

For more details and enhancements please refer to the journal papers listed on <http://www.rfreeman.net/>

# Topological Tree Clustering of Web Search Results

Dr Richard T. Freeman

Capgemini, Business Information Management  
richard.freeman@capgemini.com  
<http://www.rfreeman.net>

**Abstract.** In the knowledge economy taxonomy generation, information retrieval and portals in intelligent enterprises need to be dynamically adaptive to changes in their enterprise content. To remain competitive and efficient, this has to be done without exclusively relying on knowledge workers to update taxonomies or manually label documents. This paper briefly reviews existing visualisation methods used in presenting search results retrieved from a web search engine. A method, termed topological tree, that could be used to automatically organise large sets of documents retrieved from any type of search, is presented. The retrieved results, organised using an online version of the topological tree method, are compared to the visual representation of a web search engine that uses a document clustering algorithm. A discussion is made on the criteria of representing hierarchical relationships, having visual scalability, presenting underlying topics extracted from the document set, and providing a clear view of the connections between topics. The topological tree has been found to be a superior representation in all cases and well suited for organising web content.

**Keywords:** Information retrieval, document clustering, search engine, self organizing maps, topological tree, information access, faceted classification, guided navigation, taxonomy generation, neural networks, post retrieval clustering, taxonomy generation, enterprise portals, enterprise content management, enterprise search, information management.

## 1 Introduction

The rapidly growing volume of electronic content is leading to an information overload. On the Internet, the use of web search engines is critical to finding and retrieving relevant content. Despite the numerous advances in information visualisation [1], the most popular way of presenting search results still remain ranked lists. In this format, the user generally never looks beyond the first three pages, after which they will rather lengthen their search query by adding more terms or refine the initial query [2]. Although ranking mechanisms help order the web pages in terms of their relevance to the users query (e.g. Google<sup>1</sup>), they do not provide any guide as to the overall themes described in the web pages or their relationships. Some efforts have been made to provide different visual representation of the search results, such as suggesting keywords to refine the search (e.g. Webcrawler<sup>2</sup>), representing a graph

---

<sup>1</sup> <http://www.google.com/>

<sup>2</sup> <http://www.webcrawler.com/>

For more details and enhancements please refer to the journal papers listed on <http://www.rfreeman.net/>

view of the relations between pages (e.g. Kartoo<sup>3</sup>) or clustering the results (Vivisimo<sup>4</sup>). A major review of the methods and algorithms can be found in [3] [4].

This paper deals with methods that *organise documents* (retrieved by a web search engine) into *automatically extracted* topics. A method which clusters web pages dynamically, whilst creating a topology between them in a tree view, is presented in this paper. The *topological tree* method, first introduced by the author [3], is enhanced through weighting terms depending on their relation to the query term and making the algorithm function efficiently with dynamic datasets. Results and discussions confirm that the topological tree representation can be used to provide a user with a more intuitive and natural representation for browsing documents and discovering their underlying topics.

## 2 Visual Representation of Retrieved Content

### 2.1 The Importance of Clustering and Topology

In information access systems, the major visual representations are Self-Organising Maps (SOMs), binary or  $n$ -way trees, graphs, and ranked lists. In some cases a combination of these representations can be used. This section describes the limitations of these methods, and illustrates the benefits of using the topological tree structure.

Clustering algorithms can be used to sort content into categories which are discovered automatically based on a similarity criterion. Its typical output representation is a binary tree or generic  $n$ -way tree.  $n$  being the number of nodes at each level, value which can be fixed or dynamic at each level in the tree. Binary trees quickly become too deep as each level only has two nodes; this representation has been used for retrieval rather than browsing.  $n$ -way trees are typically generated using partitioning algorithms (e.g.  $k$ -means), or can be manually constructed such as with social bookmarks (e.g. Del.icio.us<sup>5</sup>) and web directories (e.g. Dmoz<sup>6</sup>). Web directories are particularly beneficial to users who are not familiar with the topics and their relations. However, even if some show cross links with related topics, they do not show the relations between topics at the same level, rather the topics are sorted alphabetically or by popularity. Other search engines such as Vivisimo do cluster results, however at each level in the tree there is always a category "other topics" where many document are clustered to. In addition, as with the other  $n$ -way trees, there is no relationship between the topics at each level.

