# Synthesizing composite topic structure trees for multiple domain specific documents

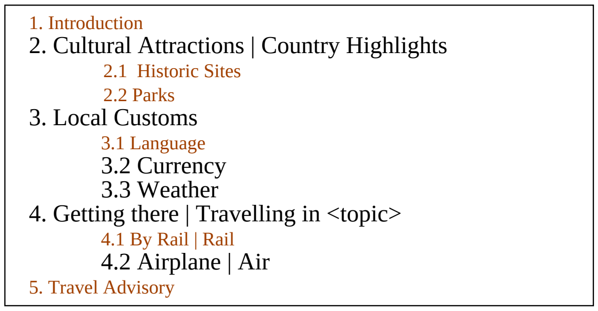
Min-Yen Kan and Judith L. Klavans and Kathleen R. McKeown
Department of Computer Science
Columbia University
New York, NY 10027, USA
{min,klavans,kathy}@cs.columbia.edu

## Abstract

Domain specific texts often have implicit rules on content and organization. We introduce a novel method for synthesizing this topical structure. The system uses corpus examples and recursively merges their topics to build a hierarchical tree. A subjective cross domain evaluation showed that the system performed well in combining related topics and in highlighting important ones.

Documents that address similar subjects for specific purposes often exhibit common structure. Examples of this phenomenon include consumer information on diseases, departmental websites, artist's biographies, company annual reports and travel brochures. For instance, a travel brochure may list the topics of cultural attractions, local customs and airfare information, in that particular order. In some cases, limitations on content and structure may even be codified (e.g. journal submission guidelines).

In this paper, we illustrate a novel method for learning these topical structures across documents. Using multiple example documents belonging to a specific types of text, the system merges the documents' individual topic structures to form composite topics. This process results in a prototypical topic tree that holds a model representation in which the example documents can be viewed as instances. Figure 1 shows a hand-crafted composite topic tree for country travel.



1. Introduction
2. Cultural Attractions | Country Highlights
    2.1 Historic Sites
    2.2 Parks
3. Local Customs
    3.1 Language
    3.2 Currency
    3.3 Weather
4. Getting there | Travelling in <topic>
    4.1 By Rail | Rail
    4.2 Airplane | Air
5. Travel Advisory

Figure 1: Excerpt of a sample composite topic tree. Font size and color indicates importance.

Our research is most related to work in topic analysis. We begin with a topical representation of target documents similar to work in topic segmentation, but we differ from many approaches by building a hiearchical structure rather than a list of linear chunks (Choi, 2000; Hearst, 1993). Among hierarchical approaches, our approach utilizes topic headers for analysis rather than a group of keywords (Yaari, 1999). Finally, a unique aspect of the work is the ability to generate composite topic structures from multiple documents. As our research examines documents at the topic level, it differs from much of current work on discourse analysis (Marcu, 1997; Knott, 1996), which focuses on the sentential and clausal levels.

Knowing the topic structure for a set of documents of particular genre and domain can enhance performance of systems in both analysis and synthesis. Indexing topical structures and content can enhance information retrieval applications in search (Hahn, 1990; Belkin et al., 1994). Topical knowledge can also serve as an organization for text summaries (DeJong, 1982; Lin, 1998), or can be used to select sentences for an abstract (Liddy, 1991). For specific domains and genres, this logic is often coded manually by human developers (DeJong, 1982; ARP, 1996). As systems are extended to work in multiple genres and domains, it becomes necessary to have the system learn these topical conventions on its own.

In the next section, we define *text type*, which characterizes the necessary relationship between input documents. We then discuss the representation for topics that make up a document's topical structure. We then present our algorithm for producing the composite topic tree. We conclude with an evaluation of the system on documents from two domains, and a discussion of the possible applications for composite topic trees.

## 1 Text type

Our system operates on documents that belong to the same text type. We define a *text type* to be a set of documents that share the same domain and genre.

- (Genre restriction) Must be intended for the same purpose or communicative act. (Must also be expository in style)
- (Domain restriction) Must be in the same subject area.

