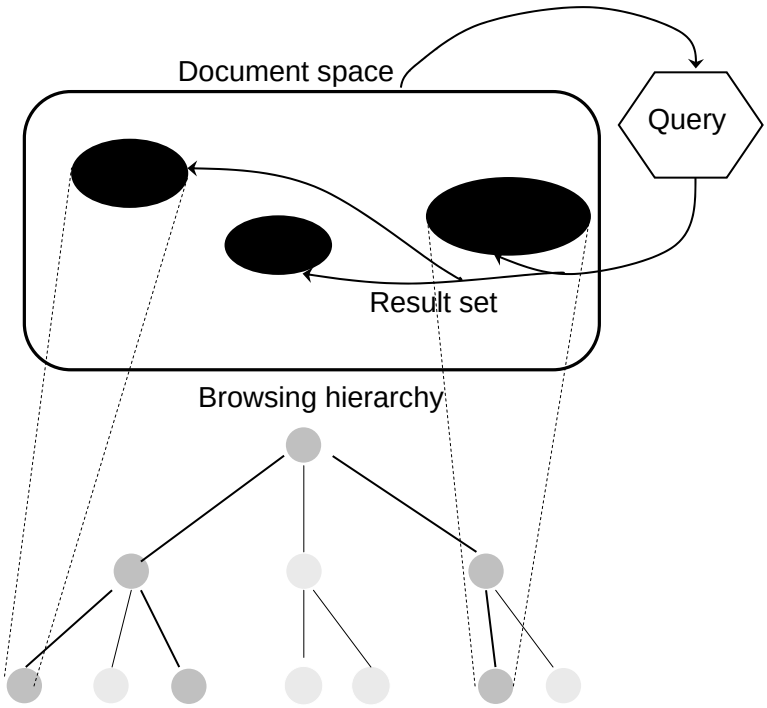


Figure 5: Browsing search results using dynamic browse–trees.

4 Search by Hierarchy Location

Hierarchy distance as a distance measure

The hierarchy organizing an information collection is global metadata that can be used to search. Given an existing hierarchy, a document's position in this hierarchy relative to another constitutes the hierarchy designer's measure of similarity. One should therefore be able to find browse nodes that are similar to the current browse node in terms of what we call hierarchy distance. The distance between two browse nodes $B1$ and $B2$ (and implicitly their constituent documents) may be measured by

- Minimum number of steps to reach $B2$ from $B1$ moving along the tree.
- Difference in hierarchy level (steps from root or steps above closest leaf) between $B2$ from $B1$

Hierarchy distance can be combined with other techniques. For example, searching for nodes that are content-similar to $B1$ based on its metadata and that are also hierarchy-similar to it will likely yield nodes at the same level of information granularity.

One can think of this as creating a site map for a website customized from the point of view of the current browsing node or document.

Lateral Browsing

There are sometimes relationships between documents or document groups (nodes) other than can be well represented in a hierarchy, even using multiple virtual categories (Section 7). This is often implemented using a "also see" link. This allows a shortcut between distant regions of the information hierarchy. These kinds of link metadata should also be searchable and browsable.

5 Browse with Persistent Search: Filtered browsing

In some situations the user does not have a clear query term in their mind but instead a set of terms which might apply. They may not also know which top-level category their target documents might lie in. In addition, the query terms may be common enough that the result set from searching at the root node is very large (this is almost always true for a web search).

In a filtered browsing session, the system presents a three-panel interface with separate areas for entering search terms, for the browsing hierarchy and for the result set. The user uses the browse panel to navigate as usual. At any time they may enter or alter their current search terms. The result set panel presents documents that match both the search and the browse constraints. Any subsequent navigation presents only documents matching these terms. The filter terms can be dynamically modified.

This is similar to the dynamic browse–trees method but is more useful in cases where the user is unsure of the terms to use and makes frequent query modifications based on the result set they see. The navigation hierarchy is not pruned but left as-is so the user can see the (lack of) results of the current query. The document result set is also immediately viewable to give the user feedback rather than having to navigate to a leaf browsing node. Every navigation step in the displayed browsing hierarchy is implicitly a hierarchy–constrained search on the selected browsing node (as described in Section 2). Thus, the system presents categories from the browsing hierarchy but documents from the continuous query result set. As the user navigates the browsing hierarchy, the search result set is altered to only contain target documents under the current browsing node. In this way, the user is guided down the hierarchy with the current search result set acting as both an information scent [PC99] marker and the end result. This kind of information exploration is very much the kind of tightly–coupled search/browse that we advocate.

