

# Exploring Discussion Lists: Steps and Directions

Paula S. Newman

Xerox Palo Alto Research Center  
3333 Coyote Hill Road  
Palo Alto, CA., 94304, USA  
+1 650 812 4239  
pnewman@parc.xerox.com

## ABSTRACT

This paper describes some new facilities for exploring archived email-based discussion lists. The facilities exploit some specific properties of email messages to obtain improved archive overviews, and then use new tree visualizations, developed for the purpose, to obtain thread overviews and mechanisms to aid in the coherent reading of threads. We consider these approaches to be limited, but useful, approximations to more ideal facilities; a final section suggests directions for further work in this area.

## Keywords

Email, email archive, discussion list, thread, persistent conversation, treetable, narrowtree, on-line forum

## ARCHIVE REPRESENTATION CHALLENGES

An archived email-based discussion list, public or private, is a relatively new form of publication. It generally represents the conversations of a virtual community drawn together by a shared task (e.g., a standards group), or by a common interest (e.g., motorcycle owners). Archived discussion lists are consulted by:

- Potential list participants, to decide whether subscription or occasional review is of interest.
- New participants of task-oriented lists, to understand group operations and concerns.
- Active list participants, to review prior or current discussions.
- Inactive list participants, to occasionally review archive activity.
- Non-participants, to find information on a topic, to understand the background of a work-product of a working group, or for purposes of sociological or historical research.

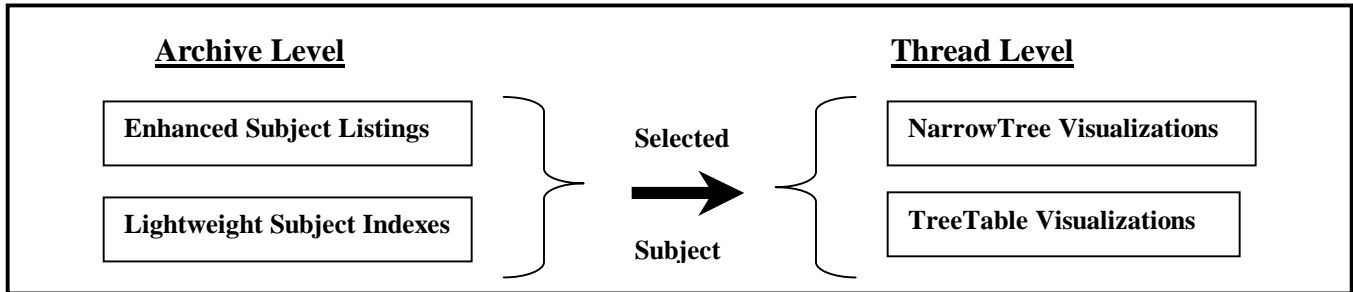
Like other forms of publication, archived discussion lists present form-specific presentation challenges. One major challenge is to convey to readers an understanding of the issues covered by the archive, and to help them find

conversations of interest. This is difficult because, unlike formal publications such as books or journals or conference proceedings, email archives may contain a very large number of components, i.e., the separate conversations (threads), and the components are not naturally organized into meaningful groupings to aid in exploration.

The standard method of portraying the content of an archive, if given separately from a listing of all messages in the archive, generally consists of a list of thread subject lines and message counts. For mailing lists containing 300 or more conversations per month, this approach does not permit easy determination of whether, and to what extent, the archive contains material of interest to a particular user.

A second major challenge for representing archived discussion lists is to portray the individual threads so that they can be read coherently and efficiently. While email threads are often called “persistent conversations”, they are not conventional turn-taking sequences. Instead, an initial message may generate multiple, simultaneous responses, each addressing a different aspect of that message, and each of those responses may have the same effect, giving rise to multiple, simultaneous sub-conversations. So representing longer threads in a way that allows the reader to comprehend the issues covered and the different opinions expressed, is a non-trivial problem.

Most current thread representations are based on Hypermail [7], ultimately derived from Intermail [8]. They consist of indented lists of message authors, representing (approximate) message/reply sequences. Individual messages may then be selected for reading, one by one. Recently Google Groups [5] has provided an incremental improvement, in which an indented author list reflecting the thread structure is given in the left frame of a display, and the full messages are concatenated in the same order in the right frame. However, since messages often contain a large amount of extraneous material, reading messages remains a “one at a time” sort of thing. Also, for long, complex conversations covering several subtopics, it is not at all clear what those subtopics might be, or where they might be found.



**Figure 1: MailContent Facilities**

The MailContent system is an ongoing effort to substantially improve the accessibility of large-scale email archives. It currently incorporates a collection of archive- and thread-level facilities outlined in figure 1 and described in the next two major sections. We consider these facilities to represent a first step, exploiting some specific characteristics of email messages, and combining the result with new tree visualizations, to obtain approximations of more ideal capabilities. The characteristics used are: (a) messages tend to be relatively direct, so that subject lines and initial message fragments provide useful overview material, and (b) messages contain considerable extraneous material that, when removed, often yields readable, relatively short, conjoined message sequences. The last part of the paper discusses important omissions, and directions for further development.

**MAILCONTENTARCHIVE-LEVEL FACILITIES**

MailContent currently provides two types of archive level facilities: *enhanced subject listings*, and *lightweight subject indexes*, discussed in sequence, below. The facilities are intended to provide reasonable overviews of active, voluminous discussion lists, and assistance in finding conversations of interest.