Another important trend in industry is the taxonomy generator packages, e.g. Autonomy / Verity Thematic Mapping [5]. These allow the construction of topic hierarchies that can be used for browsing or classification (matching a new document to existing topics). However they are out of the scope of this document, as they are

---

<sup>3</sup> <http://www.kartoo.com/>

<sup>4</sup> <http://www.vivisimo.com/>

<sup>5</sup> <http://del.icio.us/>

<sup>6</sup> <http://www.dmoz.org/>

For more details and enhancements please refer to the journal papers listed on <http://www.rfreeman.net/>

generally constructed offline and / or manually by subject matter experts. As with other tree representations, these taxonomies rarely represent the relationships between the topics at any one level, i.e. only the hierarchical relations and limited cross links are shown.

Graph representations or SOMs can be used to compensate for this lack of topology in these tree representations or taxonomies. Graphs can represent hyperlinks, relationships or links between topics. A web example of a graph generated representation is Kartoo. Other knowledge representations such as Topic Maps (e.g. Omnigator<sup>7</sup>), can also be represented as graph structures. Although they do capture the inter topic / document relations, the major drawback is that they cannot scale easily, i.e. the more nodes / links are added the less legible it becomes. SOMs typically have a 2-dimensional grid structure which adapts to the content space and the number of nodes need not change to represent the underlying number of topics. The SOM-based methods have two distinct properties over other methods, namely non-linear dimensionality reduction and topology preservation. The non-linear projection property ensures that the input space is mapped onto a lower dimensional space with minimum information distortion. The topology preserving clustering enables documents that are similar to be located closely on the map. However one the major weakness of 2-dimensional SOMs, is it is difficult to navigate between different levels of detail. Hierarchical variants of the SOM, such as the Growing Hierarchical SOM [6] have been developed for this purpose; however only one map can be shown at any time and their size is sensitive to fixed parameters. In addition, tables or complex graphics are required to represent the 2-dimensional maps efficiently.

The topological tree method, first proposed by the author [3], compensates for all these factors by exploiting a simple tree view structure to represent both *hierarchical* and *topological relationships* between topics. Previous work undertaken by the author focused on clustering a fixed set of documents. This paper deals with the clustering of search results of multi author / non-uniform documents with different formatting and content. The topological tree can be used to combine the tree structure with that of the topology inherent in SOMs. The tree structure allows a user to visualise different levels of detail and hierarchical relationships. The topology, a novel feature specific to the topological trees and SOMs, additionally allows the viewing of the relationships between the topics. Fig. 1 clearly shows the difference between having a topology and not having one. On the left, the topics appear to be randomly placed, but on the right they naturally flow downward as economics, microeconomics, finance, biology, and anatomy making it more intuitive and natural to the user.

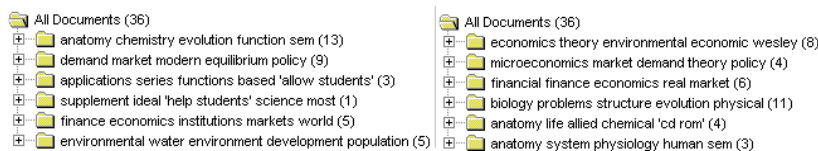


Fig. 1 –  $k$ -mean with no topology (left) and root level in the topological tree (right)

<sup>7</sup> <http://www.ontopia.net/omnigator/models/index.jsp>

### **3 The Topological Tree Method**

#### **3.1 Overview of the Method**

There are a number of essential steps in the method:

1. The user enters a query term into the local web application, and selects the search options and search engine.
2. The application submits the query term to the search engine and crawls the returned results.
3. Each page is indexed and transformed into a document vector.
4. Feature selection and term weighting is performed on the vector.
5. The documents are organised in a growing chain (see section 3.3).
6. Each chain is labelled and added to the topological tree, if further child chains are required (see section 3.3) return to 4.
7. The user is presented with the resulting generated topological tree.

