

# Email Archive Overviews using Subject Indexes

**Paula S. Newman**

Xerox Palo Alto Research Center,  
3333 Coyote Hill Road  
Palo Alto, CA 94304 USA  
pnewman@parc.xerox.com

## **ABSTRACT**

Archived discussion lists are becoming significant reference sources. This paper describes a new type of overview for such lists, using a back-of-the-book style index containing headwords selected from subject lines and subentries derived from their subject-line-contexts.

## **Keywords**

Email, email archive, discussion list, automatic indexing

## **INTRODUCTION**

Archives of public and private discussion lists have become useful sources of information. However, their utility can be improved by enhanced overview facilities, allowing users to determine whether an archive contains material of interest, and to locate that material. Only two types of facilities are currently available for high-level archive exploration: (a) lists of subject lines, and (b) keyword-based searches. The first can be daunting for high-volume lists containing over 400 threads per month. The second is useful only if the user knows what contained material might be of interest, and also how to identify it.

## **EMAIL SUBJECT INDEXING**

Email subject indexing addresses this issue in a lightweight, but, we believe, effective way. Figure 1 shows a short excerpt from a generated subject index covering 50000 messages of the comp.lang.python newsgroup (representing about a year).

The listing is in the form of a “back-of-the-book” index, with headwords followed by subentries giving the contexts in which they appear in subject lines of the archive. Selecting an entry or subentry obtains a list of the threads associated with the entry, with their initial message fragments. In figure 1 the selected headword is “namespaces”. Most of the headwords selected can be seen in figure 2, where subentries are omitted and frame sizes adjusted appropriately. Indexes over shorter periods tend to yield more specialized topics.

## **Headword Selection**

Index headwords are selected by a scan of subject lines to isolate terms not occurring in a list of the 5000 or so most frequent words in the language (in this case English). The terms are then ranked based on their relative importance in the archive, which is influenced by the number of thread subjects and by the lengths of those threads, and the highest-ranking terms are selected.

How can such a method be effective? Essentially because, on most serious discussion lists, (a) subject lines usually capture the issue being raised, and (b) the terms used tend to be specialized ones.

The dependence on the use of specialized terms implies that the method applies primarily to lists where the subjects discussed are not usually identified by common words. But there are many lists of this type, on subjects ranging from programming languages to medicine to gardening. Also, the word frequency lists can be modified for a particular list to remove words that are technical terms for that subject (e.g., “window” for lists relating to computing).

The initial listing of headwords is a relatively short one, and serves as an indicator of the most important topics of the collection. More inclusive indexes are selected by the “more” and “most” options of Figures 1 and 2.

## **Subentry Formation**

Subentries are formed by a method of equal simplicity. Given a subject line and a contained term of interest, the term is put into a context by scanning to the left and right of the term in the subject line, adding words to the context until predetermined “barrier” words, or punctuation marks, are reached. If no words can be added in this way, the restriction on barrier words is heuristically relaxed, and the search proceeds in a selected direction, adding barrier words and then other content words, until another barrier is reached.

This approach is adequate because subject lines tend to be noun phrases or some predictable alternative (“How can I ...”), which, in turn, allows the use of a simple context identification algorithm instead of substantive parsing. Also, where the context determination method fails, the result is simply a somewhat longer context.