**Enhanced Subject Listings**

Both extensions made to conventional subject listings are shown in figure 2, which contains the beginning of a thread list for a month of the python discussion list. First, by option, threads are listed together with the first few lines of their initial messages. Because thread-initial messages tend to be direct, the combination of subject lines and initial fragments provides a useful indication of whether the thread is of interest to a user.

Second, again by option, threads are listed in order of decreasing length, either overall or within a particular time period. For an initial perusal of a list archive, this ordering indicates the kinds of issues of most interest to list participants. For an occasional review, the ordering indicates the “hot topics” for a time period. It provides a type of “collaborative” filtering for threads, without requiring any overt rating actions.

**Lightweight Subject Indexes**

Lightweight subject indexes are a departure from conventional archive-level views. The left frame of Figure 3 is taken from a generated subject index covering about a year (50000 messages) of the python discussion list.

2000	Feb 2001	ChangeOrder: (earliest) (latest)	ChangeIntros: (hide)
<a href="#">September</a>	<a href="#">CPAN functionality for python</a>	(126)	02/11/01 - 03/02/01
<a href="#">October</a>	(ssthapa) I've created a sort of proof of concept that implements something similar to perl's CPAN. Right now, it allows basic user interaction and should be able to download and install modules. At this point, I want to know ....		
<a href="#">November</a>	<a href="#">. Python 2.1 function attributes</a>	(91)	01/26/01 - 02/09/01
<a href="#">December</a>	(Katz) Hello.. I posted before a much longer version of this question but I got only one response. What do function attributes give that cannot be accomplished with function objects?		
2001	<a href="#">New to OO concepts - re-usability</a>	(88)	02/20/01 - 03/28/01
<a href="#">January</a>	(Gagne) I just had an interesting conversation today with my brother who is learning programming from scratch. He's learning Smalltalk, and was watching me debug a program I'm working on and we visited the subject of reuse. One of ....		
<a href="#">February</a>	<a href="#">What to do after Python?</a>	(87)	02/18/01 - 02/21/01
<a href="#">March</a>	(Eaton) I am learning Python as a first language, and I have been wondering what is a good language to learn after Python? I'm thinking of either going into C or Java but I'm not sure which one because I've heard many arguments ....		
<a href="#">April</a>	<a href="#">Nested scopes resolution -- you can breathe again!</a>	(61)	02/22/01 - 03/05/01
<a href="#">May</a>			
<a href="#">June</a>			
<a href="#">July</a>			
<a href="#">August</a>			
<a href="#">September</a>			

**Figure 2: Enhanced Subject Listing**

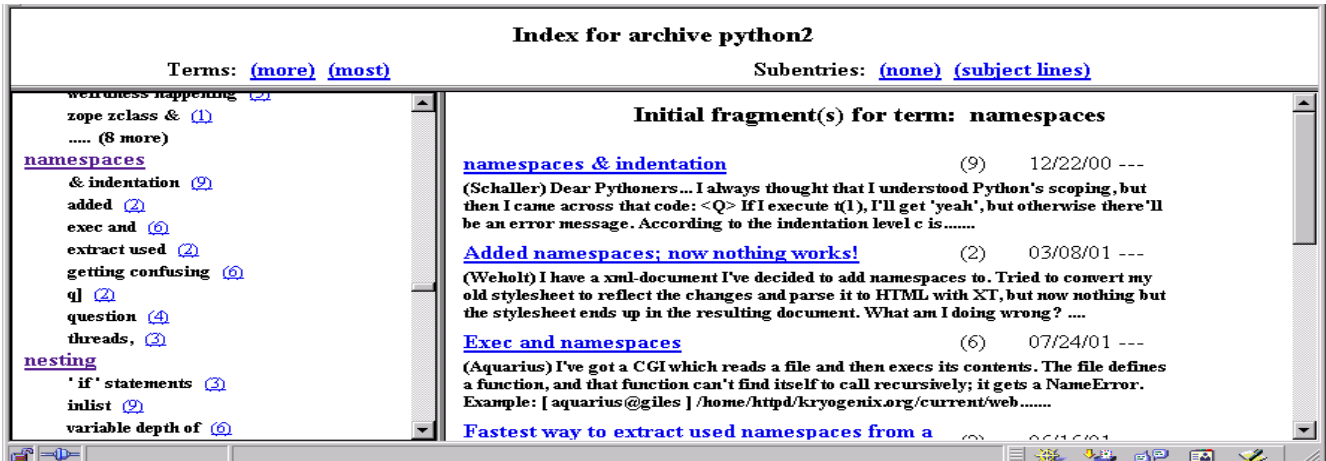


Figure 3. Lightweight Subject Index

In the “back of the book” index style used, headwords are taken from subject lines, and subentries give the contexts in which they appear in those subject lines. Selecting an entry or subentry, such as “namespaces” in figure 3, produces a listing in the right-hand frame of the threads associated with that entry.

The indexes serve several purposes depending on index length. A standard size index, of which figure 3 is an excerpt, is relatively short, and contains headwords found in many subject lines. It serves as an indicator of the more important topics of the collection, and can be used to determine whether a discussion list contains a substantial amount of material of interest to a reader. Most of the headwords in the standard size index for the above corpus can be seen in figure 4, where subentries are omitted and frame widths adjusted appropriately so that the full list can be seen.