This simplified, broad view of text type is well-suited for our system, when we further restrict the genre to also be expository. Biber describes the concept of genre as "the text categories readily distinguished by mature speakers of a language" (Biber, 1989) , which would be distinguished by their location (e.g. in a newspaper) and by format. We differ from Biber's notion of genre, using purpose to define a genre rather than his features of location and format. Hoffman's work gives a clear inventory of features to use to identify text type (Hoffman, 1991). He focuses on a feature set that considers both the linguistic aspect (macrostructure, coherence, syntax, vocabulary and grammatical categories) as well as the communicative aspect (compentence of writer/reader, intention/function, situation, subject matter). For our purposes, the intersection of Hoffman's intention (genre) and subject matter (domain) features define our notion of text type.

Documents which share the same text type have a similar topical structure (Hoffman's macrostructure). There are three aspects to this topical structure: 1) similar topical content, 2) similar ordering between these topics (e.g. attractions before airfare, to entice tourists first), and 3) notion of importance among topics (e.g. local travel will not have airfare). Our algorithm addresses each of these three aspects in constructing the composite topic tree.

## 2 Topic anatomy and representation

Figure 2 shows the section headers for example documents from the text type of foreign travel brochures. The discussion of the *topics* (in bold) of Laos, Cambodia, Thailand and Morocco are faceted into a number of structured *subtopics* (in italics) that form the internal structure. A text type may consist of documents that discuss infinitely many topics, but the subtopic tree structure stays consistent.

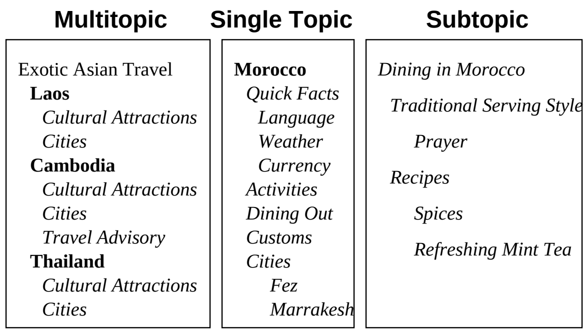


Figure 2: Documents in topic tree format

### 2.1 Document granularity

Physical documents belonging to the same text type may differ in their granularity. Figure 2 shows three different possible granularities. *Multitopic documents* address several topics in one physical document (e.g. "Exotic Asian Travel"). *Single topic documents* are most typical; each document addresses a single topic (e.g. "Morocco"). Some resources that are more comprehensive may divide information into seperate documents, resulting in *subtopic documents*, that provide detail on a specific subtopic (e.g. "Dining in Morocco"). Our algorithm capitalizes on these differences in granularity to conserve computation time and improve accuracy.

### 2.2 Topic data structure

We first need to convert each example document into the appropriate topical representation. To do this, we must identify section headers in a document. For this preprocessing step, we manually identify headers and their nesting level in this work, but we are currently pursuing an automatic approach that harnesses both linguistic and layout information (Anonymous, 2000) to replace this manual step.

Each header in each document is then converted to a (sub)topic node as a slotted data structure. Currently, (sub)topics are represented by the words in its header. This can be expanded to include information from the section body (e.g. important NPs, technical terms, logical propositions) which can be provided by preprocessing tools, such as topic segmentation programs. Other slots in the data structure, such as `level`, `order`, `parents` and `children`, encode the ordering information about the (sub)topic's relative position to others within document. At this initial point, we do not know which nodes are topics and which are subtopics, so all nodes begin with a `generic` type. As the system progresses, nodes are retyped to reflect whether they contain a `topic` or `subtopic` and whether they have been `merged` in the local document or across documents. Figure 2.2 lists the basic fields in the data structure.

Converting all of a document's headers into topic data structures yields an individual document topic tree whose nodes are interconnected by their children and parent pointer fields. The conversion of each of the example text type document to the data structure format thus yields a forest of document topic trees.

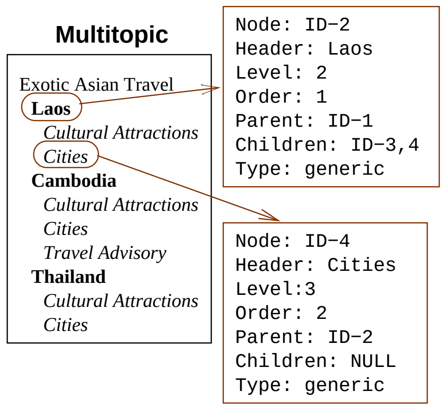


Figure 3: Sample data structure for Figure 2's multitopic document

## 3 Creating the composite topic tree

The composite topic tree algorithm has three main design features that increases its performance over a straightforward design. Instead of describing the algorithm in a linear fashion, we structure this section to highlight these contributions.