Figures 6 through 8 shows three screenshots of the MEERCAT implementation of filtered browsing. In the first, the user has entered a query ("any=building" meaning find anything related to the term "building") at the root node of the static hierarchy. The navigation hierarchy labeled "child albums" is shown in the middle panel and the search result set is shown in the bottom panel. Each child node link actually points to a query URL that is executed when the child node is clicked on. The second screenshot shows the result of the user clicking on the "Art Images" browsing node. The hierarchy is expanded downward but the search result set is empty because no target documents for the current query exist under this browsing node. The third screenshot shows the state after the user has backed up to the root and then selected the "Misc HP Images" child node. The result set here is non-empty.

Note that if the user modifies the query, it is executed only for the current browsing node. If the user then navigates up toward a parent or higher node, the query is automatically executed on the larger domain.

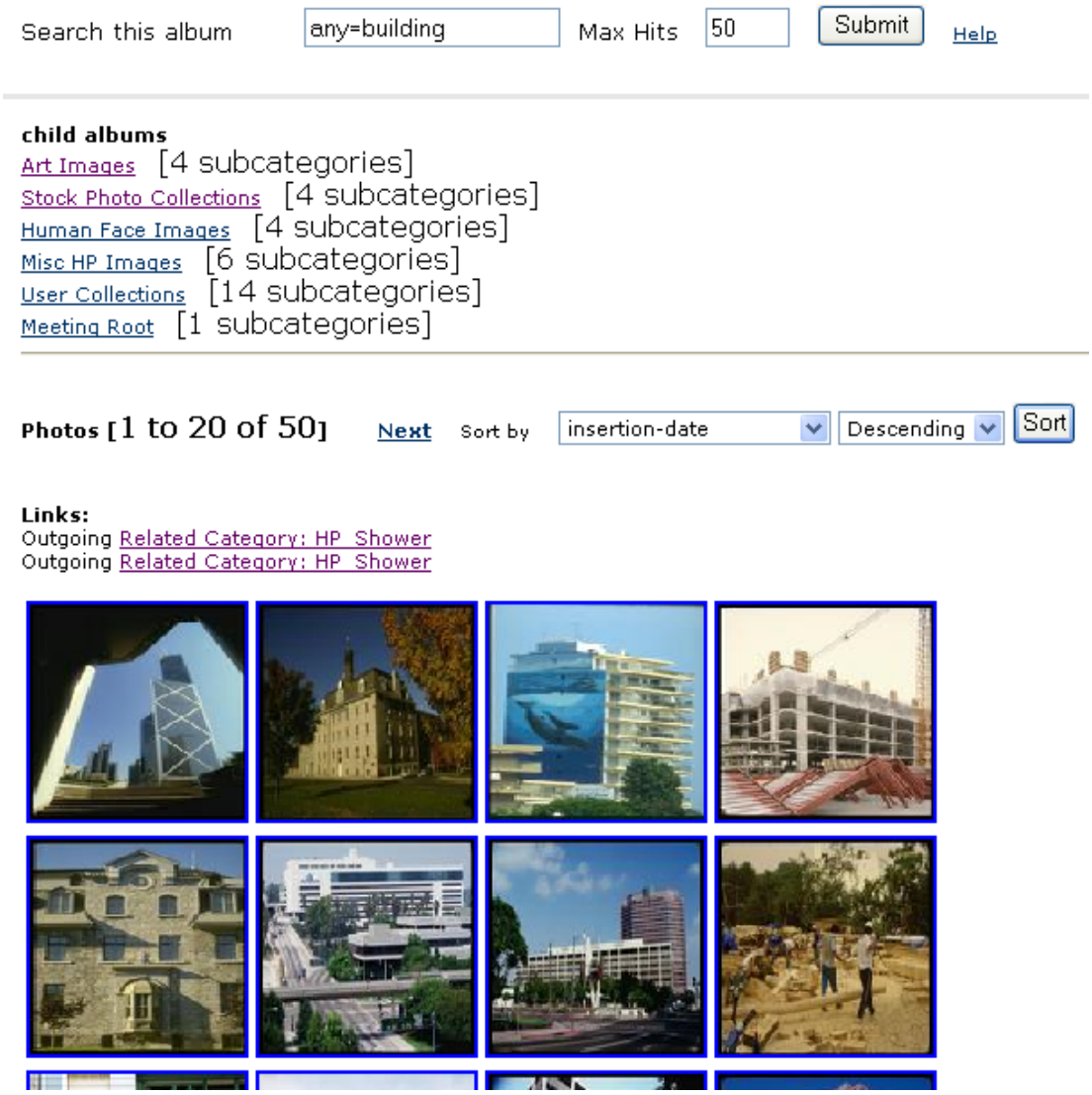


Figure 6: Screenshot of filtered browsing in the MEERCAT system. At a top-level node, the user has entered a query. The system presents the navigation hierarchy as well as the search result set.