#### **3.2 Text Pre-Processing**

Text pre-processing is essential to any search or retrieval system, since the quality of the terms will have an impact on the results. There are generally three steps, the indexing, feature selection and term weighting. In the first step of indexing, the HTML is parsed and extracted terms are transformed into vector forms to allow fast mathematical comparisons. The second step involves selecting the most relevant terms. This feature selection is required to reduce the number of terms and select the most discriminative terms. The terms that are not frequent or too frequent can be discarded, as they do not help find common patterns in the document set.

In the third step, the remaining terms are weighted to give more mathematical importance to potentially more significant terms. The keyword query term, as well as their context, can be considered more relevant to the search; hence these are weighted more heavily in the document vectors. In essence the terms in the web metadata, title, search engine snippet (distinct excerpt from retrieval results), and context of the query are all weighted more heavily since these are likely to be most significant. In the growing chain, these weighted document vectors are used to compute similarities to a node's weight vector via a dot product.

#### **3.3 Growing Chains and Topological Tree Method**

SOMs are generally associated with 2-dimensional structures that help visualise clusters and their relationships in a topology. However, equally 1-dimensional chains can also be used. The topological tree method uses 1-dimensional chains where each node may spawn a child chain. The number of nodes in each chain is guided by an independent validation criterion. The algorithm used to grow the 1-dimensional SOM is termed growing chain (GC) and shares growing properties with the growing grid (used in the GH-SOM [6]) and growing SOM variants, but is more suited for 1-dimension.

As with the SOM, there are two major steps in the GC algorithm: the search for the best matching unit and the update of the winner and its neighbouring nodes. At time  $t$ ,

For more details and enhancements please refer to the journal papers listed on <http://www.rfreeman.net/>

an input document vector  $\mathbf{x}$  is mapped to a chain consisting of  $n$  nodes with a weight vector  $\mathbf{w}$ . The best matching unit  $c(\mathbf{x})$  is the node with the maximum dot product amongst nodes  $j$  and document vector  $\mathbf{x}(t)$ ,

$$c(\mathbf{x}) = \arg \max_j \{S_{dot}(\mathbf{x}(t), \mathbf{w}_j)\}, \quad j = 1, 2, \dots, n \quad (1)$$

where  $n$  is the current number of nodes. Once the winner node  $c(\mathbf{x})$  is found the neighbouring weights are updated using,

$$\mathbf{w}_j(t+1) = \frac{\mathbf{w}_j(t) + \alpha(t)h_{j,c(\mathbf{x})}(t)\mathbf{x}(t)}{\|\mathbf{w}_j(t) + \alpha(t)h_{j,c(\mathbf{x})}(t)\mathbf{x}(t)\|} \quad (2)$$

where  $\alpha(t)$  is the monotonically decreasing learning rate and  $h_{j,c(\mathbf{x})}(t)$  the neighbourhood function, typically a Gaussian kernel. When the learning has stabilised for the current number of nodes  $n$ , the entropy of the chain is recorded and a new node is inserted next to the node with the highest number of wins. The weights of the new node are initialised by interpolating or extrapolating existing nodes weight values. New nodes are added until  $n_{max}$  nodes are reached which corresponds to the maximum allowable chain size. Finally the validation criterion, the entropy-based Bayesian Information Criterion that penalises complexity, gives the optimum number of nodes per chain as:

$$\tau = \arg \min_n \left\{ \frac{1}{m} \sum_{j=1}^n m_j \cdot H(C_j) + \frac{1}{2} n \log m \right\}, n = 2, \dots, n_{max} \quad (3)$$

where  $m$  is the number of documents,  $n$  the current number of nodes in the chain,  $H(C_j)$  is the total normalised and weighted sum of entropies for cluster  $C_j$ .

Then in the hierarchical expansion process, each node in the chain is tested to determine if it will spawn a child chain. This is performed using several tests. The first test counts the number of document clustered to that node to see if it is less than a fixed threshold. The next test analyses the vocabulary present in those documents to determine if there is a sufficient number of terms. The final test uses cluster tendency method. It aims to test if a set of documents contains random documents with no or few relations or if there are strong underlying clusters [7]. If any of these tests fail for a particular node, then it does not spawn a child chain and becomes a leaf node in the final topological tree representation.