### 3.1 Design feature 1: Using relative topic level (RTL) to normalize for different granularities

By recognizing different document granularities in Section 2.1, we also have to process them correctly when merging. An example of this is when subtopic nodes from one document reside on different physical levels across different documents. For example, *Exotic Asian Travel* mentions countries on level 2, whereas another document may just list the countries on level 1. It is important to normalize the levels such that subsequent similarity calculations can understand that these are on the same nesting level conceptually. We do the normalization by introducing a new field in the node's data structure, the *relative topic level* (RTL), which is the node's nesting level relative to the topic level in the document. This value is calculated for all nodes in a document once the topic level is identified. It can also be propagated from a document with known RTL (as in Figure 4, document A) to a new document when nodes belonging to the two documents are merged, as in (document C). We can see that RTL plays a particularly important role in the connection of subtopic documents (such as in document B), when the topic itself is not present in the document. RTL allows these documents to be attached at the appropriate level, normalizing the differences in document granularity.

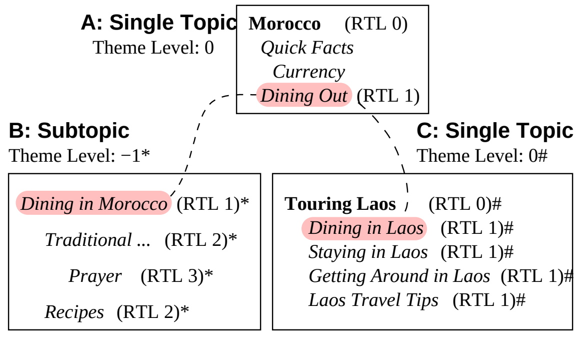


Figure 4: RTL in action. Merging results from A and B marked by *; A and C, by #. RTL propogated from shaded nodes to others in the documents.

### 3.2 Design feature 2: Using a three tiered merging approach

A straightforward approach to constructing the topic tree is to compare the topic data structures across the example documents and merge them if they are similar. The merging process would link the individual document topic trees together and gradually form a single composite topic tree.

We adopt this approach, but modify it by dividing the task into three discrete phases. The initial phase only considers a narrow set of topics that are very likely to be similar. Subsequent phases expand the pool of topics under consideration by lowering its similarity threshold. This modification improves the algorithm by using a tiered approach which considering high quality resources first. Figure 5 illustrates the overall control flow. We will now detail the three phases of the modified algorithm.

#### 3.2.1 Multitopic identification and merging

We begin the topic merging process with multitopic documents. In multitopic documents,

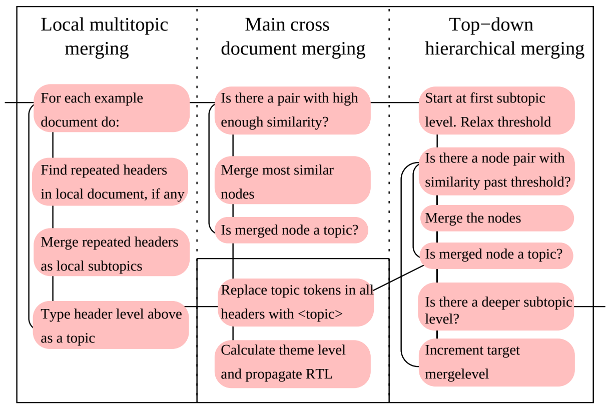

Local multitopic merging
Main cross document merging
Top−down hierarchical merging
For each example document do:
Find repeated headers in local document, if any
Merge repeated headers as local subtopics
Type header level above as a topic
Is there a pair with high enough similarity?
Merge most similar nodes
Is merged node a topic?
Replace topic tokens in all headers with <topic>
Calculate theme level and propagate RTL
Start at first subtopic level. Relax threshold
Is there a node pair with similarity past threshold?
Merge the nodes
Is merged node a topic?
Is there a deeper subtopic level?
Increment target mergelevel

Figure 5: Algorithm architecture

the wording of headers across topics is likely to be identical: a document that describes travel in several countries will most likely repeat the header “Cultural Attractions” for each of the country descriptions, but is unlikely to alternate that header with a variant like “Country Highlights”. We can make this assumption because of Grice’s maxim of manner: “Be clear, avoid obscurity” (Grice, 1975): since the different countries share the same topical structure, the best way to make this clear is to use the same wording for each individual topic.