6 Browsing Search Result Sets

This section discusses techniques to allow the user to perform browse-type operations on an evolving search result set.

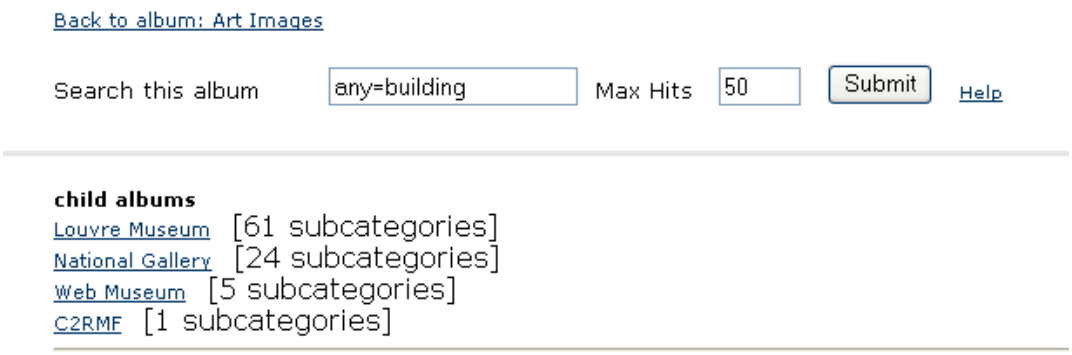


Figure 7: Screenshot of filtered browsing in the MEERCAT system: the user has navigated the hierarchy with an active query. The resulting filtered browsing node yields an empty result set.

6.1 Merge Search

It is a well known defect of a static hierarchy that the need to hard-cut the set of documents at each node can result in two very closely related documents lying along very different paths from the root.

In addition, documents that are closely related in some way unknown or unaccounted for by the system designer, but present in the users mind, will end up in very distant nodes. The ability to perform hierarchy constrained search (Section 2) only under a single node is compromised by these facts. Ideally the user should be able to create a result set node where the items from these distant regions of the hierarchy are accumulated for efficient browsing, download or other operations.

To achieve this, the system allows a feature we call *Merge Search*. After the user has performed a search under a node, they may click the Merge button which indicates to the system that the user wishes to keep the current result set active for further accumulation. Subsequent searches (NOT browses) add to the current result any new resulting documents. As a hypothetical example, the user may search under Art, Commercial, Advertising, Posters, Retail. They may then turn Merge Search on, navigate to Computers,Printing,Services,Posters and perform a search there. The result set presented is the combined result of both searches. The merged result set may be sorted by absolute relevancy to the appropriate query or dynamic browse–trees can be used to keep the two result sets distinct.

Merge search fulfills one requirement of Bates' "berry-picking" model [Bat89] by allowing partial results to be accumulated.