Index for archive python2	
Terms: <a href="#">(more)</a> <a href="#">(most)</a>	Subentries: <a href="#">(none)</a> <a href="#">(subject lines)</a>
<p>.....</p> <p><b>namespaces</b></p> <ul style="list-style-type: none"> <li>&amp; indentation(9)</li> <li>[ q](2)</li> <li>added(2)</li> <li>exec and(6)</li> <li>extract used(2)</li> <li>getting confusing(6)</li> <li>question(4)</li> <li>threads_(3)</li> </ul> <p><b>nesting</b></p> <ul style="list-style-type: none"> <li>'if' statements(3)</li> <li>inlist(9)</li> <li>variable depth of(6)</li> <li>zope tag ___ madness(4)</li> </ul> <p><b>nested</b></p> <ul style="list-style-type: none"> <li># pragma( ___ scopes)(14)</li> <li>dictionary(10)</li> <li>function(2)</li> </ul>	<p><b>Initial fragment(s) for term: namespaces</b></p> <p><b>namespaces &amp; indentation</b> (9) 12/22/00 ---          (Schaller) Dear Pythoners... I always thought that I understood Python's scoping, but then I came across that code: &lt;Q&gt; If I execute t(1), I'll get 'yeah', but otherwise there'll be an error message. According to the indentation level c is.....</p> <p><b>[Q] Namespaces</b> (2) 05/27/01 ---          (Cheong) I noticed the following: If I have moduleA.py and moduleB.py, such that the functions defined in moduleB uses those defined in moduleA, an "import moduleA" statement is necessary in moduleB.py. If now moduleB is imported into the interpreter,.....</p> <p><b>Added namespaces; now nothing works!</b> (2) 03/08/01 ---          (Weholt) I have a xml-document I've decided to add namespaces to. Tried to convert my old stylesheet to reflect the changes and parse it to HTML with XT, but now nothing but the stylesheet ends up in the resulting document. What am I doing wrong? ....</p> <p><b>Exec and namespaces</b> (6) 07/24/01 ---          (Aquarius) I've got a CGI which reads a file and then execs its contents. The file</p>

Figure 1.

Index for archive python2	
Terms: <a href="#">(more)</a> <a href="#">(most)</a>	Subentries: <a href="#">(phrases)</a> <a href="#">(subject lines)</a>
<p><a href="#">embedding</a> <a href="#">embedded</a> <a href="#">embedding</a> <a href="#">embedded</a></p> <p><a href="#">exe</a> <a href="#">exceptions</a> <a href="#">exception</a> <a href="#">execution</a> <a href="#">execute</a></p> <p><a href="#">executable</a> <a href="#">executables</a> <a href="#">executing</a> <a href="#">executes</a></p> <p><a href="#">extending</a> <a href="#">extendible</a> <a href="#">extensions</a> <a href="#">extension</a></p> <p><a href="#">fails</a> <a href="#">files</a> <a href="#">filesize</a> <a href="#">format</a> <a href="#">formatting</a></p> <p><a href="#">formatted</a> <a href="#">formatting</a> <a href="#">generators</a> <a href="#">generator</a></p> <p><a href="#">global</a> <a href="#">globale</a> <a href="#">gui</a> <a href="#">guy</a> <a href="#">html</a> <a href="#">http</a> <a href="#">ides</a> <a href="#">ide</a> <a href="#">id</a></p> <p><a href="#">idle</a> <a href="#">idl</a> <a href="#">imaging</a> <a href="#">images</a> <a href="#">importing</a> <a href="#">import</a></p> <p><a href="#">imported</a> <a href="#">info</a> <a href="#">inherit</a> <a href="#">inherited</a> <a href="#">inheriting</a></p> <p><a href="#">inheritance</a> <a href="#">input</a> <a href="#">inputting</a> <a href="#">installing</a></p> <p><a href="#">installer</a> <a href="#">installed</a> <a href="#">install</a> <a href="#">installation</a> <a href="#">integer</a></p> <p><a href="#">integers</a> <a href="#">interaction</a> <a href="#">interactive</a> <a href="#">interacting</a></p> <p><a href="#">interact</a> <a href="#">interface</a> <a href="#">interfaces</a> <a href="#">interfacing</a></p> <p><a href="#">internet</a> <a href="#">interpreter</a> <a href="#">interpreted</a> <a href="#">issues</a> <a href="#">java</a></p> <p><a href="#">jython</a> <a href="#">jython</a> <a href="#">library</a> <a href="#">links</a> <a href="#">linux</a> <a href="#">mac</a> <a href="#">mod</a></p> <p><a href="#">mode</a> <a href="#">modules</a> <a href="#">module</a> <a href="#">modul</a> <a href="#">ms</a> <a href="#">mxodbc</a></p> <p><a href="#">mysql</a> <a href="#">mysqldb</a> <a href="#">namespace</a> <a href="#">namespaces</a></p> <p><a href="#">nesting</a> <a href="#">nested</a> <a href="#">nt</a> <a href="#">numeric</a> <a href="#">numerics</a> <a href="#">numpy</a></p> <p><a href="#">odbc</a> <a href="#">online</a> <a href="#">operator</a> <a href="#">operators</a> <a href="#">os</a> <a href="#">ot</a></p> <p><a href="#">parameters</a> <a href="#">parameter</a> <a href="#">parsing</a> <a href="#">parse</a> <a href="#">pep</a></p> <p><a href="#">perl</a> <a href="#">php</a> <a href="#">pickled</a> <a href="#">pickle</a> <a href="#">pickles</a> <a href="#">pickling</a> <a href="#">pil</a></p> <p><a href="#">pnw</a> <a href="#">processing</a> <a href="#">programming</a> <a href="#">programing</a></p> <p><a href="#">proxy</a> <a href="#">proxi</a> <a href="#">py</a> <a href="#">py2exe</a> <a href="#">pyqt</a> <a href="#">python</a> <a href="#">pythonic</a></p>	<p>(Thread fragments for clicked terms shown here)</p>

