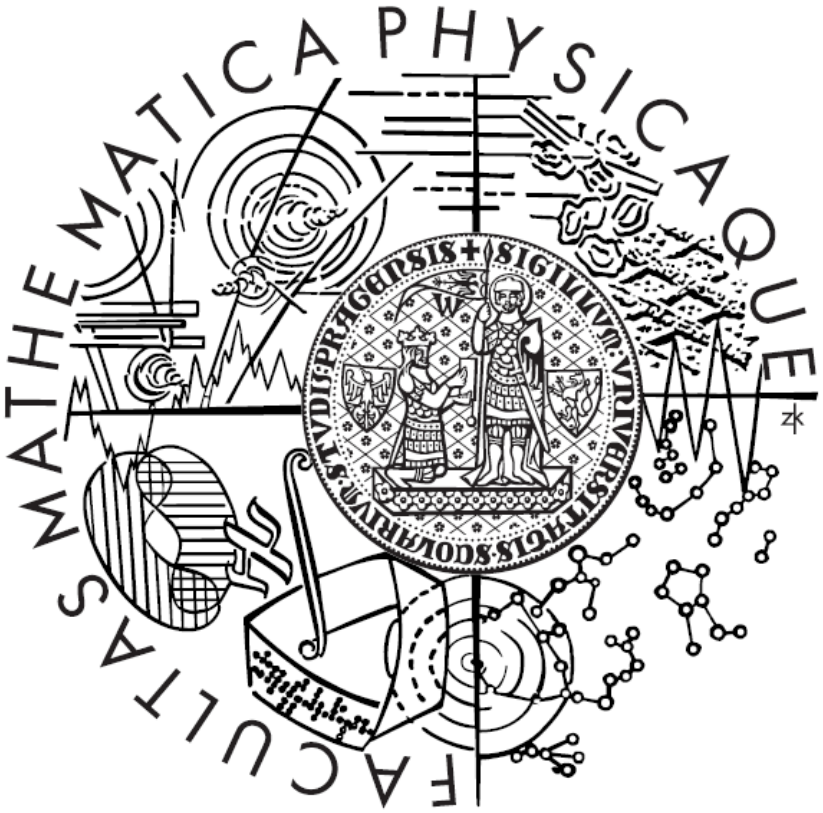
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Petr Jenček

Využití vizualizačních technologií pro vyhledávání dokumentů

Katedra softwarového inženýrství
Vedoucí diplomové práce: Prof. RNDr. Peter Vojtáš, DrSc.
Studijní program: Informatika, softwarové systémy

2009

Rád bych zde poděkoval mému vedoucímu Prof. RNDr. Peteru Vojtášovi, DrSc. za cenné rady, připomínky a vedení této práce a také Bc. Cyrilu Höschlovi, jehož znalosti metody Sociomapování® byly pro tuto práci velkým přínosem.

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze dne 9. dubna 2009 Petr Jenček

Contents

Název práce: Využití vizualizačních technologií pro vyhledávání v dokumentech
Autor: Petr Jenček
Katedra: Katedra softwarového inženýrství
Vedoucí diplomové práce: Prof. RNDr. Peter Vojtáš, DrSc.
E-mail vedoucího: vojtas@ksi.mff.cuni.cz

Abstrakt: Cílem této práce je prostudovat možnosti prezentace výsledků vyhledávání dokumentů dle uživatelem zadaných klíčových slov pomocí metody sociomapování. Zaměříme se jak na vyhledávání na webu pomocí několika vyhledávacích služeb, tak na vyhledávání dokumentů uložených lokálně. Z takto získaných výsledků vytvoříme shluky dokumentů pomocí některého ze studovaných shlukovacích algoritmů za použití některé ze studovaných metrik podobnosti klíčových slov. Takto získané shluky poslouží jako vstupní data pro vytvoření přímé, nebo nepřímé sociomapy. V závěru porovnáme výsledky jednotlivých shlukovacích algoritmů z několika hledisek.

Title: Usage of Visualization Techniques for Document Retrieval
Author: Petr Jenček
Department: Department of Software Engineering
Supervisor: Prof. RNDr. Peter Vojtáš, DrSc.
Supervisor's e-mail address: vojtas@ksi.mff.cuni.cz

Abstract: The aim of this work is to study possibilities of presentation of document search results according to keywords given by a user using sociomapping method. Both web search and locally stored documents search will be our interest. We'll create document clusters from such results using one of the clustering algorithms and measures of keywords similarities. These clusters will be input data for creation of direct, or indirect sociomap. Finally we'll compare results of the clustering algorithms from several points of view.

1 Introduction

1.1 Work Overview

In present human meets large amount of data every day. Most of these data are text data (data which may be translated into plain text without losing much of its information). These data are for example web sites, e-mail messages or books, magazines and other monographs in a library (let's call them documents). Results of search in these documents are aside from few exceptions limited to a text list sorted according to relevance in relation with the search query (e.g. "obama politics", "earthquake italy" or "personal income tax"). User goes through first several items and reformulates the search query in case he doesn't find what he is looking for. After several search query reformulatings the user gives up stating that the document space doesn't contain the requested document. Relevance difference among the first several documents is quite low and therefore it is possible that the requested document isn't on the first page of search results but it is e.g. on 6th page which the user usually doesn't reach. It is necessary for the user to get an overview of the search results with at least minimal relevance to the search query. Since a user is able to understand graphical information much more than the textual one a suitable graphical representation of the search results would be very helpful for the user. The aim of this work is to provide a way to construct such graphical representation.

1.2 Work Organization

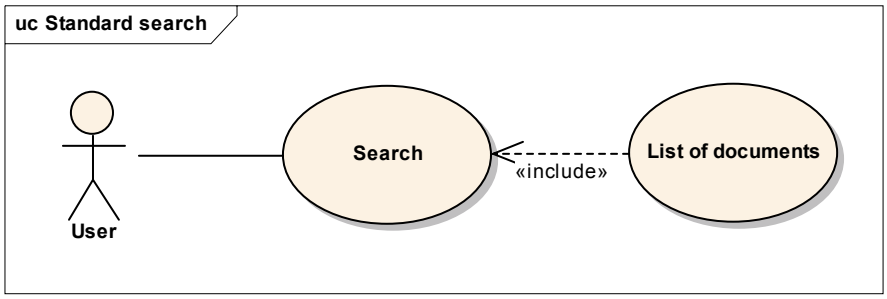
In chapter Current Search Results Presentation Methods current ways of presenting search results provided by web service are briefly introduced. Description of Sociomapping®[1] and its application on presenting document search results is given in Sociomapping Method Description chapter. Software tool written for the purpose of testing of this method is in-depth described in chapter Analysis. Search visualization results are presented in Experiments chapter. Last chapter, Search Visualization Tool User's Manual, contains instructions on how to use the software tool mentioned in Analysis.

---

[1] Sociomapping® is a method protected by patent. Text of this patent may be found at http://www.freshpatents.com under patent number 20050256813. Sociomap® and Sociomapping® are registered trade marks. This fact won't be mentioned in further text and these words will be written with lower case letters and without trade mark character.

2 Current Search Results Presentation Methods

Typical search engine works in a very simple way as we may in Picture 1.

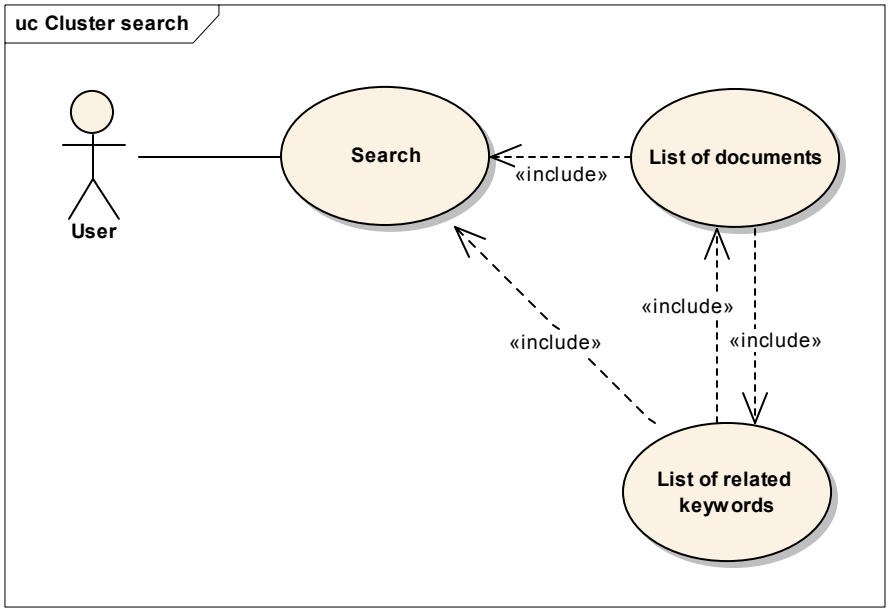


Picture 1

Since List of documents in Picture 1 is usually sorted according according to relevance of query string (this notion will be explained further, consider it as a string which user writes as a query to the search engine) usually the document we are looking for is at the beginning of the list (on the first page if the result is paged). However the user sometimes doesn't know what he's looking for exactly (in cases when he wants to find out information about too unspecific topic, e.g. global warming or economic crisis). Other difficulties of document searching might occur when a user doesn't specify the search string in an optimal way (e.g. "tube" meaning underground in London).