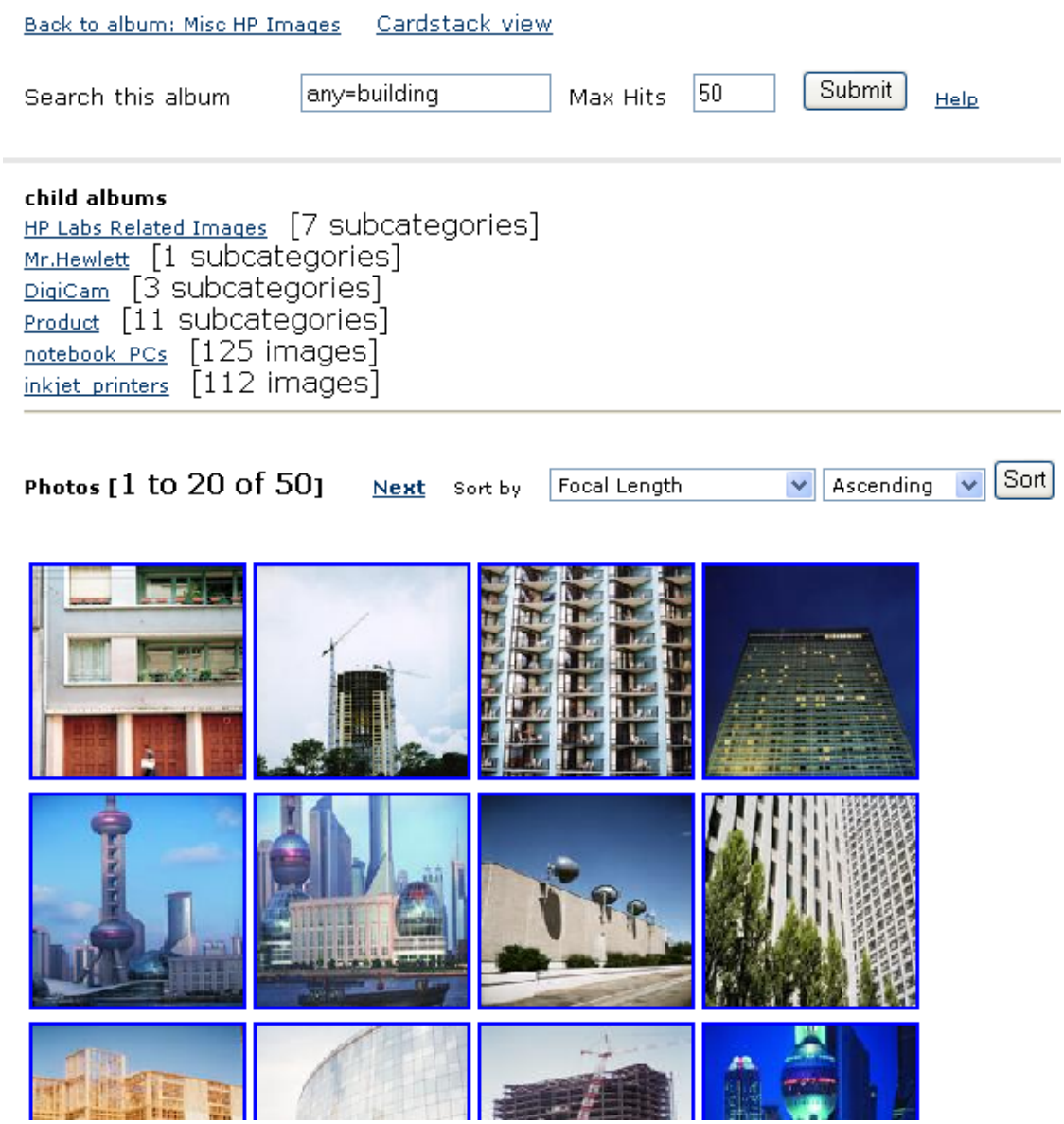


Figure 8: Screenshot of filtered browsing in the MEERCAT system: the user has re-navigated the hierarchy with an active query. This filtered browsing node has a non-empty result set.

6.2 Making search result sets equal citizens

Sorting Result Sets

The same sorting attributes are available for sorting result sets as for browse node document lists. For example, in the MEERCAT system, browse nodes of image collections may be sorted by any EXIF digital image header attributed. Search result sets may also be sorted by these attributes.

Bookmarking search result sets

It is commonplace for users of search engines to use a query result from the engine as a static bookmark. One might say "search for term X and it's somewhere on the first page" for example, and expect the listener to be able to find the referred to documents, as if the search engine provided a static information collection.

This propensity of users to use search queries as shorthand for a path to a set of documents can be made explicit by making each query a permanent node in the system. The user can then actually bookmark the node by adding it to a lightbox for example. In the MEERCAT systems, query results are retained as browsing nodes with unique IDs and can hence be referred to by the user. The user's history of past information exploration sessions can be made explicitly available to them. Such meta-information exploration can be self-revealing to users. The Google toolbar available from http://www.google.com offers a primitive version of this feature by making past queries viewable via a drop down menu.

Downloading search result sets

Since search result sets are considered the equal of browse result sets (browse nodes), all operations possible on the latter should be possible on the former. This includes bookmarking as mentioned above. It also includes downloading search result sets. The documents should retain their browsing metadata so ideally a dynamic browse-tree should be generated. In addition, any query-sensitive summaries used in the search result-set should be retained so that offline local browsing of the downloaded data is exactly equivalent to online browsing of the search result set node.

Searching within search result sets

Many systems allow searches to be refined by searching within results. This is natural in a tightly-coupled browsing and searching environment where searching within search results is just like a hierarchy-constrained search at a browsing node.

6.3 Browsing search result sets across sessions

Another enhancement is to allow users to browse search sessions. Conceptually, all search result sets behave like browse nodes in the tree of queries. Clustering result sets may be achieved by clustering the queries that generated them (queries are clustered anonymously across users and sessions). At query submission time, a query is matched to the nearest cluster. Therefore every search result page has a "member of categories" or "also see" link field that can be used for lateral browsing to the cluster of queries or query sessions that contains it. This can help a user who does not know what query terms are available to use.