Finally each node in the chain is labelled using the most representative terms of the node's weight and its frequency. Once the chain is labelled, then it is added to the current topological tree structure. If further hierarchical expansions in its child chains are required, then the process is repeated for each of the child chains, otherwise the process is terminated and the results presented to the user. The full pre-processing and topological tree method is shown in Fig. 2.

For more details and enhancements please refer to the journal papers listed on <http://www.rfreeman.net/>

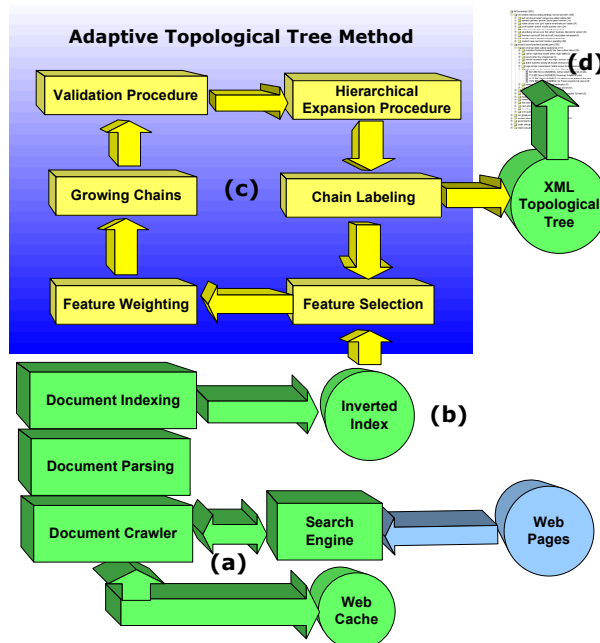


Fig. 2. **The Topological Tree Method.** (a) The search engine is queried, the pages are returned and crawled by the Web Application. (b) An inverted index is generated from the retrieved documents. (c) The closed loop represents the necessary processing for each growing chain in the topological tree. It is grown using an independent validation procedure that estimates the optimum number of nodes that maximise the information value. (d) Once the topological tree is complete it is exported to XML.

## 4 Results and Discussions

The dataset was dynamically generated from a search query. The query was “cookie”; other queries were also tested but omitted for space considerations. The Vivisimo tree, shown in Fig. 3, was generated by directly submitting the same query to the search engine and taking a sample snapshot of the tree. Fig. 4 shows the topological tree that was generated from running a Google query and crawling the returned ranked listing.

### 4.1 Comparison

Although Vivisimo uses meaningful pre-crafted labels compared to the topological tree, it suffers from the fact that the number of categories tends to grow large at root level and this number seems arbitrary. In addition the relations between topics at each level are ambiguous (only the hierarchical relations are represented) and many documents remain unclassified as “other topics”. In comparison, the topological tree representation appears more intuitive and natural to the user, as closely related topics are located close to one another in each chain. Each chain does not grow to a large number of nodes, as this number is guided by an independent validation criterion that penalises complexity. In addition hierarchical relations between a parent node and child chain help abstract different levels of detail.

For more details and enhancements please refer to the journal papers listed on <http://www.rfreeman.net/>



Fig. 3 – A partial snapshot of a tree generated using Vivisimo on the query “cookies”. Clearly the tree becomes confusing as the web and edible cookies ordering is intermingled at the same level in the tree, making it less understandable.

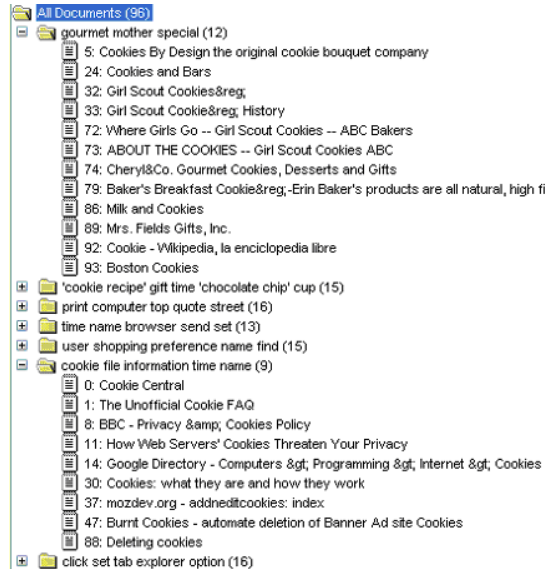


Fig. 4 – A topological tree generated from the pages retrieved using the query “cookies”. Clearly the topology ensures that the web cookies and edible cookies are separated. From top to bottom the main theme of edible cookies, cookie recipes, cookies sales, web cookies in browsers, and data stored in cookies can be observed making it visually intuitive.