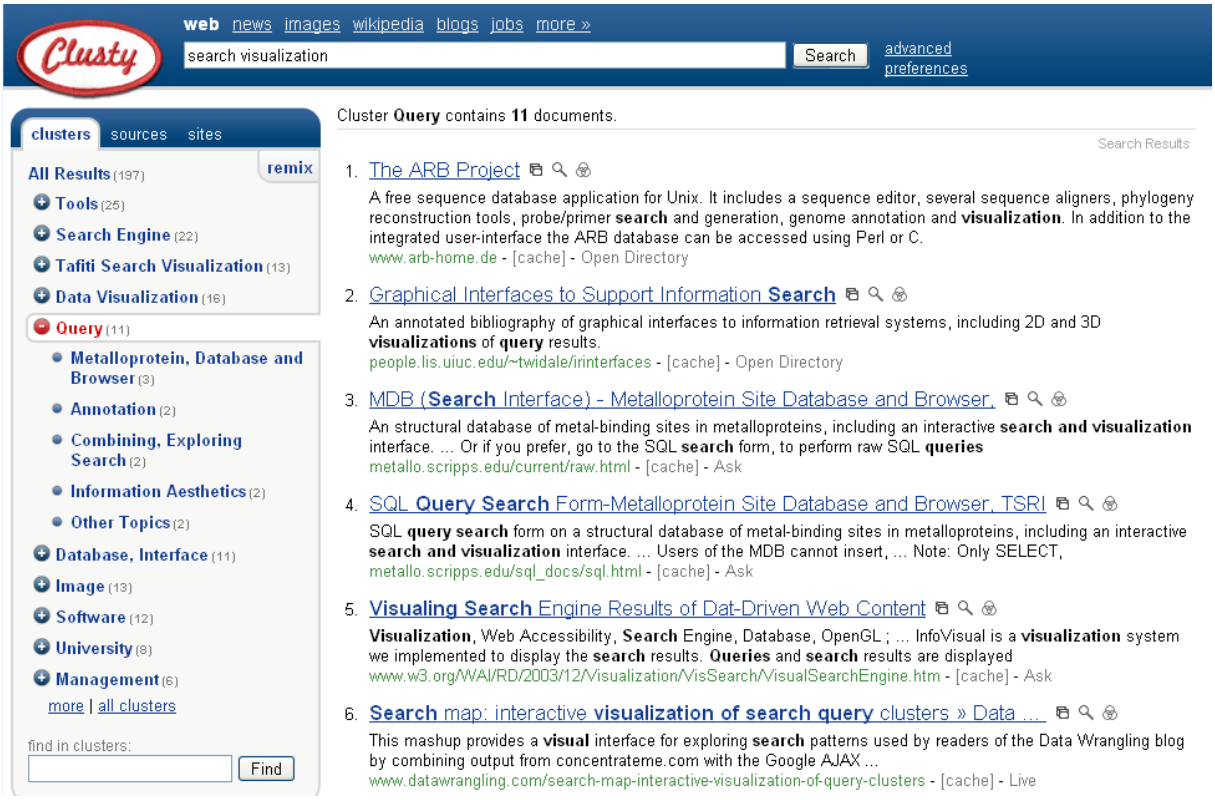
This situation is quite similar to a situation when a customer in a shopping mall, wants to buy shoes and therefore goes to the information desk and asks where he/she can buy shoes. Usually the response provided by the information staff is a question on which kind of shoes does the customer need – casual, sport, style etc. By a serie of such questions and answers the information staff finally tells few specific shops which the customer should visit.

Let's call this search strategy cluster search strategy. Cluster, because the items among which the user searches are presented as clusters with a representative value. In this case the items are documents and representative values are keywords with high relevancy of the documents within the cluster. Cluster search strategy applied on standard document search use case (Picture 1) gives cluster search use case (Picture 2).



Picture 2

2.1 Clusty



Clusty is a web search engine which uses cluster strategy. On a given search string it provides both list of documents sorted by relevancy to the search string (on the right side) and names of the clusters (on the left side). The list of the documents may be restricted by clicking on the desired cluster name. Clusty works at http://clusty.com.

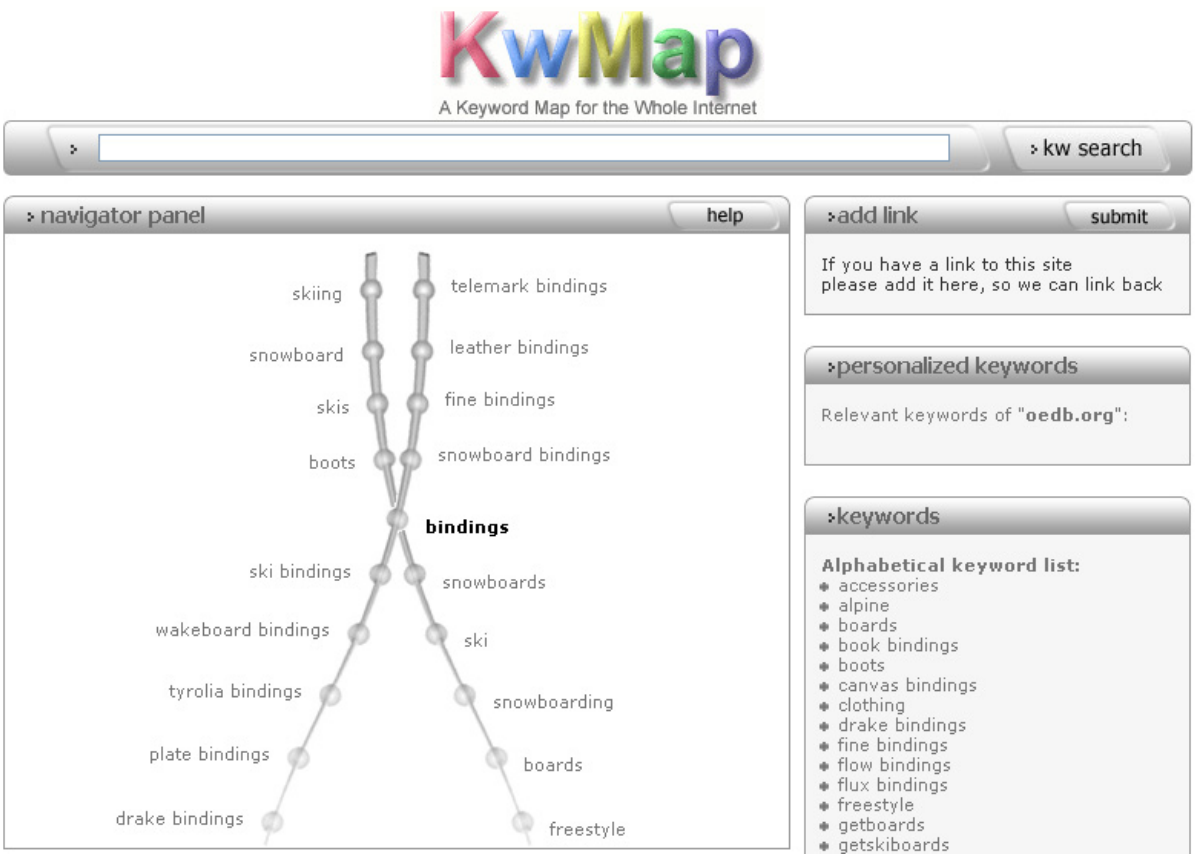
2.2 Kartoo

Web search engine Kartoo uses the clustering strategy as well. Additionally it displays documents (websites) and keywords in 2D space. Keywords are connected to documents which are the most relevant for them.

Using this method it is not possible to display greater number of documents, because it doesn't provide an aggregated view. This search engine works at http://www.kartoo.com/.

2.3 KwMap

KwMap is more a keyword refining tool which should help user to find additional keywords for another search engine. This tool displays related keywords in 2 lines (2 topics) which cross exactly in the search string. This method is useful for keywords which occur mostly in 2 topics but not more, not less.



Picture 3

In Picture 3 the result of "bindings" search is displayed. Since this word is mostly related to snowboarding or skiing the 2 lines are about snowboarding and skiing. However this word also occurs in IT-related documents (as displayed in Picture 25). This fact is not visible in this search engine's visualization. This tool runs at http://www.kwmap.net/.

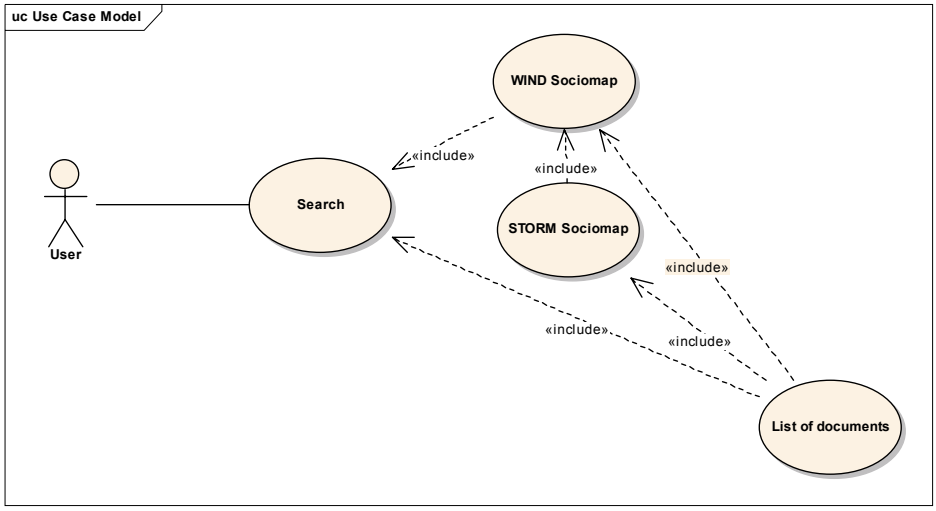
None of the 3 methods described above provide an overview of more documents (tens or hundreds) using keywords together with number of documents to which these keywords are relevant and mutual relations of the keywords (e.g. "bindings" is usually contained on a document where "ski" or "snowboard" is contained as well and "Burton"[1] is much more contained on documents with "snowboard" than "ski").