7 Flexible Organization with Multiple Category Views

There is no reason to impose a single-inheritance hierarchy on an information collection [Fil03]. It is possible to allow more than one hierarchical organization of the same collection. We refer to this as flexible organization. A non-IR

example of where this would be useful is in capturing the multiple ways that people in a business organization actually relate. There is the official organization chart and there are multiple overlapping organization sub-charts representing people's contacts, collaborations, past history, unofficial relationships, etc. Our concept of flexible organization is not that a completely new hierarchy is built. Instead, the set of links that a node in a graph has are clustered differently. Each such clustering is referred to as a view or virtual category. MEERCAT implements flexible organization through stateful browsing of multi-viewed hierarchies. Multiple virtual categories can be defined during the data ingestion phase by specifying an XML file detailing additional views for any nodes that need them, in addition to the default filesystem directory hierarchy.

Any time a user navigates the hierarchy, at any node where there is more than clustering of the child nodes, they get to pick the view or one is chosen as the default. Further navigation down that sub-tree has the view at that node saved in the session information. There may be a cascade of views, one for each node navigated. When a user backs up from a node to its parent, the view history is unrolled and the same clustering is presented.

The dynamic clustering of images based on capture time metadata possible in MEERCAT shown in Figure 4 is also presented as a virtual category view. As another example, MEERCAT was used to create an index to a commercial image library (the hpimages.com collection). During the ingestion phase, product images were classified not just into type categories (the available organization) but also into market segment categories.

Browsing of search result sets (Section 6) can be combined with virtual category views to create an even more powerful mechanism for exploring a hierarchy. In such a scenario the user picks a view, gathers target documents, switches to a different organization of the same document sub-tree and appends more target documents to their result set.

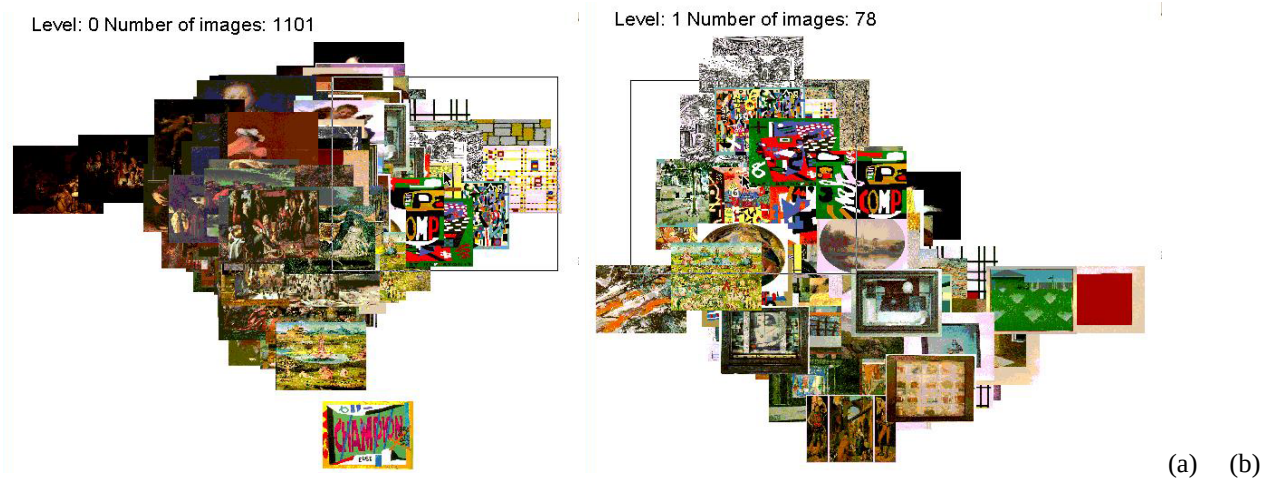
Automatic discriminator selection

At any node in the tree, there can be a number of attributes that may be used as discriminators for a category view. Picking one attribute results in one set of clusters that is orthogonal to the other possible groupings. Traditionally, only one grouping is possible in information hierarchies, usually fixed by the system designer. Yee et al [YSLH03] use multi-faceted metadata to refer to a node having a set of orthogonal attributes (facets), each of which can be used as a discriminator. In their user interface, the different possible independent groupings at a node or search result are shown in a left panel in addition to individual result documents. Our system achieves something in between these two by displaying only one grouping at a time, but allowing the user to pick from a list of discriminators to use dynamically.