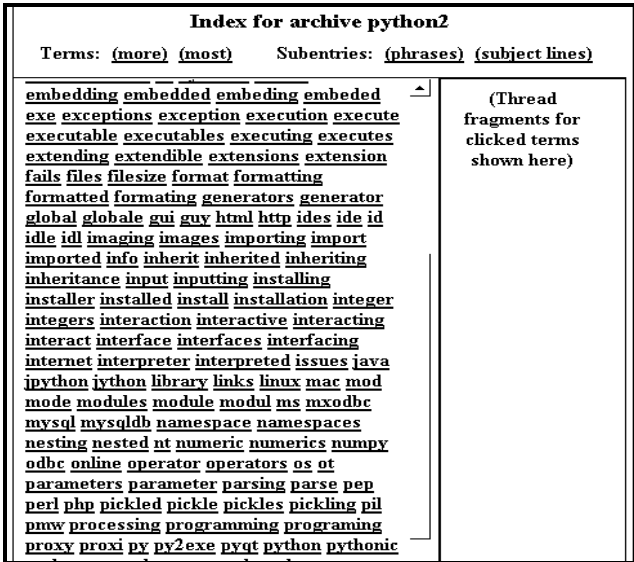


Figure 4. Subject Index Without Subentries

Alternative indexes, selected by the “more and most” options of Figures 3 and 4, obtain more inclusive listings that serve as alternatives to keyword-based search, and are useful when precise keywords are unknown. (Note: indexes over shorter time periods yield more specialized headwords than those shown in Figure 4).

**Subject Index Construction**

The index construction method requires almost no linguistic processing and minimal computational resources. Candidate headwords for a subject index consist of those words occurring in subject lines whose lemmatized forms do not occur in a list of the 5000 or so most frequent words of English.

The extracted lemmas are ranked based on their relative importance in the archive, evaluated as a simple linear combination of the number of thread subjects in which they appear, and the summed lengths of those threads, and the highest-ranking ones are selected. Note that, in contrast to word-usage related topic identification techniques, we are seeking individual words that are most indicative of archive concerns, rather than word usage patterns that distinguish groups of documents or document sections.

How can such an approach to headword selection be effective? Essentially because, on many lists, (a) subject lines usually capture the major issue being raised, and (b) the terms used tend to be specialized ones. This is the case for many, if not most, task- and interest-based lists, on subjects ranging from programming languages to medicine to gardening. Also, the word frequency list can be modified to remove words that are technical terms for a particular archive (e.g., “window” for archives relating to computing).

Subentries are formed by a method of equal simplicity. Given a subject line and a contained headword, a context is found consisting of the minimal string containing the

headword that is bounded on both sides by a “barrier”, which is either a subject boundary or a token from a small list of pronouns, determiners, prepositions, and punctuation marks. If the result is a single word, a sequence of barrier words to one side of the headword is heuristically added if possible, and then the minimal string containing that combination, bounded by further barriers, is found.

This simple context determination approach generates entries that are visually similar to those of a manually produced index, largely because subject lines tend to be noun phrases or some predictable alternative (“How can I ...”). Also, where the context determination method fails, the result is just a somewhat longer context.

### Work Related to Archive-Level Views

While there is considerable work on topic identification and browsing for general collections, very little relates directly to email archives.

Conversation Maps [13] do address email archive browsing. They focus on social networks within discussion lists, and (among many interesting aspects not discussed here) connect sets of people with archive “themes”. The themes are identified as highly salient chains of nouns in quote-related messages that fall into related WordNet [4] categories, using a method similar to that used in [6]. The approach requires, at the very least, lexical reference material, which is often unavailable for terms on specialized lists, as well as part-of-speech identification. Also, while the result is undoubtedly valuable, further study is probably needed to determine the extent to which the themes cover the archive excerpt involved, and the extent to which they serve as convenient overviews.

The work most closely related to lightweight subject indexing is a prolific research direction exemplified by [15] and [16], producing hierarchic indexing phrases or key phrases for a general document collection, and using them for browsing via scrolled collection vocabularies or search. Because the phrases are extracted from full texts, sophisticated methods are needed for phrase isolation and selection. In contrast, the method described here makes the vastly simplifying assumption that the most important phrases are those that (a) are found in subject lines and (b) contain specialized terms. For email archive overviews, this may be sufficient.

Finally, there is the large amount of work on increasingly sophisticated statistical approaches to unsupervised document clustering based on word usage similarities, such as that of [21]. Such methods, applied to full message texts, would expose topics not directly mentioned in subject lines. However, the word-related results of such methods generally consist of not-easily-scannable word-lists for each cluster, where the

individual words are not necessarily suitable for use in term-based corpus overviews.

### MAILCONTENT THREAD LEVEL FACILITIES

When a thread is selected from one of the archive-level views discussed above, a default thread visualization appears, along with a choice of some alternatives. The thread visualizations currently incorporated into MailContent are intended to provide useful thread overviews, and to facilitate coherent, efficient reading of threads, in whole or in part.

The visualizations are all variations on two basic forms. One form, called a *narrow tree*, is an incremental improvement to linear indented trees facilitating text embedding. A second form, called a *treetable*, is a 2D representation that gives a clearer picture of thread structure, and also serves as a guide to deeper exploration of contained sub-conversations.

Both forms embed not entire messages, but some or all of their *essential texts*, that is, the texts shorn of extraneous material such as prefixed or suffixed messages and their introductory material, and various kinds of contact material, and with other excerpts abbreviated.

Also, both forms depend on detailed thread structure analysis identifying, for each message, a “best” thread predecessor. While it is acknowledged that a particular message may take into account several prior messages, using a strict tree structure imposes a necessary organization. We discuss below first the *narrow tree* representation, then some details of message and thread structure analysis, and then go on to the more novel *treetable* representation.