[1] Burton is a snowboard brand

3 Sociomapping Method Description

3.1 Usage

The idea of this method is to provide the user not only list of documents and their clusters but also a visual representation of the document space using keyword clusters and layout of the documentation with regards to these keyword clusters as it may be seen in Picture 4. In fact method described in this work should do the same job as KwMap does (display keyword clusters visually) but improve it by assigning documents to these keyword clusters and display them in a human readable form.



Picture 4

3.2 Problem Formalism

In previous chapter words like “document” or “document space” have been used without any formal definition. These definitions will be provided in this chapter.

Definition: Keyword is a text string.

Usually keyword is a word or a phrase of words.

Definition: Document is any searchable subject which has assigned a list of keywords together with number of their occurrences in it.

Definition: Document space is a set of all documents which can be returned as a part of

any search result.

According to this definition document may be even for example an image or a music file with a list of keywords assigned. This work will be interested only in text documents due to the simplicity of the results checking and amount of data needed to store however in principle there is no problem to apply method described in this work on non-text documents.

Definition: Keyword frequency is defined as number of occurencies of j^{th} keyword (t_j) in i^{th} document (d_i).

Definition: Inverse document frequency of j^{th} keyword (Idf_j) is defined by Formula 1

$$Idf_j = \log \frac{|D|}{|\{d \in D : t_j \in d\}|}$$

Formula 1

In Formula 1 the following notation is used:

D	Set of all documents
d	Set of all keywords which are presented in the document d
t_j	j^{th} keyword

The greater Idf_j is the less documents within the document space contain this keyword and therefore the more valuable it is to have the document containing this keyword within the search results.

Definition: Static document space is a document space with directly accessible index, which is used to search for documents, built above it. If static document space changes the index has to be changed as well.

In this work static document space (and its index) is stored on a local media. This index contains *Tf* and *Idf* of all keywords and documents respectively.

Definition: Dynamic document space is a document space where it is possible to search

only using given API[1].

In this work dynamic document space is web and API is provided by search engines (Live! Search and Yahoo!).

Definition: Document vector model (hereafter DVM) is a matrix containing weights of keywords in the returned documents. Weight of j^{th} keyword in i^{th} document will be denoted by w_{ij}. DVM is denoted by $W = \{w_{ij}\}$ [14].

Definition: Search string is a list of keywords.

Search string is input of the document search process given by a user. Relevancy of the document to search string is usually defined by Formula 2.

$$TfIdf_i = w_{i\bullet} \otimes Idf = \sum_{j \in K} w_{ij} Idf_j$$

Formula 2

In Formula 2 the following notation is used:

$TfIdf_i$	Relevancy of the given search string K to i^{th} document
$w_{i\bullet}$	Vector of i^{th} document's keyword relevancies as definded above
Idf_j	Inverse document frequency of j^{th} keyword
K	Set of indeces of keywords presented in the search string

Usually weight of the keywords within DVM is $TfIdf_i$ for search string containing only the keyword. Unfortunatelly Idf_j is not a known value in case of dynamic document spaces. If we consider Idf_j of all keywords to be 1 the weight of the keyword will be exactly its frequency within the document.

[1] Application programming interface

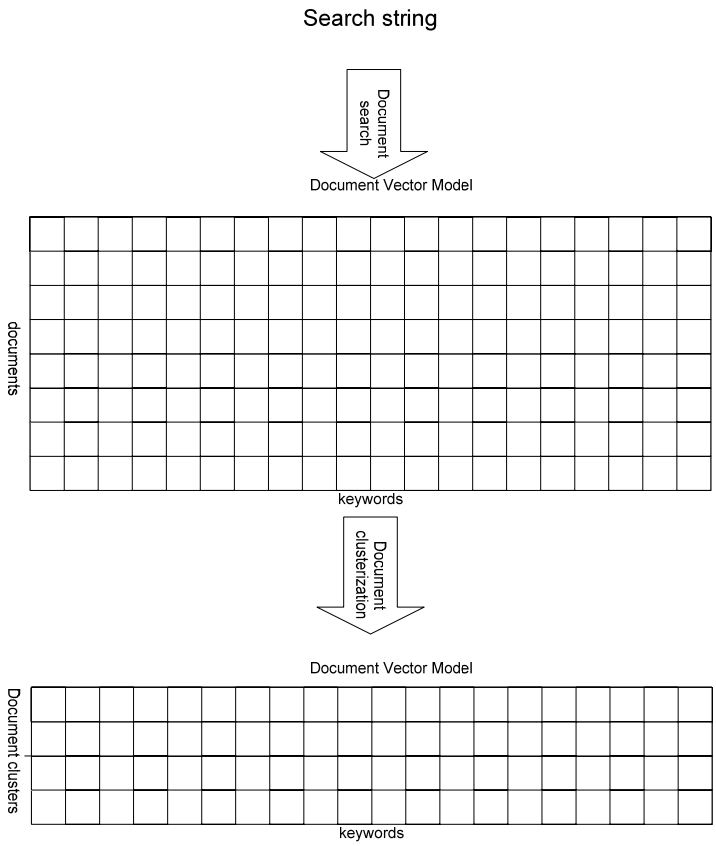
Definition: Document search means selecting a list of documents which have a non-zero weight to the search query string from the document space. This subset of document space is called search result.

This list is ordered according to weight of the document to the search query string. In this work we'll denote search result by $D = \{d_1, d_2, \ldots, d_n\}$.

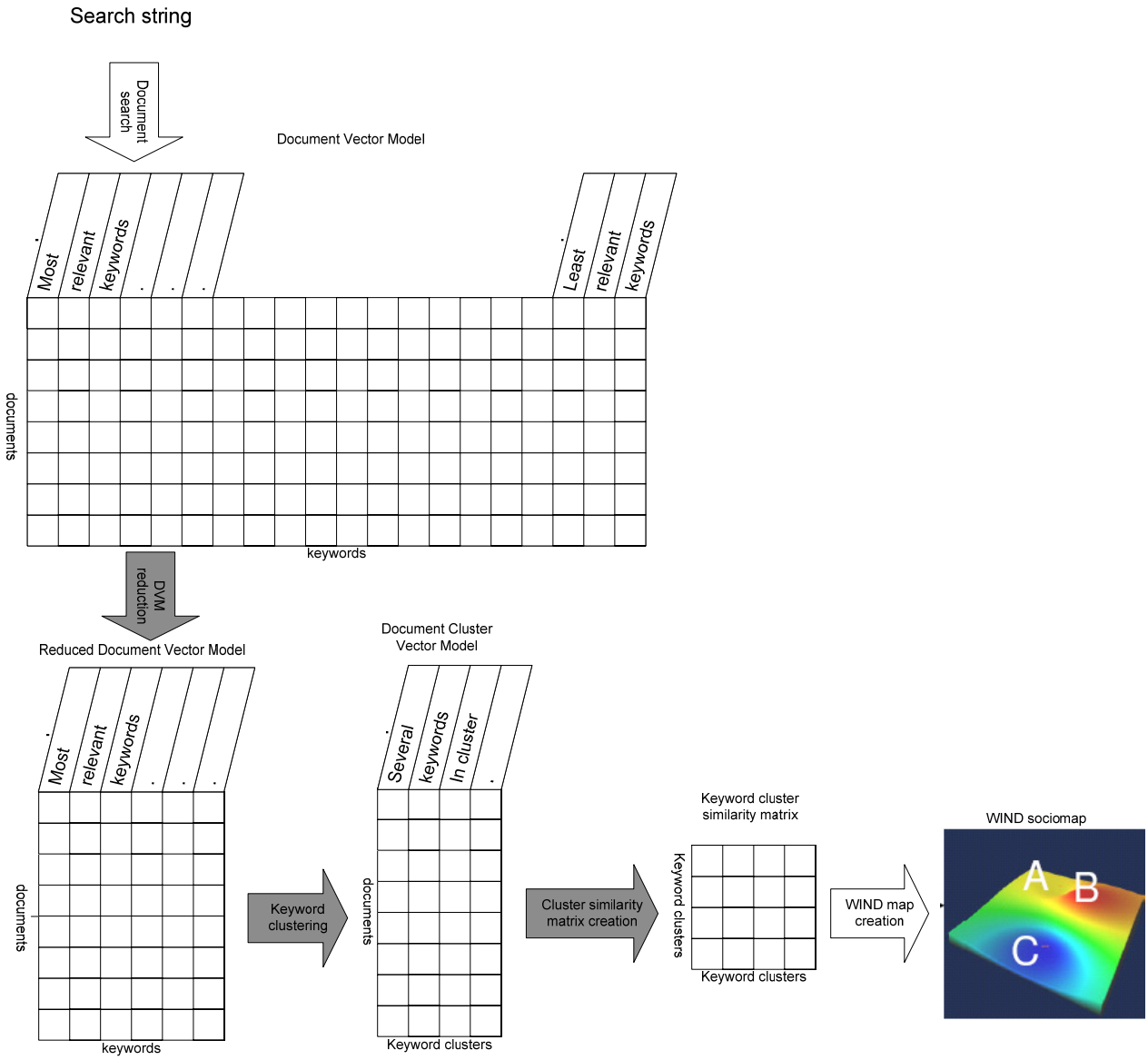
Definition: Search visualization is a graphical representation of search result.