A hand-crafted static hierarchy can be very useful as a default. But ideally the user should be able to navigate the information using any combination of attributes specified at runtime. This is usually done by providing a search feature. However, the results of the search are again usually a flat listing, throwing away the benefits of a hierarchy. The methods proposed earlier improve upon this, but they are still constrained by the original browsing hierarchy. One way to remove this limitation is to let the system automatically choose the dimension (feature, attribute) to use to split up (organize) the members of a node. For example, the most discriminating feature can be determined by running any of several standard pattern recognition methods (e.g. Fisher discriminant analysis, PCA analysis).

One can generalize this to more than one dimension by generating a two- or three-dimensional visualization where each axis of the display represents either a fixed attribute or some discriminatory subspace axis. Figure 9 shows an example of such a scheme. Here low-level image features such as color and texture are extracted offline for each image. When browsing a particular sub-collection of images, the high-dimensional features of that subset of images are reduced to two dimensions using an approximate algorithm that maintains dissimilarities [FL95]. The images are then laid out to correspond to these dimensions, resulting in similar images lying nearby and dissimilar images being further apart. Thus, the dimensions (attributes, features) to use in order to best separate the current result set are dynamically and automatically determined. The first screenshot shows the initial result set of 1101 images and the user's selection of a subset of 78 images via a rectangular region selection tool (this implementation used Java for the interface and C for the dimensionality reduction and was viewed remotely using VNC [RSFWH98]). The second

screenshot shows the subset dynamically clustered and laid out.



(a) (b)

Figure 9: Screenshots of (a) starting result set and (b) dynamic clustering in two dimensions using automatic discriminator selection.

8 Related Work

Cox [Cox92] proposed a system for information access by browsing alone. According to him, the query model has dominated information retrieval because of the belief that browsing requires examination of too many documents, but searching has its own problems, namely:

- User difficulties in query formulation
- Null returns and too many matches
- Variation in importance of query components and term dependencies
- Adaptation to different forms of objects such as pictures and sounds
- Finding good surrogate representations for objects.

Browsing, on the other hand, has the advantages of using humans spatial ability to navigate, In his proposal, searching is entirely eliminated and a system based on Bates "berry picking" model [Bat89] (very briefly: the user's information needs (nature of the query) evolve during a session and their need is not satisfied by a final result set but instead by information berries they pick over the whole session) is presented. A rich set of interconnected nearest-neighbor networks provides a browsing structure. Both objects and sets of objects can have multiple similar-neighbor networks. There are multiple parallel networks consisting of nearest neighbor links between (sets of) objects and there are interconnections between the networks themselves.

We may mention here that some of the problems with searching that Cox summarizes above may be ameliorated by the use of the tightly–coupled browse/search techniques presented in this paper. For example, "null returns and too many matches" can be solved by using dynamic browse–trees (Section 3) and hierarchy–constrained searching (Section 2) respectively. "Variation in importance of query components and term dependencies" will be helped by using filtered browsing (Section 5), possibly in conjunction with multiple category views (Section 7).

The etymological history of the word "browse" is brought to the fore again in Pirolli and Card's work on Information Foraging theory [PC99]. In that work, they cite Dennett's characterization of humans as informavores and suggest that people adapt their strategies and the information environment to maximize their rate of gaining valuable information. Their analysis develops the theme of Information Scent, the (imperfect) perception of the value, cost or access path of an information source obtained from proximal cues, such as bibliographic citations, links, icons, or summaries.

Bates [Bat90] analyzes information retrieval systems in terms of 1) the degree of user involvement vs. system automation and 2) the size, or chunking, of activities that the user can direct the system to do at once and discusses certain useful configurations of these two dimensions.

The Microsoft MSDN website (http://www.msdn.com) has an interesting feature that displays a result set document within its context in the MSDN document hierarchy. A query results in a standard list of matching document titles with summaries. When one is selected, this displays the full document with a left panel with the expanded hierarchy path to that document. Figure 10 shows the initial result set formed by the query "data warehouse design" on the MSDN site. Clicking on the top hit presents the target document with its context on the left. This technique is somewhat similar to the dynamic browse–tree method described in Section 3.

The work on faceted metadata for flexible organization [YSLH03] by Marti Hearst's group at UC Berkeley is also interesting in that it explicitly allows a user to choose the organization of information at a browsing node or a search result set. The Cha-Cha search engine [CHHL99] provides context for hits by organizing intranet search results dynamically into a hierarchical outline based on offline computation of the shortest paths from the root node to target documents. This is similar to the idea of dynamic browse trees presented in Section 3.