## 4.2 Discussion

There are four important criteria for creating an effective browsing experience of documents and topics:

1. **Hierarchical Representation:** the topics need to show different levels of detail simultaneously. This is especially true when the number of topics is large, e.g. the Dewey decimal classification or web directories.
2. **Scalability:** the ability to view a large number of topics and documents in the same window.
3. **Visualise key topics and their related documents:** key topics should be easily be discernable using a label and documents should be shown to belong to one of more of them.
4. **Visualise key relationships:** the ability to visualise the relationships between different topics as well as the connections between documents.

Table 1 compares the differences between visualisation methods based on these four criteria. It shows that only the topological tree meets all hierarchical, scalability, major topics and topology visual criterions required for efficient browsing.

For more details and enhancements please refer to the journal papers listed on <http://www.rfreeman.net/>

**Table 1.** Comparison of the differing automatically generated representations for organising documents.

	1. Hierarchical Representation	2. Scalability	3. Visualise key topics	4. View documents / topics connections
SOM	N	Y	Y	Y
GH-SOM	N <sup>1</sup>	Y	Y	Y
Binary tree	Y <sup>2</sup>	Y <sup>2</sup>	N	N
<i>n</i> -way tree	Y	Y	Y	N
Graph View	N	N	Y	Y
Ranked List	N	N	N	N
Topological Tree	Y	Y	Y	Y

<sup>1</sup> only one map can be shown at any one time.

<sup>2</sup> not efficiently as there are two nodes per level leading to a deep structure.

## 5 Conclusion and Future Work

A topological tree is a tree view structure that does not require complex 2-dimensional graphics or tables such as used in SOMs or graphs. Yet it can complement current faceted classification solutions (e.g. guided navigation used in Endeca), by showing the key relationships between extracted topics thus helping reveal previously unknown associations automatically. It also helps make a tree structure appear more intuitive, i.e. related topics are located close to one another in the tree. This topology can be thought of as a graph representation that has been optimised into a tree view, where only the strongest relationships between topics are preserved. Through building on top of existing search engines, the topological tree method benefits from pre-filtered content where it only needs to organise a relevant subset of the content. This paper has shown that the topological tree can be built on top of a typical web search engine and produce an insightful overview of the underlying topics contained in the top ranking web pages. Future work could look at extracting and combining knowledge from web directories and social networks, with results returned from a web search engine, into a topological tree.

## References

- [1] Herman, I., Melancon, G., and Marshall, M., Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*. **6**(1). 24-43, 2000.
- [2] Search Engine User Behavior Study, White Paper, iProspect, April 2006.
- [3] Freeman, R.T. and Yin, H., Adaptive Topological Tree Structure for Document Organisation and Visualisation. *Neural Networks*. **17**(8-9). 1255-1271, 2004.
- [4] Freeman, R.T., Web Document Search, Organisation and Exploration Using Self-Organising Neural Networks, PhD Thesis, Faculty of Engineering and Physical Sciences, School of Electrical & Electronic Engineering, University of Manchester: Manchester, 2004.
- [5] Chung, C.Y., et al. Thematic Mapping – From Unstructured Documents to Taxonomies. in *Proceedings of the 11th International Conference on Information and Knowledge Management VA*, 2002.
- [6] Rauber, A., Merkl, D., and Dittenbach, M., The Growing Hierarchical Self-Organizing Map: Exploratory Analysis of High-Dimensional Data. *IEEE Transactions on Neural Networks*. **13**(6). 1331 -1341, 2002.
- [7] Freeman, R.T. and Yin, H., Web Content Management by Self-Organization. *IEEE Transactions on Neural Networks*. **16**(5). 1256-1268, 2005.