3.3 Visualization Method Description

The purpose of sociomap is to present relation of several data objects in a human-readable form. In order human readability to be achieved the sociomap has to display relatively low number of objects. In this chapter keyword clusters are considered to be objects. Since the number of keywords in a search result DVM is relatively high (usually hundreds or thousands) this number has to be minimized with as small loss of information as possible. Majority of the keywords within DVM is presented only in few documents and therefore does not carry any usable information of the documents within the search result from the global point of view. This allows disregarding these keywords (in Picture 6 marked as Reduction of DVM). However there are still several tens of keywords which appear in more documents. Probably the best way how to display these keywords in the resulting sociomap is to cluster them to a small number of keyword clusters (in Picture 6 marked as Keywords clustering) which can be displayed in a human-readable sociomap. Document clustering is a problem which was solved in several papers [4-6, 10-12]. All of these papers consider rows of the DVM (documents) as items (as displayed in Picture 5). In this work columns of DVM (keyword vectors) will be the items which will be clustered. This makes a good sense because the keyword is consider to be similar with another keyword if both keywords are contained in the same documents. In Picture 6 we may see whole process of creating WIND sociomap where arrows are used as data transformations. White arrows mean that a 3rd party API was used to achieve this transformation and gray arrows mean that this transformation is a part of this work. This process will be described in the next chapters. Name of the chapter is equal to the text in the corresponding process arrow.


Search string
Document search
Document Vector Model
documents
keywords
Document clusterization
Document Vector Model
Document clusters
keywords

Picture 5



Picture 6

3.3.1 Document Search

For testing of search visualization using sociomaps 6 document spaces were used. 3 of them were static (Enron mail database [16] with Oracle text keyword index, Enron mail database [16] with Lucene – http://lucene.apache.org/ fulltext index and categorized Reuters articles database [1]) and 3 of them were dynamic (Live! - http://dev.live.com/livesearch/, Yahoo! - http://developer.yahoo.com/search/boss/ and Google).

In the last 2 static document spaces both *Tf* and *Idf* are known values and therefore

weights of the keywords are their *TfIdf* values (see Problem Formalism). In none of the dynamic document spaces both *Tf* and *Idf* are not known values.

Definition: Snippet is a short text returned as a part of search result. This text is contained in the target document and it is in neighborhood of the keywords presented in the search string.

Definition: Word stoplist is a list of words which have low *Idf* and therefore appear in most of the documents.

Searching according to words in stoplist doesn't make sense, because almost whole document space would match the search string.

All 3 APIs of dynamic document spaces return a snippet which is used to create DVM. *Tf* means number of occurrences of the keyword within this snippet. *Idf* is 0 if the word is in word stoplist and 1 otherwise.

3.3.1.1 Linguistic Notes

In this work we work with documents in English for simplicity. For purpose of construction of DVM we consider plural of a noun to have the same meaning as singular. In case of static document spaces this problem is already solved by text index Lucene.Net which considers plural and singular of a noun to be the same. This problem therefore exists only when searching in dynamic document spaces when the list of keywords is created from the snippet. The easiest approach is to trim "s" at the end of a noun. This also makes verb forms in 3rd person the same as other forms of the same word which is desirable as well. Unfortunately it will also cause misspelling of several other words ending by "s". Since no plural form of a noun contains "ss" at the end only the words with "s" at the end and any other letter before the "s" will be considered as plurals. Additionally no noun is shorter than 2 characters in singular form and therefore words with length of less than 3 characters won't be considered as plurals. To make it clear let's define a word in plural form for purpose of visualization.

Definition: Word in plural form is a word which has at least 4 characters, last letter is equal to "s" and the letter before the last letter is not equal to "s".

According to keywords list taken from Reuters database [1] (where no nouns in plural

form are stored) there are 1371 unique words matching the word in plural definition out of 47,236 in the database. If we consider that some of these words ending by "s" are misspelled (e.g. "boathous") it is an acceptable price for remedy of this problem.

The problem with words of the same meaning is much more difficult to solve. Other example of such set of words is a set of synonyms (words with different spelling and pronunciation but the same meaning). For production use there would have to a kind of dictionary of words with the same or similar meaning would have to be used. The fact that synonyms problem may be combined with homographs (words with the same spelling but different meaning) problem and therefore the meaning of the word depends on the context of the word occurrence within the text document (e.g. "plant" may be considered as "flower" or as "factory") makes this problem much more sophisticated and its solution is not a subject of this work.

In Enron database the messages usually contain phone (and other) numbers which appear in a human readable form with dashes (e.g. "222-55-11-55") and therefore e.g. "222" appears in the database as a keyword in many documents. Since it doesn't have any specific meaning by itself it should be skipped. This is the reason why all keywords containing only numeric characters are removed from DVM when Enron database is used.

3.3.2 DVM Reduction

As stated above most of the keywords within DVM do not carry useful information about overview of the search result documents which allows us to remove them.

Definition: The relevance of the keyword to the search result is defined as number of documents in the search result to which this keyword has non-zero relevance.

In order to remove irrelevant keywords from the DVM keywords are sorted decreasingly according to their relevance to the search result and only several first keywords are added to reduced DVM. In Picture 6 this process is marked as DVM reduction. Reduced DVM will be input of all clustering algorithms in Keyword Clustering.

3.3.3 Keyword Clustering

Clustering is a process of creating a small number of clusters (denote the number of clusters by C and i^{th} cluster by c_i) from a set of M vectors $v_1, v_2, \ldots v_M$ ($M >> N$) where

$c_1 \cup c_2 \cup ... \cup c_N = \{v_1, v_2, ..., v_M\}$ and $c_1 \cap c_2 \cap ... \cap c_N = \varnothing$. The aim of clustering is to put similar vectors to one cluster. Similarity is usually measured by distance from an "average vector of the cluster" - cluster centroid. Euclidian (Formula 3), cosine (Formula 4) and Manhattan (Formula 5) distance functions are used as similarity functions in this work.

$$d_e(v, w) = \sqrt{\sum_i (v_i - w_i)^2}$$

Formula 3

$$d_c(v, w) = 1 - \frac{v \times w}{\|v\| \cdot \|w\|} = 1 - \frac{\sum_i v_i w_i}{\sqrt{\sum_i v_i^2} \cdot \sqrt{\sum_i w_i^2}}$$

Formula 4

$$d_M(v, w) = \sum_i |v_i - w_i|$$

Formula 5

Keywords together with their relevancies in documents (columns of DVM) will be considered as clustered vectors. All three distance formulas should make sense, because they express vague formulation of keywords similarity: "Keywords are similar when they are related mostly to the same documents". Several clustering algorithms will be tested and compared in order to achieve the best results in search results visualization. In this chapter a brief description of all clustering algorithms which have been tested will be given.

Keep in mind that keywords are related to the documents (according to DVM) and therefore final phase of creating keyword clusters is assigning documents to these clusters according to the selected similarity function. For purpose of sociomap creation crisp membership of document in keyword cluster will be considered (further explanation will be given in Cluster Similarity Matrix Creation).

3.3.3.1 K-means

Let desired final number of clusters (denote it by k) is given. Standard K-means algorithm works in the following way:

```
(1)      initialize k cluster centroids;
(2)      do
(3)        assign every item to cluster with the lowest distance
   between the item and the cluster centroid[1];
(4)        calculate new centroids as average points of all cluster's
   members
(5)      while at least 1 item is assigned to a different centroid
   than it was assigned to during previous iteration of this cycle;
```

Usually centroids are initialized (line (1)) randomly. We can't initialize the centroids randomly, because it would make k-means instable (the result of algorithm would be different result sets with the same input). At the beginning of clustering clusters are sorted according to number of documents for which they are relevant. The ideal clustering result is a set of k clusters around k most relevant keywords and therefore we'll start k-means with the centroids equal to first k keyword vectors. After performing line (3) one or more clusters may be empty (empty cluster is a cluster which doesn't contain any keyword vector). Such cluster will remain empty in next iterations of the main cycle (lines (2) – (5)) as well, because centroid of empty cluster is keyword vector with relevance of 0 to all documents. This is the reason why we remove such cluster, so result of k-means may be a cluster set of k or less clusters of keyword vectors. Average point on line (4) is calculated according to Formula 6 where c_{ij} is j^{th} element of i^{th} cluster centroid, C_i is set of indexes of keywords which belong to ith cluster and w_{kj} is a member of search result DVM.

$$c_{ij} = \frac{1}{|C_i|} \sum_{k \in C_i} w_{kj}$$

Formula 6

K-means algorithm used in this work works in the following way:

```
(1)      sort keywords according to number of documents to which they
   are relevant;
(2)      set first k keyword vectors as centroids;
(3)      do
```

[1] If the minimal distance is achieved with more centroids the first one is chosen.

```
(4)        assign every keyword to the cluster with the lowest
   distance between the corresponding keyword vector and the cluster
   centroid;
(5)        remove clusters with no keywords assigned;
(6)        calculate new centroids as average points of all cluster's
   members;
(7)      while at least 1 item is assigned to a different centroid
   than it was assigned to during the last iteration of this cycle;
```

When this algorithm finishes every document in search results is assigned to the cluster which has maximum value of *TfIdf* similarity to the keyword vector [5].

3.3.3.2 Optimized K-means

Optimized K-means algorithm is fully described by Wang in [12]. This clustering algorithm is in this work applied on keyword vectors rather than on document vectors. Let us recall original K-means algorithm.

```
(1)      initialize k cluster centroids;
(2)      do
(3)        assign every item to cluster with the lowest distance
   between the item and the cluster centroid[1];
(4)        calculate new centroids as average points of all cluster's
   members
(5)      while at least 1 item is assigned to a different centroid
   than it was assigned to during previous iteration of this cycle;
```

The main and only difference between original and optimized K-means is different line (3). In optimized K-means the main thought of Self-Organizing maps algorithm (hereafter SOM) is applied. SOM were originally described by Kohonen in [14]. When a new vector is assigned to the cluster its centroid moves towards the currently added vector. Since purpose of this movement is to decrease dissimilarity of items within cluster the size of the movement has to decrease with increasing number of items within the cluster, so there is a scalar move factor *a(n)* where *n* is number of vectors in the current cluster. However the first vector which is assigned to the cluster may not be very similar to the vectors which will be assigned to the cluster in the future optimized K-means modifies the centroid placement only in case when the vector has been assigned to the same cluster also in the previous iteration of global K-means cycle (lines (2) – (5)). More formally in optimized K-means algorithm line (3) from original K-means is replaced by the following algorithm (set of all vectors to be clustered is denoted as *vectors*).