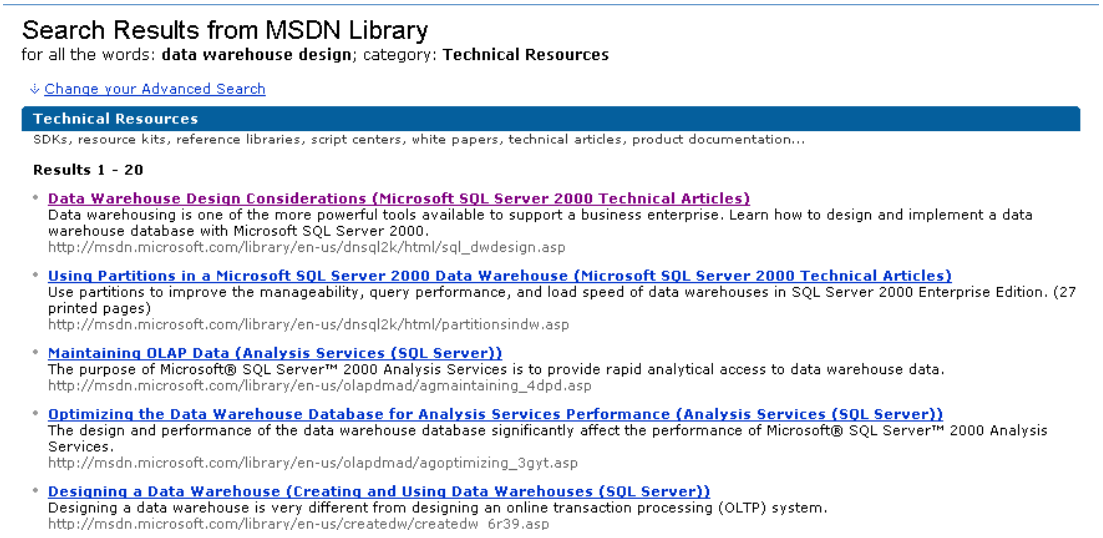
The WebGlimpse prototype [MGS96] allowed "any web site to offer a combination of browsing and searching by automatically analyzing the site, computing neighborhoods, and attaching search interfaces to existing pages. The search was efficient both in terms of time (neighborhoods are explored only at indexing time) and space (only one small index per site)".

The ScentTrails system [OC03] also attempts to integrate browsing and searching. Search begins with keyword matching. Hyperlinks in a web page that point to documents with high relevancy to the query are highlighted in some way (increased font size). There are two problems with this approach: first, any algorithm to estimate information scent could possibly be folded into the original search matching algorithm to begin with; second, why use an algorithmic method to estimate document relevance to a query (scent) when a query-sensitive summary (e.g., implemented as a tooltip over the hyperlink) would let the user decide document relevance without clicking through?