Repeated identical headers that are subordinate to different parent topics classify a document as a multitopic document. Furthermore, as in Figure 2.2, the nodes with repeated headers must be part of the subtopic tree, not topic nodes. The algorithm begins by examining each document for nodes with identical local headers. Nodes having identical headers are merged (to be discussed in Section 3.3) to form a new node typed as a locally merged subtopic.

Identifying repeated subtopics in a multitopic document also lets us find the parent topics. This is done simply by looking at the level above the least nested subtopics. For example, the repeated subtopic “Cultural Attractions” might have three parent nodes that directly dominate it: Laos, Cambodia and Thailand, all of which are now retyped as locally merged `topic` nodes.

Occasionally, topic names appear in subtopic headers, such as in “Dining in Morocco”, such as in Figure 4, document B. If these subtopic nodes are merged as-is, the merged node will not be general to all topics (e.g. "Dining in Morocco" is viewed as a constituent topic for all topics, including "Laos"), which is clearly incorrect. To correct for this phenomenon, each time a new topic is identified, we replace all instances of it with a generic <topic> string (e.g. "Dining in <topic>"). Later, when creating the topic tree for a specific topic, we can then replace all <topic> tokens with the preferred form of the topic.

### 3.2.2 Main cross document merging

At this point, if multitopic documents were present in our collection, we have most likely identified them, and exhausted the advantages of processing them first. The second phase of the algorithm expands our scope to examine the entire document collection to perform cross document merging. Across documents, related subtopic and topic nodes often may not use the exact same wording nor be placed at the same level or order, so the merging uses a notion of similarity rather searching strictly for identical nodes (again, discussed in Section 3.3).

This phase iteratively merges the two most similar nodes across documents, at any level. All nodes are first placed into a pool and if two are found to be similar enough, a new node representing their merge replaces both of them in the pool. The process starts by merging pairs of nodes highest in similarity and continues until there are no pairs of nodes with similarity above a high minimum threshold. The high threshold is necessary to keep improper merges to a mininum.

Most merging during this stage occurs when the similarity metric has deemed two nodes to be similar enough for merging. As we have expanded the scope of investigation to include all documents, it is possible to capitalize on a special circumstance that may occur in some single topic documents.

**Repeated strings in a series of siblings nodes.** When all children of a parent node use repeated phrases or words as part of their header. An example of this is illustrated in Figure 4, document C. Here, a single parent node has children "Dining in Laos", "Staying in Laos", "Getting Around in Laos", and "Laos Travel Tips", which all share the string "Laos". We can guess two things from this: 1) the shared "Laos" is a topic, and is a variant form of the its parent node ("Touring Laos") and 2) the sibling headers are all first level subtopics and have valid variant forms without the shared string (i.e. "Dining", "Staying", "Getting Around" and "Travel Tips").

### 3.2.3 Hierarchical merging

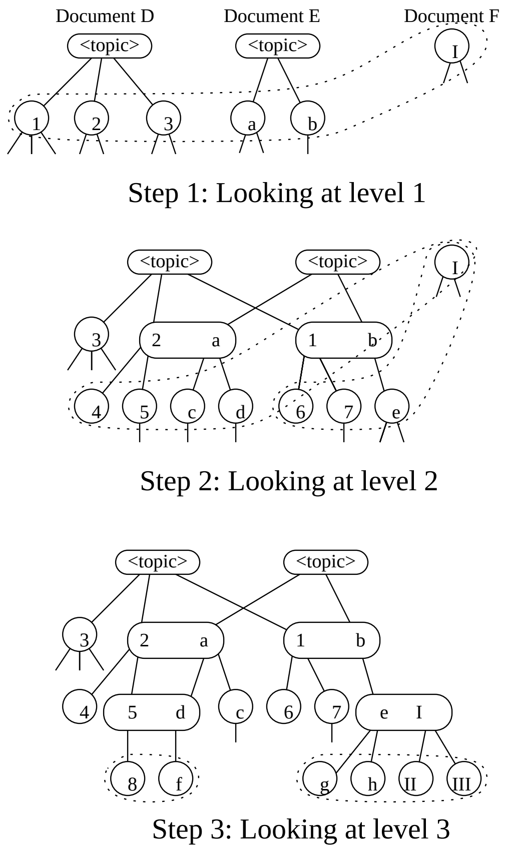


Figure 6: 3 steps in the top-down hierarchical merging process

At this point, we have merged similar topics and subtopics across the different documents and have produced a pool of merged topics. We had set the similarity metric threshold high to limit noise from introducing cascaded errors into the merged nodes. However, at this point there are still many remaining nodes that should be merged: these nodes may express variant headers with few or no words in common. We can merged these nodes if their parent nodes are merged and identical, and if their intralevel ordering is similar. We need to lower the similarity threshold to catch these topics, but we again have to be prudent to limit the candidate target nodes.