[1] If the minimal distance is achieved with more centroids the first one is chosen.

```
(1)      for each vector in vectors
(2)        c := cluster with centroid nearest to vector;
(3)        if (vector was assigned to c during last iteration of
   global K-means cycle) then
(4)          assign vector to c;
(5)          c.centroid := c.centroid + a(c.itemsCount) * (vector -
   c.centroid);
(6)        else
(7)          add vector to unassignedVectors;
(8)        end if;
(9)      end for;
(10)     for each vector in unassignedVectors
(11)       c := cluster with centroid nearest to vector;
(12)       assign vector to c;
(13)     end for;
```

In [12] Wang proposes the new position to be calculated according to Formula 7 and scalar move factor defined by Formula 8.

$$c(i+1) = c(i) + \alpha(i)v$$

Formula 7

$$\alpha(i) = 2^{-i}$$

Formula 8

This formula led to wrong results and therefore Formula 9 is used in optimized K-means implementation in this work.

$$c(i+1) = c(i) + 2^{-i-1}(v - c(i))$$

Formula 9

In Formula 7, Formula 8 and Formula 9 the following notation is used:

c(i+1) centroid of the cluster after adding $(i+1)^{th}$ vector (new centroid)

c(i) centroid of the cluster containing *i* vectors

v vector which is currently being assigned to the cluster

Documents are assigned to the clusters in the same way as in case of K-means.

3.3.3.3 C-means

C-means may be considered as K-means with fuzzy logic implemented. Let

$U = \{u_{ij} \in \langle 0;1 \rangle\}$ be a keyword-cluster membership matrix (according to standard fuzzy logic – 0 means no membership, 1 means total membership) where $\forall i \sum_k u_{ik} = 1$ (each keyword's membership to clusters is distributed among clusters so that total membership of any keyword to all clusters is 1). C-means algorithm works in the following way:

```
(1)      initialize U;
(2)      do
(3)        calculate centroids;
(4)        calculate items' membership to clusters (update U);
(5)      while at least one element of U has been changed by more
   than epsilon during this iteration;
```

Within C-means algorithm description we denote centroid of j^{th} cluster c_j, degree of membership as $U = \{u_{ij} \in \langle 0;1 \rangle\}$, keyword vector of i^{th} keyword k_i and number of clusters as C. Centroids calculation (line (3)) is performed according to Formula 10 and Clusters' membership (line (4)) accrding to Formula 11 where d is distance function of 2 vectors.

.

$$c_j = \frac{\sum_i u_{ij}^m k_i}{\sum_i u_{ij}^m}$$

Formula 10

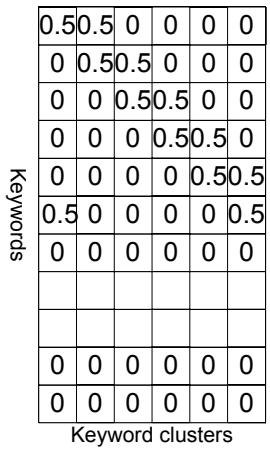
$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{d(k_i, c_j)}{d(k_i, c_k)} \right)^{\frac{2}{m-1}}}$$

Formula 11

It is not possible to initialize the U matrix similary as in k-means to total membership of first C keyword vectors ($u_{ij} = 1$ for $i = j$ and $u_{ij} = 0$ otherwise) because in this case there would exist $c_j = k_i$ which would cause that $d(c_j, k_i)$ would be zero and therefore the denumerator of Formula 11 wouldn't be defined. The U matrix can't be initialized randomly because stability of clustering algorithm is required. The U matrix therefore will be initialized in the following way:

$$u_{ij} = \begin{cases} 0.5 \forall i = j \vee i = (j+1) \bmod C \\ 0 otherwise \end{cases}$$

An example of such initial U matrix is on Picture 7.



0.5	0.5	0	0	0	0
0	0.5	0.5	0	0	0
0	0	0.5	0.5	0	0
0	0	0	0.5	0.5	0
0	0	0	0	0.5	0.5
0.5	0	0	0	0	0.5
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Picture 7

When this algorithm finishes every document in the search results is assigned to the cluster which has maximum value of *TfIdf* similarity [5] to the keyword vector weighted by the keyword membership to the cluster.

3.3.3.4 Hierarchical Clustering

Hierarchical clustering is a clustering algorithm which in fact creates a forest (set of trees) of keyword clusters trees.

```
(1)       forest := all_keywords;
(2)       while |forest| > maxClusters
(3)         kws := getNearestNodes(forest);
(4)         forest := forest – kws[0] – kws[1];
(5)         node.Left := kws[0];
(6)         node.Right := kws[1];
(7)         node.Centroid := (node.Left.Centroid +
   node.Right.Centroid) / 2;
(8)         forest := forest + node;
(9)       end while;
```

This algorithm begins with a forest of trees containing exactly 1 node with their Centroids equal to the corresponding keyword vectors (line (1)). In each iteration 2 nodes with nearest centroids are found (line (3)), new tree node is constructed from these 2 nodes and it is added into the resulting forest (lines (4) – (8)). The clusters are the trees in the forest. Each cluster contains all keywords within its tree.

Advantage of this algorithm in comparison with K-means and C-means is its determinism. Since number of items within the forest is decreased by 1 during each iteration and getNearestNodes's effectivity varies from $O((n)^2)$ to $O((n-m)^2)$ the effectivity of this algorithm is $O((n-m)^2 n)$ where n is number of keywords and m is desired number of clusters.

3.3.4 Cluster Similarity Matrix Creation

Before sociomap may be created matrix of relations among keyword clusters has to be calculated. Relations matrix is a matrix (denote it by $R = \{r_{ij}\}$) of dimensions N x N where N is number of objects in the sociomap and r_{ij} is normalized relation of i^{th} object to j^{th} object. For this purpose inverse distance is used (Formula 12) where c_i and c_j are cluster centroids and d is the same distance function as was used when creating keyword clusters (see Keyword Clustering).

$$r_{ij} = \frac{1}{d(c_i, c_j)}$$

Formula 12

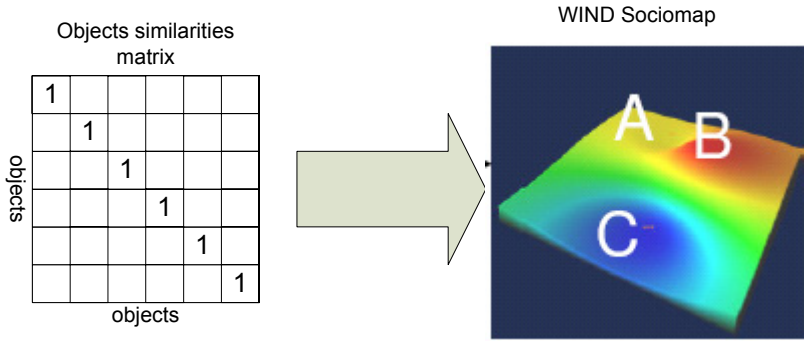
Additional input of sociomap creation process are object heights. In the case of keyword clusters this information may be used to tell the user how many documents are mostly related to this keyword cluster. Since clusters with more keywords would have much more documents assigned to themselves height is not number of documents itself but ratio of number of documents and number of keywords in the cluster. Heights of all objects (clusters) are normalized to interval $\langle 0;1 \rangle$ in the same way as relations are normalized (Formula 13).

3.3.5 WIND Map Creation

Sociomapping is a method of displaying data objects according to their mutual similarities (or relations) into 2D space so that mutual distance of objects is this 2D space is as similar as possible to their mutual similarities. 3rd dimension (height and color) is used to express specified attribute of the data items. This method has already been used to display e-mail communication in Mail Archiver software project [3] and many other applications. In Mail Archiver application objects were users, number of messages sent among these users

were their relations and height was defined as number of messages sent by the user.

As described in [2] there are 2 kinds of sociomaps: direct sociomaps (WIND[1] maps) and indirect sociomaps (STORM[2] maps). Only brief introduction into Sociomapping specific to usage for document search visualization is provided in this chapter. Full description is given in [2] and related work.



Picture 8

WIND map is in this work used visualize mutual similarities of several[3] keyword clusters (let's recall that number of clusters was denoted by *N*). In this work keyword clusters will be considered as objects (in Picture 8 denoted by letters A, B and C) and their mutual similarities will be the facts which will be displayed in WIND map. As we can se in Picture 8 input data for direct sociomap is a similarities matrix. Calculation of this matrix is described in Cluster Similarity Matrix Creation. In this work keyword clusters similarity (defined in Keyword Clustering) will be considered as objects similarity. These similarities should be displayed in WIND map as Euclidian distances between objects represented by points in 2D space. Euclidian distance has been selected, because human lives in 3D space where Euclidian distance is used to measure distance and therefore he is used use this distance. Although WIND map doesn't require similarities matrix to be symmetric only symmetric relations matrices will be used in this work. All values in relations matrix are normalized to <0; 1> interval of real numbers using Formula 13.