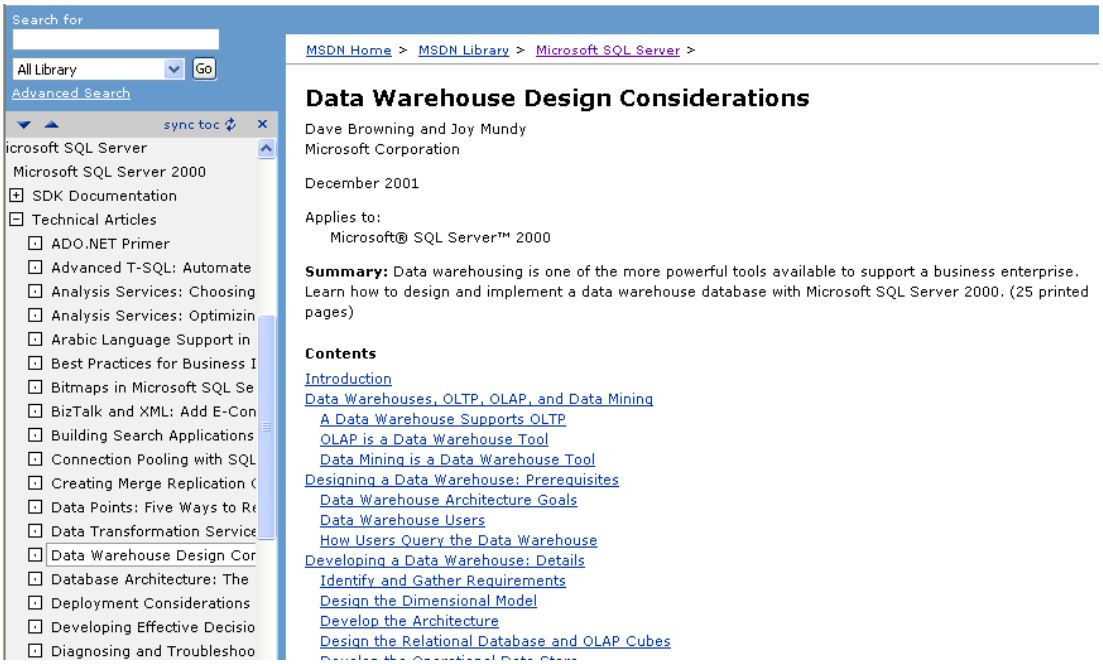
9 Conclusion

Browsing and searching are the two dominant modes for information exploration offered by information retrieval and exploration systems. However these two modes are usually offered separately in terms of both the interfaces and implementations of most information access systems. We argue that this dichotomy is unnatural and sometimes forces users to take multiple steps to achieve what could be done in one step in a better designed system. This paper has suggested techniques to tightly couple the browsing and searching information access methods without removing the distinction between the two. Specifically we have suggested the following techniques under this framework:

1. Hierarchy constrained searching: use the navigation hierarchy to constrain search.
2. Dynamic browse trees: a new method to organize search result sets using the navigation hierarchy.
3. Filtered browsing: continuous querying during browsing navigation.
4. Merge search: merging search result sets across non-contiguous browsing nodes.



(a)



(b)

Figure 10: Screenshots of (a) initial result set listing and (b) target document presentation in context on the MSDN web site for the query “data warehouse design”.

5. Making search result sets equal citizens: allowing browsing node operations on search result set nodes.

6. Flexible organization with multiple category views: allowing a browsing node's contents to be organized in multiple complementary ways, either statically or dynamically.

These techniques are designed to allow the user to use the appropriate method at every step of an information exploration session without losing any work already done. Use of these techniques by designers of information retrieval systems will help create a quicker and more natural information exploration experience for the user.

10 Acknowledgments

Dan Tretter helped clarify many of the ideas presented here during numerous discussions.

11 References

[Bat89] Marcia J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13:407–424, October 1989.

[Bat90] Marcia J. Bates. Where should the person stop and the information search interface start. *Information Processing and Management*, 26(5):575–591, 1990.

[CGNS99] Francine Chen, Ullas Gargi, Les Niles, and Hinrich Schutze. Multi-modal browsing of images in web documents. In *Document Recognition and Retrieval VI*, volume Proceedings of SPIE volume 3651, pages 122–133, 1999.

[CHHL99] Michael Chen, Marti A. Hearst, Jason Hong, and James Lin. Cha-cha: A system for organizing intranet search results. In *USENIX Symposium on Internet Technologies and Systems*, 1999.

[CKPT92] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Fifteenth Interntional Conference on Research and Development in Information Retrieval (Copenhagen, Denmark)*, pages 318–329, June 1992.

[Cox92] Kevin Cox. Information retrieval by browsing. In *5th International Conference on New Information Technology*, pages 69–80. HongKong University of Science & Technology, Kowloon, Hongkong, W. Newton, MA: MicroUse Information, Nov. 30-Dec. 2, 1992.

[CP95] Lara D. Catledge and James E. Pitkow. Characterizing browsing strategies in the world-wide web. In *WWW95*. Georgia Institute of Technology, 1995.

[CW88] J. F. Cove and B.C. Walsh. Online text retrieval via browsing. *Information Processing and Management*, 24(1):31–37, 1988.

[DSG03] Brian Detlor, Susan Sproule, and Chris Gupta. Pre-purchase online information seeking: Search versus browse. *Journal of Electronic Commerce Research*, 4(2):72–84, 2003.

[Fil03] Robert Filman. Weather forecast. *IEEE Internet computing*, Jan/Feb 2003.

[FL95] Christos Faloutsos and King-Ip Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In Michael J. Carey and Donovan A. Schneider, editors, *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, San Jose, California, 22–25 1995.

[Gar03] Ullas Gargi. Consumer media capture: Time-based analysis and event clustering. Technical Report 2003-165, Hewlett-Packard Laboratories, Palo Alto, CA, USA, 2003.

[MGS96] Udi Manber, Burra Gopal, and Michael Smith. Combining browsing and searching, May 1996. A position paper for the W3 Distributed Indexing/Searching Workshop.

[OC03] Christopher Olston and Ed H. Chi. Scenttrails: Integrating browsing and searching on the world wide web. *ACM Transactions on Computer-Human Interaction*, page To Appear, September 2003.

[PC99] P. Pirolli and S. Card. Information foraging, January 1999.

[RSFWH98] Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood, and Andy Hopper. Virtual network computing. *IEEE Internet Computing*, 2(1), January/February 1998.

[YSLH03] Ka-Ping Yee, Kirsten Swearingen, Kevin Li, and Marti Hearst. Faceted metadata for image search and browsing. In *ACM Computer Human Interfaces (CHI)*, 2003.