We employ top-down hierarchical merging to limit the comparison. This means that we consider the topmost nodes that are most certain first, and proceed recursively to each of the children nodes in a breadth-first traversal, as in Figure 6. At each stage, we limit the pool of candidate topics to be considered to those within $n$ levels of the node's RTL. At this point in the algorithm, if a node still does not have a calculated RTL, its topmost nodes are also considered in the merging, since it may be in a subtopic document whose top level nodes can only now be merged (e.g. document F).

With the breadth-first traversal of the composite topic tree complete, we now have completed all possible merging, and the algorithm is finished. The final result is a collection of merged and unmerged topic and subtopic nodes, which is read off as a tree using the RTL and order information.

### 3.3 Design feature 3: Calculating similarity and merging

At this point, we have explained the algorithm but not how similarity between nodes is calculated. Since we have a structured representation of (sub)topic nodes, we capitalize on these different fields to construct a composite similarity metric. Our similarity calculation uses a single metric which combines header, level (or RTL, when defined for both nodes), ordering and parent node information. This single value facilitates picking a best match when several nodes are somewhat similar to each other. The metric can be invoked with minimum thresholds for each feature to restrict matches (e.g. "Only match if on the same level"). Similarity calculations for numeric fields, such as order and level, are straightforward (Equations 1 & 2). Header string similarity is calculated by first removing stopwords and then computing the maximum ratio of word overlap between all variant forms for each node's headers (Equations 3 & 4).

$$s_{order} = abs(avg(order_{n1}) - avg(order_{n2})) \tag{1}$$

$$s_{lvl} = abs(argmax(lvl_{n1}) - argmax(lvl_{n2})) \tag{2}$$

$$s_{head} = max(s_{string}(n1, n2)) \tag{3}$$

$$s_{string_{h1,h2}} = \frac{overlap(words_{h1}, words_{h2})}{max(|words_{h1}|, |words_{h2}|)} \tag{4}$$

$$s_{parent} = \begin{cases} 1 & \text{if } parent_{n1} = parent_{n2}, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

To merge two nodes, we conduct a pairwise comparison between a source node and all possible candidate nodes. The target with the highest similarity must select the source from its possible merging candidates (in effect, a symmetric constraint on similarity).

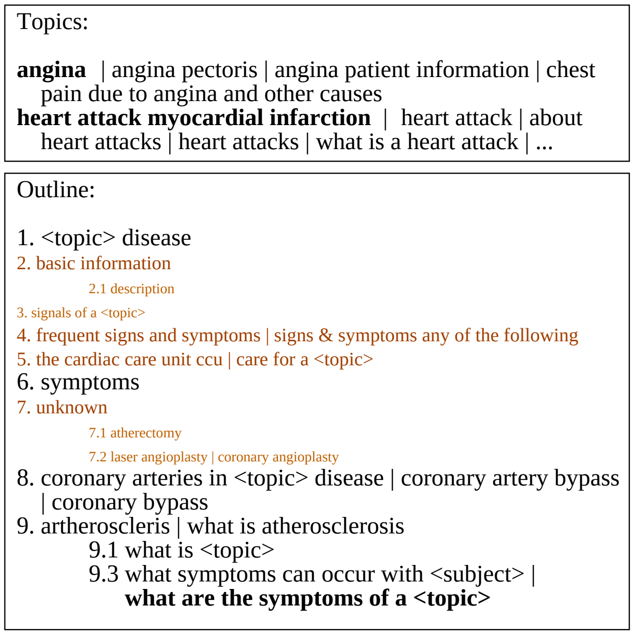
When nodes are merged, the new node stores a composite profile – all variant information for header, level and order fields is saved. For each of the data fields we have mentioned (header, RTL, level, order and parent node), we also store frequency information. The raw frequency information can be used to represent the node's importance (a node resulting from many merges is more important than a unmerged node). Frequency also helps to provide accurate information about the merged node. For example, if the "Cities" subtopic occurs five times after "Attractions" and once before it, we report that it is more likely to appear after. Similarly, if the attractions subtopic is lexicalized sometimes as "Attractions" or as "Cultural Attractions", we report the more commonly used form. In sum, frequency information allows us to use the average or most frequent value of the data slot when it makes sense. This is reflected in the similarity equations, in their use of *argmax* and *avg*.