---

[1] WIND stands for Weighted INverse Distance

[2] STORM stands for Subject To Object Relations Matrix

[3] Sociomaps are human-understandable when number of object count is at most 25

$$r_{kl} = \frac{r_{kl}' - \min_{i\in\{1,2,\ldots n\};j\in\{1,2,\ldots n\};i\neq j} r_{ij}'}{\max_{i\in\{1,2,\ldots n\};j\in\{1,2,\ldots n\};i\neq j} r_{ij}' - \min_{i\in\{1,2,\ldots n\};j\in\{1,2,\ldots n\};i\neq j} r_{ij}'}$$

Formula 13

Since $\min_{i\in\{1,2,\ldots n\};j\in\{1,2,\ldots n\};i\neq j} r_{ij}'$ and $\min_{i\in\{1,2,\ldots n\};j\in\{1,2,\ldots n\};i\neq j} r_{ij}'$ are global values for whole result set Formula 13 may be rewritten to a more familiar linear function form (Formula 14) from which it is immediately visible that the normalization is linear.

$$r_{kl} = \frac{r_{kl}' - m}{\delta}$$

Formula 14

In Formula 13 and Formula 14 the following notation is used:

n number of objects which should be displayed by sociomap

r_{kl}` original value of relationship of k^{th} object to l^{th} object

δ distance between miminum and maximum value of all objects` relationships

n minimum value of all objects` relationships

r_{kl} normalized value of relationship of k^{th} object to l^{th} object

Observation 1: Normalization according to Formula 13 is linear and therefore it doesn't change order of values (relations).

It is possible to display 3 points to a 2D space with any given mutual Euclidian distances but we'll need to display more than 3 points within a sociomap (2D space). Therefore distance approximation is used when creating direct sociomap (more specifically H-model of direct sociomap as described in [2]). Quality of this approximation is expressed by Spearman's rank correlation coefficient applied on distances among objects in direct sociomap and relations among the objects in the original (or according to **observation 1**

normalized) relations matrix[1].

The 3rd dimension in WIND map (height and color) is used to display a numerical attribute. These values are known only for the points where the objects are situated and therefore height weighted inverse distance interpolation is applied to count height of all other points on the map. The resulting map's terrain is smooth (derivation exists) everywhere. In this work this attribute is one of the 3 following functions.

$$h_c = |C|$$

Formula 15

$$h_l = \log(1 + |C|)$$

Formula 16

$$h_r = \log\left(1 + \sum_{d \in C} \frac{1}{1 + \log(1 + rank(d))}\right)$$

Formula 17

In Formula 15, Formula 16 and Formula 17 the following notation is used:

C	cluster (set of documents)
rank(d)	0 based rank of the document from result set (1st document has rank 0)
h_c	count height
h_l	logarithmic height
h_r	rank height

As described in Experiments count height doesn't return good resulting map, because usually there are few (not more than 4 clusters) clusters with a great count of documents

[1] Usually Spearman's rank correlation coefficient decreases with increasing number of objects

and the rest of the clusters has less than 10 times less documents. This makes the resulting map to have only few "hills" and the rest is almost "flat ground".

Logarithmic height makes the difference between clusters with a great count of documents and those with small count of documents much smaller and therefore (as it may be seen in Experiments) the resulting maps are much more user friendly than the same maps with count heights. Logarithmic height compares number of documents of each cluster non-linearly (difference between 0 and 50 is more evaluated than difference from 50 to 100 although the linear difference is the same). Let's assume that the user doesn't compare absolute number of documents in the cluster but ratio of numbers of documents in the 2 clusters (difference between 5 and 10 documents is evaluated equally to difference between 10 and 20). Then the logarithm function is suitable for this utilization (see calculation in Formula 18 where n+1 is number of documents in the cluster).

$$\frac{\log(2n)}{\log(n)} = \frac{\log(2)+\log(n)}{\log(n)} = \frac{\log(2)}{\log(n)} + 1 \approx$$
$$\approx \frac{\log(2)}{\log(2)+\log(n)} + 1 = \frac{\log(2)+\log(2)+\log(n)}{\log(2)+\log(n)} = \frac{\log(4)+\log(n)}{\log(2)+\log(n)} = \frac{\log(4n)}{\log(2n)}$$

Formula 18

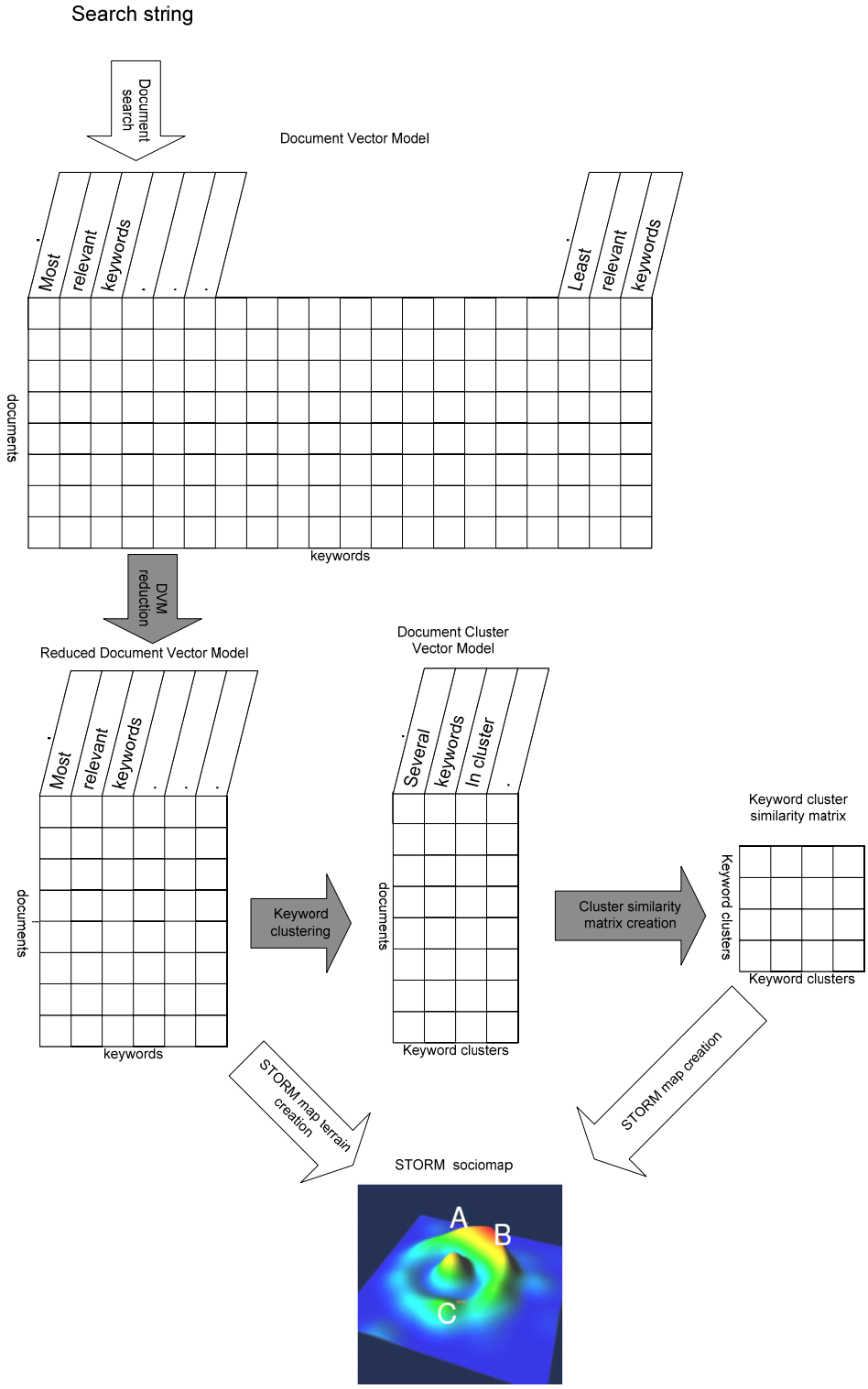
Since all search engines return the document list sorted according to relevance of the document to the search string this information may be used in cluster height (the contribution of the document to the height of the cluster where it belongs decreases with increasing rank of the document).

3.3.6 STORM Map Creation

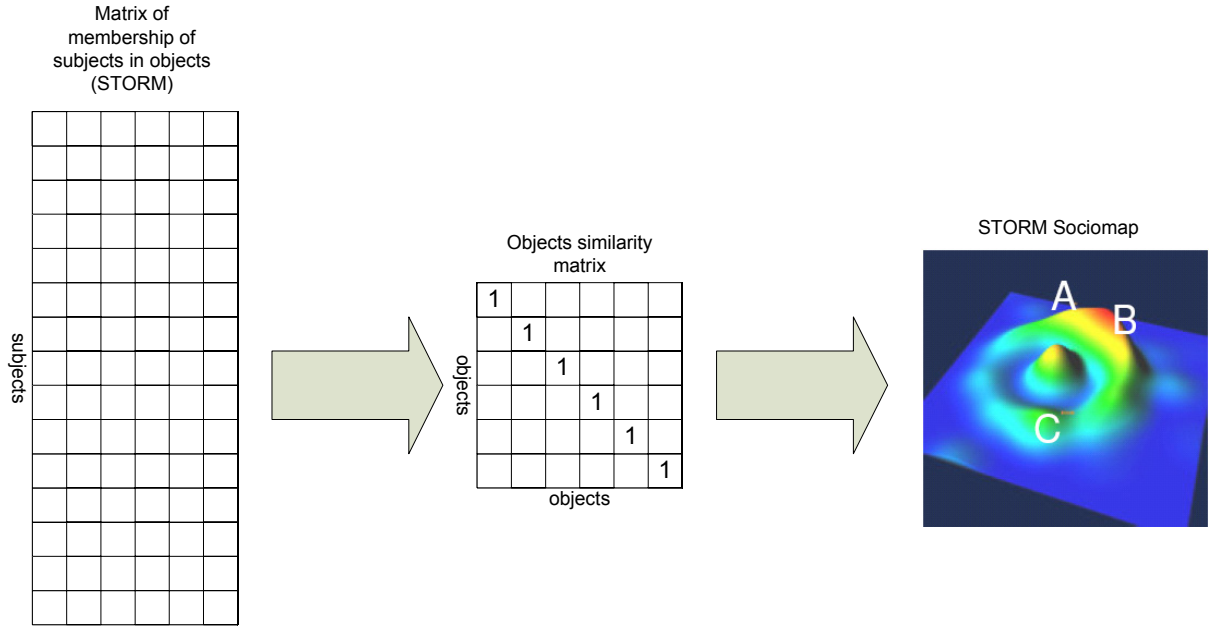
Purpose of indirect sociomaps is to display relation of great number[1] of subjects (we'll denote number of subjects as M) to several objects (denoted by N). Similarly to direct sociomap keyword clusters are considered to be objects, documents are considered as subjects and membership of documents to keyword clusters is considered as relations among subjects and objects. In this work input of indirect sociomap is matrix $S = \{s_{ij}\}$ of