#### Narrow trees

An overview of a randomly selected thread from the rec.motorcycles newsgroup in “narrow tree” form is shown in figure 5. The conversation ordering is not shown by simple indentation, but by an alternative method that allows a narrower representation, and one more suited to text embedding. Briefly, a message is indented under its predecessor only if the predecessor has more than one response. In the latter case, each response is indented and preceded by a dividing line.

So, for example, in figure 5, message 2828 is a response to initial message 2800, and message 3391 is the only response to 2828. Message 2801 is also a response to 2800, message 2894 is the only response to 2801, and 3390 the only response to 2894, etc. The contained initial fragments replace all quotes with “<Q>”.

OutLook®, and possibly other email clients, also embed initial fragments in message listings. However, they do

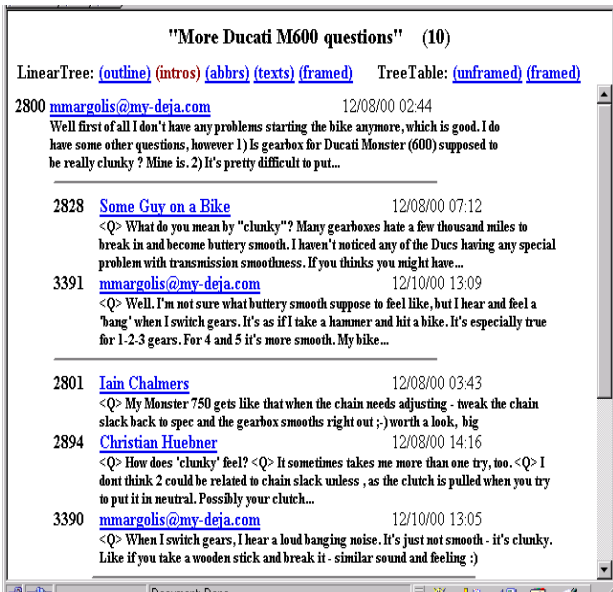


Figure 5. NarrowTree Thread Overview

not elide initial quotes, and doing so is crucial for reaching the actual content. The resultant narrow tree displays provide reasonable thread overviews, at least for short to medium length threads where the entire listing can be seen in two or three screens.

Figure 6 shows a framed narrow tree view of another rec.motorcycles thread, intended for message reading. The left hand frame contains an outline of the thread, and the right-hand frame contains a scrolled portion of the concatenated “essential texts” of the thread, both in narrow tree form. The reduction in message sizes achieved by the use of essential text is fairly dramatic. It essentially reduces the space required to represent email threads to that needed for threads of an on-line forum, and allows reading of the concatenated messages without

having to skip over large, intervening attached messages, signature blocks, etc. This is especially effective when, as in the thread shown in figure 6, much of the thread actually consists of short messages that include, by nested quotes, all previous messages.

This approach allows efficient reading of relatively short threads containing relatively short messages, but is less of an improvement on more traditional representations for long or bushy threads, as it is difficult for the reader to maintain context in the face of frequent shifts in the relevant predecessors.

### Message and Thread Structure Analysis

The reduction of messages to essential texts is obtained by a message analysis procedure that begins with a recursive descent phase to isolate excerpts and their introductions. A weighted finite state analyzer is then applied to message bodies at a particular level, with quoted passages masked. The analyzer produces a network that associates, with each line, one or more categories and category-resemblance weights, based on manually coded tests of heuristically developed features. The categories are such things as “greeting”, “closing”, “signature block line”, and “prose line”. Then the highest weighted path through the entire message is selected as the one used in the extraction of essential texts.

The method is similar to one described by Chen et.al. [3], used to analyze messages for a text-to-speech application, and works quite well, but not perfectly, on usenet newsgroup messages. However, in both versions, the features and weightings are explicitly coded and (in our system) are continually being refined, so more automated techniques would be useful. In this regard, McCallum et.al. [14] describe an approach for training an analyzer to partition messages within an FAQ digest.