## 4 Evaluation setup

We have implemented the entire algorithm and have evaluated its performance and tested its robustness across domains. Since there is no corpora containing documents of specific text types nor accurate programs for detecting text types, we needed to construct our own text type corpus. We chose to test the system on consumer travel and patient health information on heart diseases. To gather the text type corpora, we first chose the appropriate categories in three website directories (Yahoo!, About.com, and the Open Directory Project) and downloaded all documents on Laos and Morocco (for travel) and Angina and Heart Attacks (for heart diseases). Since the directories categorize by subject, these documents fulfill the domain restriction but not necessarily the genre restriction (i.e. the list includes content pages, but also browsing pages, advertising, etc.). We asked 5 volunteers to pick out the documents that contained an expository discussion of the topic to manually simulate our genre restriction. The task was reported to be difficult but reached a moderate level of agreement ($\kappa = .49$, for travel and $\kappa = .48$ for patient information, $p << 0.001$ for both). We chose all documents that appeared on a majority of the subject's lists for our corpus (120 travel documents, 72 patient information ones), splitting it into equal halves for algorithm development and testing. After finishing the implementation, we produced the output for the two text types on the testing half of the corpus.

## 4.1 Results

We asked two reference librarians who specialize in health sciences to subjectively assess the quality of both the travel (as non experts) and the patient information (as experts) topic trees. To do this, we converted the trees into an outline format where the level and order information are preserved and the relative importance (i.e. the frequency of the subtopic) was indicated by font color and size. The evaluation asked them to evaluate the topic trees in the three areas of topical structure mentioned in Section 1: *content* (Should the items on the outline be there? Are any items missing?), *ordering* (Are there items that should be promoted or demoted a level in the outline?) and *importance* (Are the larger font items ones that all documents of this type should have?).



Topics:

**angina** | angina pectoris | angina patient information | chest pain due to angina and other causes
**heart attack myocardial infarction** | heart attack | about heart attacks | heart attacks | what is a heart attack | ...

Outline:

1. <topic> disease
2. basic information
    2.1 description
3. signals of a <topic>
4. frequent signs and symptoms | signs & symptoms any of the following
5. the cardiac care unit ccu | care for a <topic>
6. symptoms
7. unknown
    7.1 atherectomy
    7.2 laser angioplasty | coronary angioplasty
8. coronary arteries in <topic> disease | coronary artery bypass | coronary bypass
9. artheroscleris | what is atherosclerosis
    9.1 what is <topic>
    9.3 what symptoms can occur with <subject> | **what are the symptoms of a <topic>**

Figure 7: Excerpt of the evaluation outline for patient information on heart diseases

The librarians reported satisfaction with variants that the algorithm merged, but occasionally reported (4 mistakes of a total 40) that semantically similar headers were not merged when they should have been. We interpret this as an indication that we can relax the similarity threshold to allow addition merging. In the consumer travel text type, "cultural attractions" and "shopping" subtopics were conspiciously missing (2 of 8 content errors). Analysis of the corpus revealed that these problems were artifacts of the testing documents: cultural attractions were directly named in the headers and shopping is not primary attraction in either country. Minor problems with grammaticality (3 other content errors) (e.g. "What causes a <topic>" works when combined with "Heart Attack" but not "Angina") were the most prominent problems in this area. Introducing shallow parsing that would identify dependent articles and prepositions could help here.

Node ordering within the outline comprised the bulk of the problems (10 misorderings, 13 misnestings). The librarians agreed that specific examples should be relegated to the end of the outline and that primary information should be moved to the front of the outline. Analysis revealed that subtopic documents that were merged incorrectly as single topic documents sometimes caused this problem. Additional restrictions on the final hiearchical merging phase may help here.

Librarians were satisfied with node importance as judged by the system (5 errors). They were also satisfied with the default header that the system chose (0 errors).