[1] Increasing count of subjects doesn't affect human readability of the sociomap

dimension M x N where s_{ij} is membership of i^{th} document to j^{th} keyword cluster (hereafter STORM) together with the keyword clusters similarity matrix (see Cluster Similarity Matrix Creation). Additionally matrix S contains relevancy of single keywords to the documents. This means that membership of document in cluster is not crisp anymore as it was when creating WIND map but fuzzy. If we denote number of keywords as K then STORM has dimension M x $(N + K)$ where only the first N columns (degree of membership of the documents to clusters) are used to create indirect sociomap (placement of the objects and terrain) and the rest of the columns is used only for statistical testing on indirect sociomap which will be described in Statistical Testing. Whole process of STORM map creation is displayed in Picture 9. Gray arrows in Picture 9 are data transformations which are part of this work. The processes which haven't been discussed in previous chapters about WIND map creation will be discussed in the following chapters.


Search string
Document search
Document Vector Model
Most
relevant
keywords
Least
relevant
keywords
documents
keywords
DVM reduction
Reduced Document Vector Model
Most
relevant
keywords
documents
keywords
Keyword clustering
Document Cluster Vector Model
Several
keywords
In cluster
documents
Keyword clusters
Cluster similarity matrix creation
Keyword cluster similarity matrix
Keyword clusters
Keyword clusters
STORM map terrain creation
STORM map creation
STORM sociomap
A
B
C

Picture 9



Picture 10

The process of STORM map creation (see Picture 10) starts by calculation of objects mutual similarity matrix. Numbers of this matrix are calculated according to Formula 19.

$$r_{ij} = \frac{|S_{\bullet i} \cap S_{\bullet j}|}{|S_{\bullet i}|} = \frac{\sum_{k=1}^{M} \min(s_{ki}, s_{kj})}{\sum_{k=1}^{M} s_{ki}}$$

Formula 19

This matrix is used to place objects into the sociomap in the same way as in case of direct sociomap.

3.3.7 STORM Map Terrain Creation

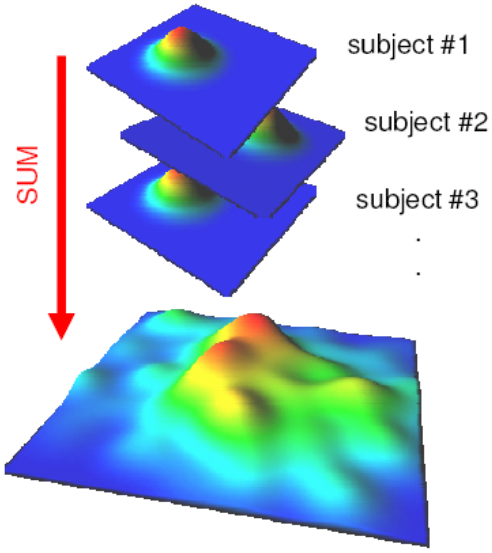
The 3rd dimension (height) of indirect sociomap displays concentration of subjects within the sociomap. Each subject is placed into the map so that its distance to objects is the most similar to this subject's relation to the objects.

Definition: Knob is a smooth real non-increasing function $h(p):(0;\infty)\rightarrow\langle 0;1\rangle$ with the

following properties:

- $h'(0) = 0$
- $\lim_{p \to \infty} h'(p) = 0$

Adding a document to the map means adding a 3D object which is created by rotating knob around 0. Knobs and their placement into indirect sociomap may be seen in Picture 11.



Picture 11

Height of the STORM map in point p is calculated using Formula 20 where $\|p - c_i\|$ is distance of point p from position of i^{th} document.

$$H(p) = \sum_{i=1}^{M} h(\|p - c_i\|)$$

Formula 20

Obviously according to the terrain creation described above the height of the terrain is fully determined by documents near the point (the more documents is near the point the greater height the point has).

This means that unlike in WIND sociomap where terrain height is defined only in the

points where the objects are situated terrain (WIND map's terrain on all other places is smoothly interpolated as Höschl describes in [2]) terrain height in STORM map carries information about placement of the subjects into the map.

The input of calculation of STORM map terrain is search DVM. Reduced DVM is created in the same way as in Document Search.

For testing of search visualization using sociomaps 6 document spaces were used. 3 of them were static (Enron mail database [16] with Oracle text keyword index, Enron mail database [16] with Lucene – http://lucene.apache.org/ fulltext index and categorized Reuters articles database [1]) and 3 of them were dynamic (Live! - http://dev.live.com/livesearch/, Yahoo! - http://developer.yahoo.com/search/boss/ and Google).

In the last 2 static document spaces both *Tf* and *Idf* are known values and therefore weights of the keywords are their *TfIdf* values (see Problem Formalism). In none of the dynamic document spaces both *Tf* and *Idf* are not known values.

Definition: Snippet is a short text returned as a part of search result. This text is contained in the target document and it is in neighborhood of the keywords presented in the search string.

Definition: Word stoplist is a list of words which have low Idf and therefore appear in most of the documents.

Searching according to words in stoplist doesn't make sense, because almost whole document space would match the search string.

All 3 APIs of dynamic document spaces return a snippet which is used to create DVM. Tf means number of occurrences of the keyword within this snippet. Idf is 0 if the word is in word stoplist and 1 otherwise.

3.3.7.1 Linguistic Notes

In this work we work with documents in English for simplicity. For purpose of construction of DVM we consider plural of a noun to have the same meaning as singular. In case of static document spaces this problem is already solved by text index Lucene.Net

which considers plural and singular of a noun to be the same. This problem therefore exists only when searching in dynamic document spaces when the list of keywords is created from the snippet. The easiest approach is to trim "s" at the end of a noun. This also makes verb forms in 3rd person the same as other forms of the same word which is desirable as well. Unfortunately it will also cause misspelling of several other words ending by "s". Since no plural form of a noun contains "ss" at the end only the words with "s" at the end and any other letter before the "s" will be considered as plurals. Additionally no noun is shorter than 2 characters in singular form and therefore words with length of less than 3 characters won't be considered as plurals. To make it clear let's define a word in plural form for purpose of visualization.

Definition: Word in plural form is a word which has at least 4 characters, last letter is equal to "s" and the letter before the last letter is not equal to "s".

According to keywords list taken from Reuters database [1] (where no nouns in plural form are stored) there are 1371 unique words matching the word in plural definition out of 47,236 in the database. If we consider that some of these words ending by "s" are misspelled (e.g. "boathous") it is an acceptable price for remedy of this problem.

The problem with words of the same meaning is much more difficult to solve. Other example of such set of words is a set of synonyms (words with different spelling and pronunciation but the same meaning). For production use there would have to a kind of dictionary of words with the same or similar meaning would have to be used. The fact that synonyms problem may be combined with homographs (words with the same spelling but different meaning) problem and therefore the meaning of the word depends on the context of the word occurrence within the text document (e.g. "plant" may be considered as "flower" or as "factory") makes this problem much more sophisticated and its solution is not a subject of this work.

In Enron database the messages usually contain phone (and other) numbers which appear in a human readable form with dashes (e.g. "222-55-11-55") and therefore e.g. "222" appears in the database as a keyword in many documents. Since it doesn't have any specific meaning by itself it should be skipped. This is the reason why all keywords containing only numeric characters are removed from DVM when Enron database is used.

Keyword clusters are then created in the same way as in Keyword Clustering. Fuzzy membership of document to keyword cluster is calculated according to Formula 21 where C_i is the set of keywords in i^{th} cluster and d_{ik} is the corresponding value of the search DVM.

$$l_{ij} = \frac{\sum_{k \in C_i} d_{ik}}{|C_i|}$$

Formula 21

Reduced DVM is attached to matrix $L = \{l_{ij}\}$ (document cluster membership matrix) which is then used for STORM map creation as described in STORM Map Creation.

STORM map allows users to see areas with abnormally high concentration of documents with high relevance to specific keyword(s) and retrieve documents within this area.