Figure 2.

**DISCUSSION AND RELATED WORK**

Our approach does not attempt conventional topic identification, in the sense of delineating disjoint issues, but, rather, tries to give a good idea of archive content by very lightweight methods. We believe the approach is useful, and might be improved by higher-level headword classification, (e.g., mysql → database access). This presents a challenge for all topic identification methods, because the concepts involved are often quite new or specialized, and thus not found in standard lexical resources, of which the most important is WordNet [2].

Our approach is different from that of Sack [1], who identifies, as archive “themes”, chains of nouns in quote-related messages that fall into related WordNet [2] categories, using a method similar to that used in [3]. This impressive work actually focuses on social networks and their interaction with the themes. In contrast, the subject-indexing method provides contextualized concepts in the far less demanding way of combining subject-line headwords with surrounding words as subentries. It does not require the use of any lexical

reference except a high-frequency word list; it is the absence of a word from that list that creates candidate headword status. Also, no part-of-speech identification or parsing is needed for headword identification or subentry creation.

There is also a large literature on statistically based approaches to (unsupervised) topic identification for a group of documents. The methods used, such as that discussed by Slonim and Tishby [4], identify topics based on (highly elaborated versions of) word usage similarities. Such methods, applied to full email texts, would, like that of Sack [1], expose topics not directly mentioned in subject lines. However, the results tend to consist of not-easily-scannable word-lists for each topic.

Two directions of further research are suggested by the above discussion. The first is to determine what in fact constitutes a useful overview for an email archive. The second is to adapt and apply work in area of lexical reference extension, such as [5], to email corpora. We are also interested in methods of displaying changes archive concerns over time.

**REFERENCES**

1. W. Sack, Conversation Map: A Content-based Newsgroup Browser, *Proceedings of IUI '00* (New Orleans, LA, January 2000)
2. C. Fellbaum ed., WordNet: An Electronic Lexical Database, MIT Press, May 1998
3. R. Barzilay and M. Elhadad, Using Lexical Chains for Text Summarization, *Proceedings of ACL/EACL '97* (Madrid, Spain, June 1997) 10-17
4. N. Slonim and N. Tishby, Document Clustering using Word Clusters via the Information Bottleneck Method, *Proceedings of SIGIR '00*, (Athens, Greece, July 2000) 208-215
5. P. Vossen, Extending, Trimming, and Fusing WordNet for Technical Documents, *Proceedings of Workshop on WordNet and Other Lexical Resources* (Pittsburgh, PA, June 2001) 125-131