Overall, the librarians both concurred that the system performed better on the patient information text type (13 errors total) than on consumer travel (27 errors total). We believe this to be caused by the fact that the consumer travel corpus included both tourist travel as well as business and investing travel. The outlines comprised of 35 merged topics and subtopics for patient information, and 95 for the consumer travel, averaging one error every 2.7 headers in the former, 3.5 in the latter. Some headers produced two or more errors, thus the header to error ratio is higher. Based on this evaluation, we believe that the algorithm performs satifactory but can be improved, especially in the area of ordering and nesting.

## 5 Conclusion

Composite topic trees can be used by a wide range of applications. The tree is a knowledge representation of a text type; a script that text generation can follow for structuring content. The topic tree encodes a descriptive relation between topics and their subtopic tree that can be used to augment a lexicon, complementing work done on the hyponym and meronym relationships (Hearst, 1992; Berland and Charniak, 1999). Text categorization can utilitize topic trees as a richer source of class data for classifying unseen documents. We are pursuing the use of the topic tree in text summarization in future work. Documents can be compared against their text type's tree to classify its topics into ones that are generic and ones that contain unique information that should be reported.

In this paper, we have shown a new method for generalizing topic structures across related documents. We have defined the nature of the relatedness between documents that gives rise to similar topical structure – the notion of a text type, defined by the intersection of domain and genre.

We make three contributions to the state of the art in topic analysis in the architecture of our merging algorithm. First, we introduced relative topic level (RTL), enabling cross document merging for documents of differing granularity. Second, our tiered approach uses higher quality sources first to reduce error and computation. Third, our header-centric topic representation consists of structured fields, which allow fine grained control when calculating topic similarity. Finally, in evaluating our algorithm, we have additionally defined guidelines for collecting text type corpora.

## References

Anonymous. 2000. Combining visual layout and lexical cohesion features for text segmentation. Technical report, Anonymous.

ARPA, Advanced Research Projects Agency. 1996. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Francisco, California. Morgan Kaufmann.

Nicolas Belkin, Colleen Cool, Adelheit Stein, and Ulrich Thiel. 1994. Cases, scripts, and information-seeking strategies: On the design of interactive information retrieval systems.

Matthew Berland and Eugene Charniak. 1999. Finding parts in very large corpora. In *Proceedings of the 37th ACL.*

Douglas Biber. 1989. A typology of English texts. *Linguistics*, 27:3–43.

Freddy Choi. 2000. Advances in domain independent linear text segmentation. In *Proceedings of NAACL'00*, Seattle, USA.

Gerald DeJong. 1982. An overview of the FRUMP system. In Wendy Lehnert and Martin Ringle, editors, *Strategies for Natural Language Processing*, pages 149–176. Erlbaum, Hillsdale.

H. P. Grice. 1975. Logic and conversation. In P. Cole and J. Morgan, editors, *Syntax and Se-*

*mantics III: Speech Acts*, pages 41–58. Academic Press.

Udo Hahn. 1990. Topic parsing: Accounting for text macro structures in full-text analysis. *Information Processing & Management*, 26(1):135–170.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING-92*, pages 539–545, Nantes, France, August.

Marti Hearst. 1993. Text tiling: A quantitative approach to discourse segmentation. Technical report, University of California, Berkeley, Sequoia.

Lothar Hoffman. 1991. Texts and text types in LSP. In Hartmut Schröder, editor, *Subject-oriented Texts: Languages for Special Purposes and Text Theory*, Research in Text Theory, pages 158–66. Walter de Gruyter, Berlin.

Alistair Knott. 1996. *A Data-driven Methodology for Motivating a Set of Coherence Relations.* Ph.D. thesis, University of Edinburgh.

Elizabeth Liddy. 1991. The discourse-level structure of empirical abstracts: An exploratory study. *Information Proc. & Management*, 27(1):55–81.

Chin-Yew Lin. 1998. Assembly of topic extraction modules in summarist. In *Proceedings of the AAAI Spring Symposium on Intelligent Text Summarization.*

Daniel Marcu. 1997. The rhetorical parsing of natural language texts. In *Proceedings of 35th ACL and 8th EACL*, pages 96–103, Madrid, Spain.

Yaakov Yaari. 1999. *The Texplorer.* Ph.D. thesis, Bar Ilan University, Israel, April.