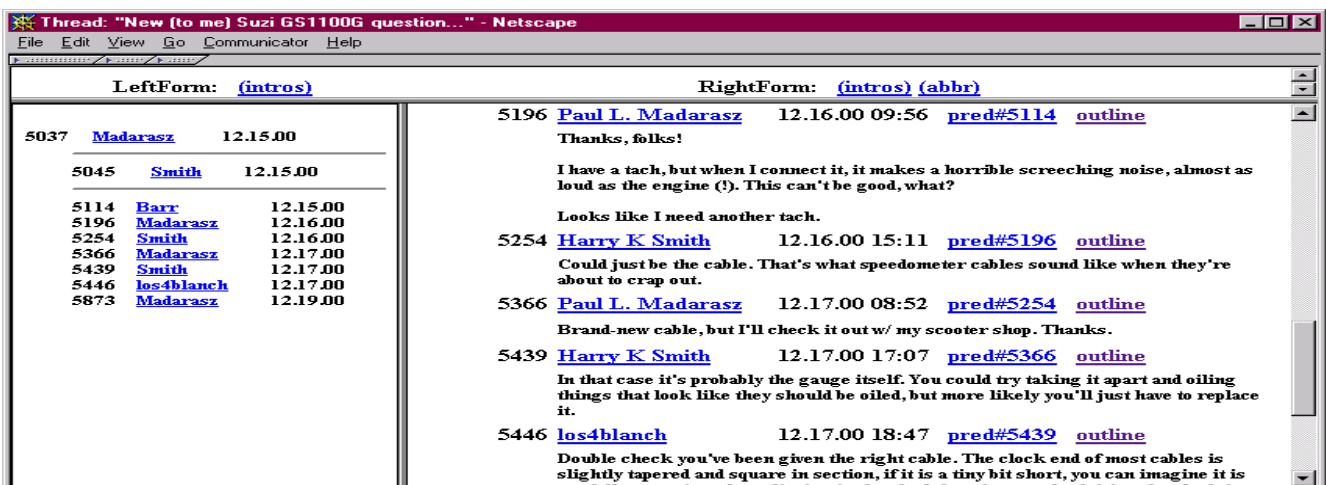


Figure 6: Framed Narrow Tree

The partitioning is into four consecutive elements: header, question, answer, and "tail". After identifying a set of features relevant to the categorization, such as the existence of a question mark, the analyzer is trained on a corpus with marked up line-labels. While it remains to be seen whether this method can be applied to delineating the components of more general messages, the automation is certainly of interest in the context of corpus-specific, and ever changing, message structure conventions.

**Thread structure analysis** consists of finding a best predecessor for each message, by combining evidence from explicit header fields (e.g., "in-reply-to"), excerpted message headers and quote introductions, and, like the work of Tajima et.al. [23], by matching quoted passages with their sources. Here, however, the matching is based on sentences or sentence parts rather than lines, to handle sentence-based quoting, as well as differences between the line partitioning in an original message, and in the quoted version created by some mailers.

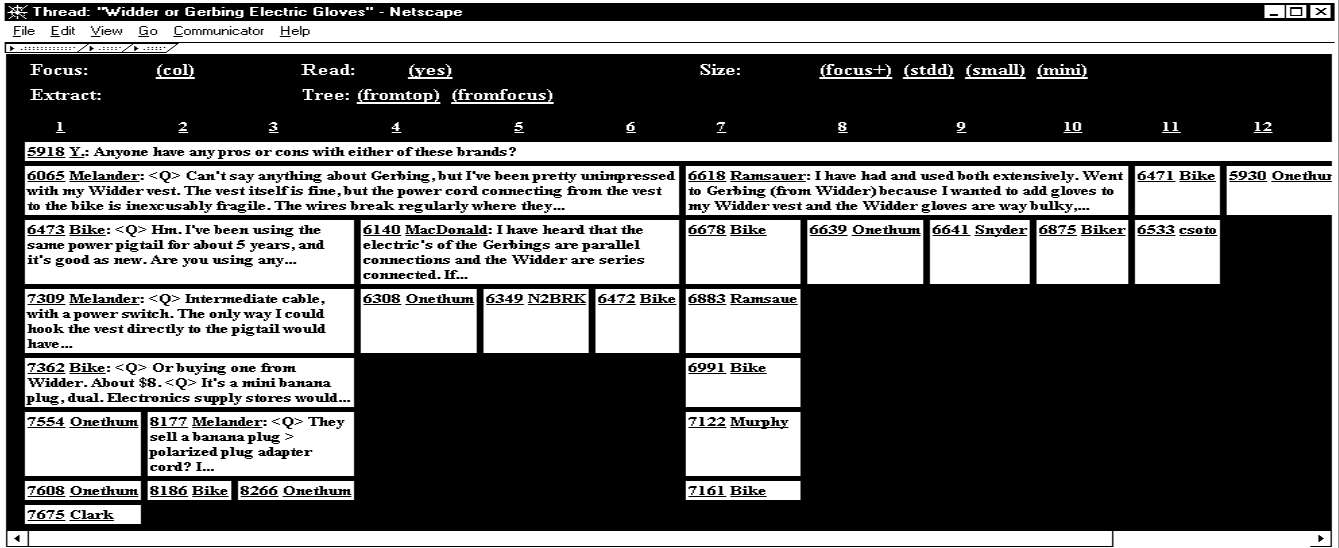


Figure 7. TreeTable

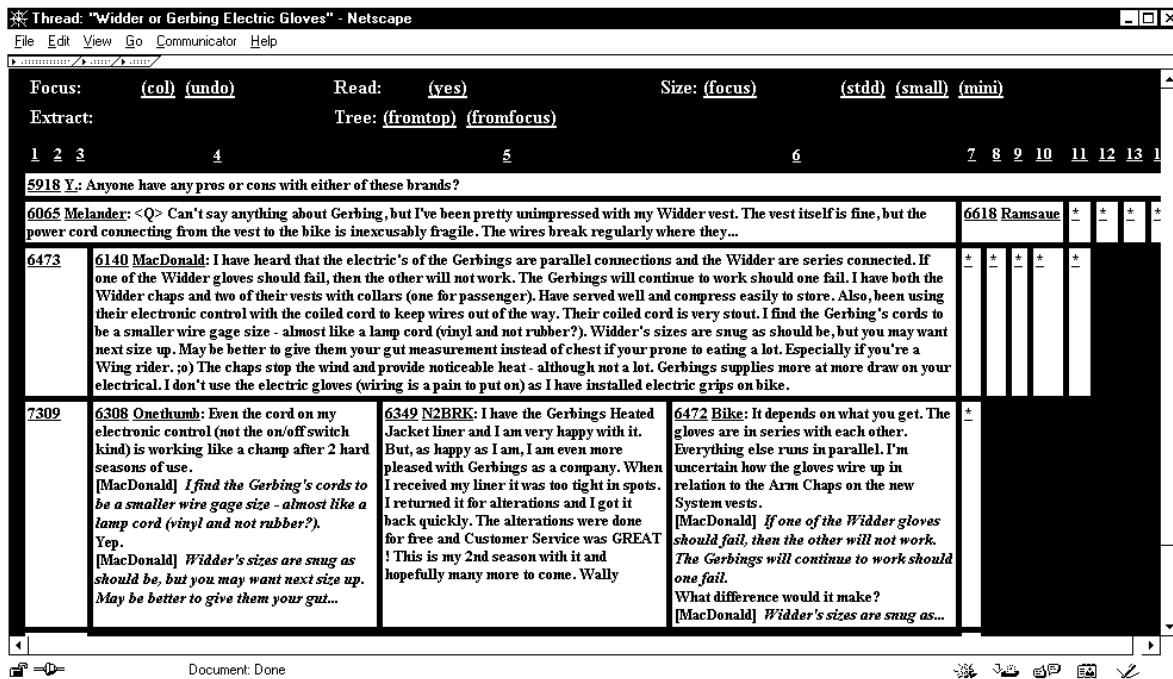


Figure 8. Focused TreeTable

## Treetables

Treetables and their associated tools represent a departure from conventional thread representations. Figure 7 shows a treetable-based overview for a medium length thread of 29 messages. Each column of the table represents a single path from the root of the thread tree to a leaf. Also, each cell in a row  $i$  exactly spans the cells representing its responses in the next row  $i+1$ .

In essence, treetables adapt conventional node + edge tree representations to allow considerable text embedding and, by using strictly linear paths, more fully realize the potential of the 2-D form for portraying structure. The columns of a treetable represent the synchronous conversations of the thread. The successors of a particular node show all responses to a particular contribution. The combination is intended to provide a good overview of thread content within a firm structural context, and to serve as a base for more detailed study.

Cell sizes for initial treetable overviews are obtained by (a) setting the base column width to a minimum that allows a selectable item for each message, and then (b) expanding that width until a nominal window size is reached. Text is inserted into cells to the extent feasible, and the basic geometry ensures that more space is available for text in the bushiest subtrees, which may be of the most interest.

The treetable thread overviews are used as a base for more detailed explorations of the messages of the thread, either by in-situ focus operations or by extraction of concatenated texts. Focus operations, illustrated in figure 8, obtain deeper overviews and, in many cases, the full content of a subpart of the display. They are adapted from "focus + context" ("fisheye"[17]) approaches, in particular ones like Table Lens [19]. The adaptation allows either subtrees or individual columns to be selected as foci, and to be expanded to different degrees. In figure 8 the subtree headed by message 6140 is selected for maximal expansion. The texts used for deep focus operations are the same "essential texts" used in the narrow tree representation, but heuristically truncated where necessary.

For full reading of all or part of a thread, treetables are used as guides to auxiliary displays presenting the full essential text of either a column, as shown in figure 9, or all the immediate responses to a message (not shown). Displays of column texts, as in figure 9, approximate turn-taking synchronous conversations, and allow fully coherent, uninterrupted reading. Displays of response texts collect all the initial reactions to a given message.

Treetables can also be used to represent extremely large threads. For such threads, extraction operations, as

shown in the upper part of figure 10, are provided to more closely examine specific subtrees or sets of columns. However, as discussed further in the final section, we believe that more powerful methods are needed for an adequate presentation of longer threads.

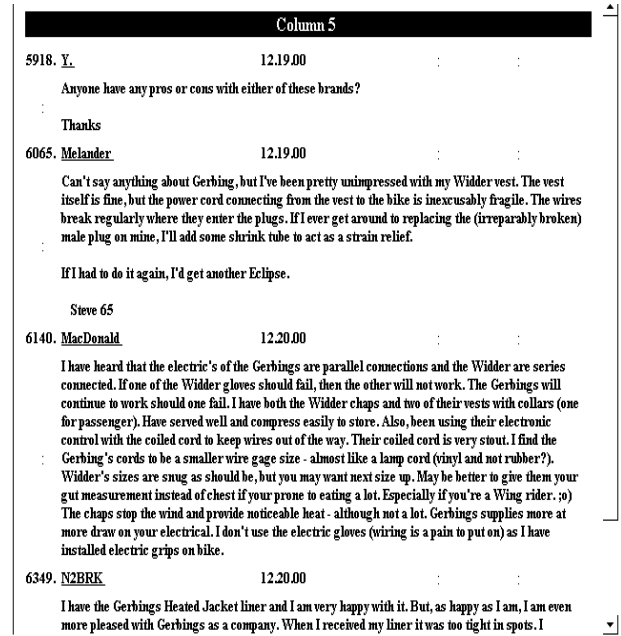


Figure 9. TreeTable Column Text

## Related Thread and Tree Representation Work

There are several kinds of experimental approaches to thread presentation that relate to the work described here. First, a radical approach to thread portrayal is described by Yabe et.al. [24]. In that approach the text of a message is recursively interspersed with responses to particular passages, and the elements are labeled by contributor avatars to further emphasize the conversational aspect. While the message integration approach, as described, founders on logistic problems, such as the fragmentation of logically-related text by inserted responses, it may be very useful in specific circumstances, such as for short responses having no successors.

Two other approaches use versions of classic node-plus-edge tree representations to represent threads. The thread analysis and display work of Smith and Fiore [22] is primarily devoted to illustrating communication patterns among thread participants. However, they illustrate thread structures by classic trees containing tiny nodes, arranged so that paths are shown vertically wherever possible. The nodes are insufficient to contain any identifying text, but the representation conveys the structure of the conversation rather clearly.

Also, in a message input approach described by Popolov et.al [18], the user views a lattice representing the state

of the thread, and then identifies the nodes to which their own input is a response. This work is interesting because it tries to explicitly capture the actual structure of a thread, in which a given response often takes into account several earlier messages even though it appears to formally respond to only one.

Finally, in the area of tree representations we should also mention “treemaps” [9], a well-known “edgeless” tree representation. While the lack of edges gives some treemap instances a superficial similarity to treetables, they differ in both layout and intent. Briefly, treemaps represent hierarchic structures by nested rectangles whose sizes are proportional to arbitrary additive attributes of their contained leaf nodes.

## DISCUSSION

We have described an operational, integrated system for email archive exploration combining analysis and presentation techniques to obtain serviceable archive and thread overviews, and to expedite the reading and assimilation of subject matter of interest. The archive overviews, consisting of enhanced subject listings and subject-line-based indexes, provide organizationally meaningful alternatives to traditional subject listings, and supply indicative content information. On the thread level, new tree representations geared to text embedding and, in the case of treetables, structural transparency, are used as thread overviews and as guides to deeper exploration both within the representations and via extraction of coherent, concatenated message sequences.

The system heavily exploits specific characteristics of email messages, namely that (a) large numbers of messages are relatively succinct and to-the-point, so that initial thread fragments and lightweight subject indexes go a considerable way toward conveying corpus content, and (b) messages contain considerable extraneous material that, when removed, often yield readable, relatively short, conjoined message sequences.

These characteristics allow us to approximate more ideal representations by very simple methods. But the existing facilities are just approximations, especially in the areas of thread and archive overviews, and there are significant gaps in the system.

### Thread level directions

Looking first at thread overviews, it is clear that representational improvements are needed for larger threads. Beyond 40 messages or so, it becomes difficult for a user to grasp the scope of the contained subject matter by a scan of initial fragments in whatever form, in order to decide whether to read the thread at all, and, if

so, what parts. This is especially problematic because, as threads increase in length, they tend to deal with multiple subtopics, either stemming directly from different facets of the initial message, or representing outgrowths of the initial discussion. Also, for very long threads, which are common on high-volume lists, the methods provided here are adequate only for sampling conversations coherently; reading entire threads is simply too laborious.

To illustrate the problems graphically, figure 10 shows an outline-form treetable overview for a thread containing 160 messages. While the user may extract portions of the thread for study, there is little guidance (except perhaps in terms of subtree volumes) as to what portions they might extract, or try to read in full. The missing element is a thread summary isolating the major subtopics of the conversation and, for each subtopic, providing at least a sampling of the more significant statements of each.

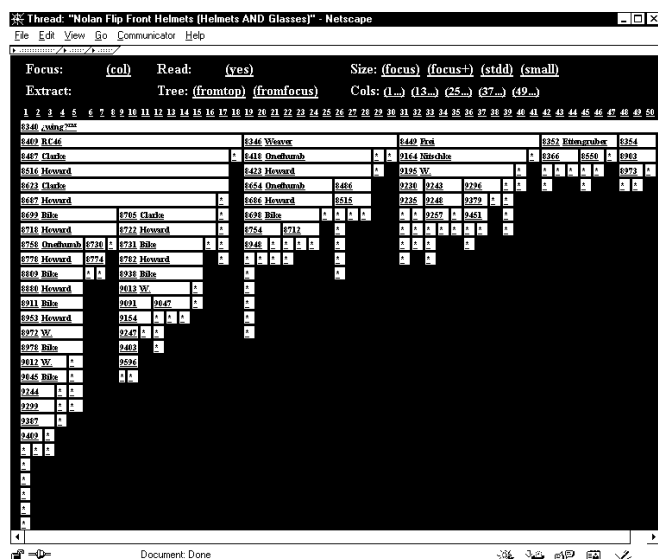


Figure 10. Outline form treetable for long thread

It is beyond the scope of this paper to itemize and discuss the many types of work that can be drawn upon, in the areas of topic and topic shift identification, and summarization, in order to develop thread summaries. However, it should be mentioned that such work requires adaptation to the email thread context. For example, to adapt clustering methods based on word-usage patterns to thread segmentation, it is necessary to determine how to account for quoted words, probably differentiated by message quoting style. Similarly, to adapt feature-based summarization [11, 12] methods, email-specific features must be identified (presumably by training on example corpora). Finally, multi-document summarization [13] seems of major importance, because of the considerable

amount of repetition especially within threads that begin with questions or requests for technical help.

### Archive Level Directions

Thread subtopic identification is also important for several aspects of archive-level facilities. First, while subject lines plus initial message lines do convey considerable information, archive overviews should also take into account embedded thread subtopics in forming extended subject line listings and subject indexes. Thread subtopics are also important in search operations, which have not yet been incorporated into the system. The results of such searches should be in terms of ranked sub-conversations, an approach studied to a limited extent by Tajima et.al. [23].

Finally, while we really don't know what kinds of archive overviews would be most useful, and that is, in itself, an important topic for further study, it is likely that some sort of conceptual hierarchy of topic identifying terms or phrases would be desirable. This is problematic for email corpora because so many of the terms used are very specialized, and do not appear in lexical references intended for computational use, such as WordNet [4]. However, new structured reference material is starting to emerge based on a number of initiatives, such as the Semantic Web [2], and taxonomies may also be derived from textual definitional sources [10].

Moving now to the most significant gap in the current system, the facilities incorporated to date completely neglect the social aspects of email archives. While many such aspects might be of interest, the most useful are probably the linkages between individuals, and the kinds of issues they discuss (as in Conversation Map [20]), and the kinds of positions they take.

There are at least two reasons for this. First, many lists serve as virtual communities, with discussions that range beyond the ostensible concerns of the list. Therefore, as for any other community, it is helpful for new members to get to know the people involved. Second, especially on task-based lists, a participant or group may focus on a set of closely related discussions, and/or take a set of closely related positions, sometimes deriving from the interests of an organization they represent. Thus highlighting those relationships would contribute an important dimension to the comprehension of the list discussions. However, isolating and summarizing the interests and positions of particular individuals or "parties", is not only a major technical challenge, but also a very delicate matter that risks discouraging participation.

### ACKNOWLEDGEMENTS

The author is deeply indebted to Michelle Q. Wang Baldonado for her initial observation that excerpted messages are often redundant, and that their elimination would facilitate the presentation of an entire thread as a single readable document. I am also indebted to Stuart Card for his contributions to treetables, and to Lance Good for naming them. And I thank Mark Stefik, manager of PARC's Information Systems Technology Laboratory, for his support and encouragement, Bill Janssen for providing many archived corpora in accessible form, and other members of the Human Document Interactions group for their useful critical comments on this work.

### REFERENCES

1. Barzilay, R., and Elhadad, M., Using Lexical Chains for Text Summarization, *Proceedings of ACL/EACL '97* (Madrid, Spain, June 1997) 10-17
2. Berners-Lee, T., Hendler, J., Lassila, O., The Semantic Web, *Scientific American* (May 2001) 35-43
3. Chen, H., Hu, J., and Sproat, R.W., Integrating Geometrical and Linguistic Analysis for Email Signature Parsing. *ACM Transactions on Information Systems* 17,4 (Oct. 1999) 343-366
4. Fellbaum, C., ed., *WordNet: An Electronic Lexical Database*, MIT Press, May 1998
5. Google groups. Available at <http://groups.google.com>
6. Hirst, G., and St-Onge, D., Lexical Chains as Representations of Context for the Detection of Malapropisms, in Fellbaum, C. (ed), *WordNet: An Electronic Lexical Database*, MIT Press, Cambridge MA, May 1998
7. Hypermail development archive. Available at <http://www.hypermail.org>
8. Jackson, S., and Yankelovich, N., Intermail: a Prototype Hypermedia Mail System. *Proceedings of Hypertext '91* (San Antonio TX, Dec. 1991) 405-409
9. Johnson, B., and Schneiderman, B., Treemaps: a space-filling approach to the visualization of hierarchical information structures. *Proc of 2<sup>nd</sup> IEEE Visualization Conference* (1991).
10. Klavans, J., and Whitman, B., Extracting Taxonomic Relationships from On-Line Definitional Sources using LEXING, *Proceedings of JCDL '01* (Roanoke VA, June 2001) 257-258
11. Kupiec, J., Pedersen, J., and Chen, F., A Trainable Document Summarizer, *Proceedings of SIGIR '95*, (Seattle, WA, July 1995) 68-73

12. Marcu, D., *The Theory and Practice of Discourse Summarization*, MIT Press, Cambridge MA, November 2000
13. McKeown, K., Klavans, J., Hatzivassiloglu, V., Kan, M-Y, Schiffman, B., and Teufel, S., Towards Multidocument Summarization by Reformulation: Progress and Prospects, *Proceedings of AAAI-99* (Orlando, FL, July 1999) 453-460
14. McCallum, A., Freitag, D., and Pereira, F., Maximum Entropy Markov Models for Information Extraction and Segmentation. *Proceedings of ICML '00* (Stanford CA, June 2000) 591-598
15. Nevill-Manning, C.G., Witten, I.H., and Paynter, G.W., Browsing in Digital Libraries: a Phrase-Based Approach, *Proceedings of ACM DL 1997* (Philadelphia, PA, July 1997) 230-236
16. Paynter, G., Witten, I., Cunningham, S., and Buchanan, G., Scalable Browsing for Large Collections: a Case Study, *Proceedings of ACM DL 2000* (San Antonio, TX, June 2000) 215-223
17. Plaisant, C., Carr, D, and Schneiderman, B., Image-Browser Taxonomy and Guidelines for Designers, *IEEE Software* 12,2 (March 1995) 21-32
18. Popolov, D., Callaghan, and M., Luker, P., Conversation Space: Visualizing Multi-Threaded Conversation. *Proceedings of AVI 2000* (Palermo, Italy, May 2000) 246-249.
19. Rao, R., and Card, S.K., The Table Lens: Merging Graphical and Symbolic Information in an Interactive Focus + Context Visualization for Tabular Information. *Proceedings of CHI '94* (Boston MA, April 1994) 318-322
20. Sack, W., Conversation Map: A Content-based Newsgroup Browser, *Proceedings of IUI '00* (New Orleans, LA, January 2000)
21. Slonim, N., and Tishby, N., Document Clustering using Word Clusters via the Information Bottleneck Method, *Proceedings of SIGIR '00*, (Athens, Greece, July 2000) 208-215
22. Smith, T., and Fiore, A., Visualization Components for Persistent Conversations. *Proceedings of CHI '01* (Seattle WA, May 2001), ACM Press, 136-143
23. Tajima, K., Mizuuchi, Y., Kitagawa, M, and Tanaka, K., Cut as a Querying Unit for WWW, Netnews, E-mail. *Proceedings of Hypertext '98* (Pittsburgh PA, June 1998) 235-244
24. Yabe, J., Takahashi, S., and Shibayama, E., Automatic Animation of Discussions in USENET. *Proceedings of AVI 2000* (Palermo, Italy, May 2000) 84-91.