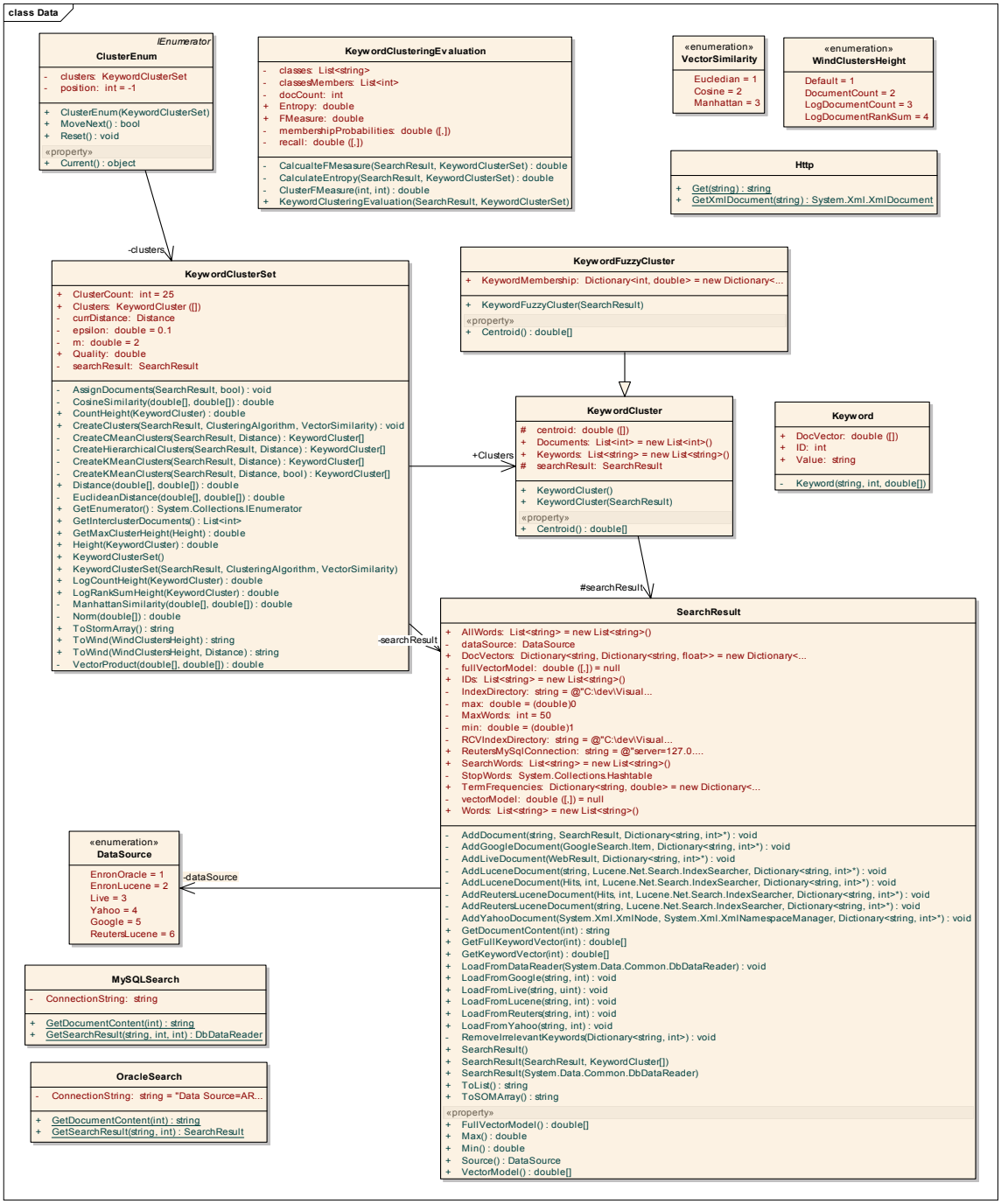
3.3.7.2 Statistical Testing

Documents placed into STORM map contain additional relevance to single keywords (count of these relevancies was denoted by K in STORM Map Creation). It is possible to test whether area on the map selected by user contains documents which have statistically significant deviation of specific keyword relevance using T-test.

4 Analysis

4.1 Organization

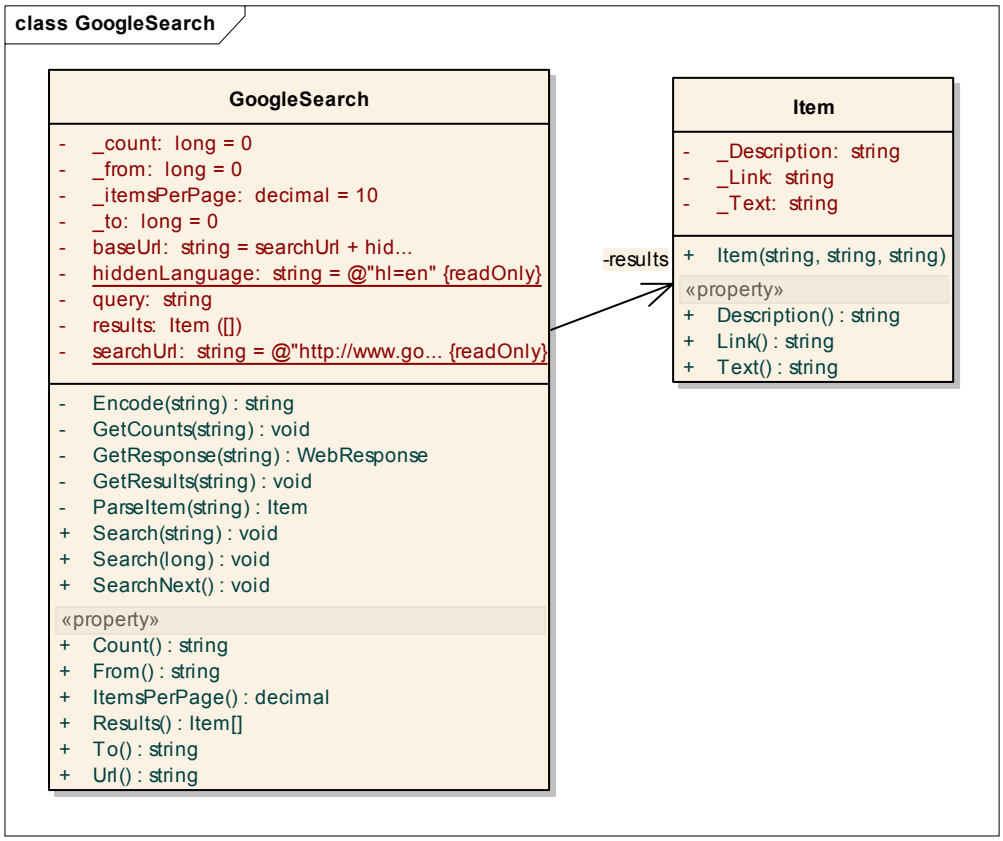
Since C# is fully object language whole program is quite easily transferrable into class diagrams. The main namespace is namespace Data (class diagram in Picture 12).



Picture 12

Class KeywordClusterSet is a set of keyword clusters which contains clustering method. It contains also methods for conversion to data needed to create WIND and STORM maps. The main property of this class is a list of instances of KeywordCluster class. This class is the class which stores keywords and documents within each cluster. KeywordFuzzyCluster derived from this class is used for fuzzy keyword – cluster membership (result of C-means). KeywordClusteringEvaluation class is used for generating values for evaluation of keyword clusters quality like Entropy and F-Measure (see Reuters Database). Other classes‘ meaning is quite self-explanative.

In order to retrieve data from Google it was necessary to create a Google pages parser. This was based on the one described at [8]. It has to be modified in order to parse websites currently returned by Google search engine. Since Google has prevention for search robots a delay has to be performed between retrieval of the result pages. Namespace containing all classes which create this parser are in GoogleSearch namespace (class diagram in Picture 13).



Picture 13

GoogleSearch is a class used for retrieving results stored as a set of instances of Item class.

4.2 Implementation

Search visualization test tool is implemented in C# (.NET 2.0). This platform has been selected mainly due to possible future implementation into Mail Archiver [3] which is based on .NET technology. All required libraries exist in .NET version and therefore there was no reason to select another platform (e.g. Java). Since sociomapping libraries exist only for Windows platform Java wouldn't bring the advantage of multi-platformness.

4.2.1 Libraries Used

The main set of libraries used when testing visualization using sociomaps is a set of sociomapping libraries. These libraries are stored in \VisualSearch\SocMaps\ directory on the attached media. Before usage of these directories it is necessary to register them by running register.bat file in \VisualSearch\SocMaps\ directory.

Since Enron mail database [16] is stored locally in Oracle database ODP.NET (Oracle Data Provider) [13] is used to connect to Oracle database.

Lucene.NET [7] is used to store fulltext index built above Enron mail database stored in Oracle.

All other libraries are standard part of .NET 2.0 libraries collection.

5 Experiments

5.1 Comparison Metrics

The best comparison metric in this case is probably user's experience. For evaluation we should have a notion of the documents within the search results. In order to make expected result more specific search keywords have been selected as test cases. The keywords have been selected so that they appear in a small number (max 4) of major topics of the document space. These topics have to contain approximately the same number of documents (approx the same partial inverse document frequency for all the topics). One such word for web is "bindings" which is mostly used for ski and snowboard bindings which will be discussed in Live! Search. Different search keyword sets have been selected for different search data sources.

Because of the fact mentioned in the previous paragraph in order to compare different clustering algorithms, metrics and data sources primarily result WIND map was used. Other important attribute is time needed to accomplish clusters calculation (all of them are iterative algorithms). If the desired result of clustering algorithm was known (in case of Reuters DB) entropy and F-Measure [12, 18] have been used to compare results additionally. Definition of these 2 comparison metrics will be given in Reuters Database together with structure of data needed for these metrics to be used.

5.2 Results on Various Data Sources

Nowadays document search is mostly performed on web and in company documents. According to [15] approximately 12 times more searches are performed on company documents than on web documents but definitely more people search on web than in company documents. Due to this fact static document spaces (company documents) and dynamic document spaces (web documents) are used for experiments equally. Since the search results (and therefore clusters and map) change in time approximate date and time when the search was performed is noted as another property of each map.

5.2.1 Enron Mail Database

In this work Enron Mail Database means database of all sent and received messages by top management of Enron [16] before its bankruptcy in late 2001. The original plan was

to use Oracle Text for searching. This plan failed, because Oracle Text produced only several document clusters using K-means even with highest number of clusters. The search result contained documents from only 1 cluster and therefore they had assigned the same keywords. Due to this problem text index was created by Lucene.Net above Enron mail database and Lucene.Net was used to search documents in Enron mail database.

Messages in this database (documents) are usually very short with only several keywords assigned. Probably because these messages were written usually in year 2001 when primary communication channels were phone and personal communication (meetings) most of the e-mail communication in Enron database is about organizing meetings, conference calls etc. and therefore related keywords (e.g. “phone”, “dial”, “meeting”, “conference”, “call” etc.).

Picture 14

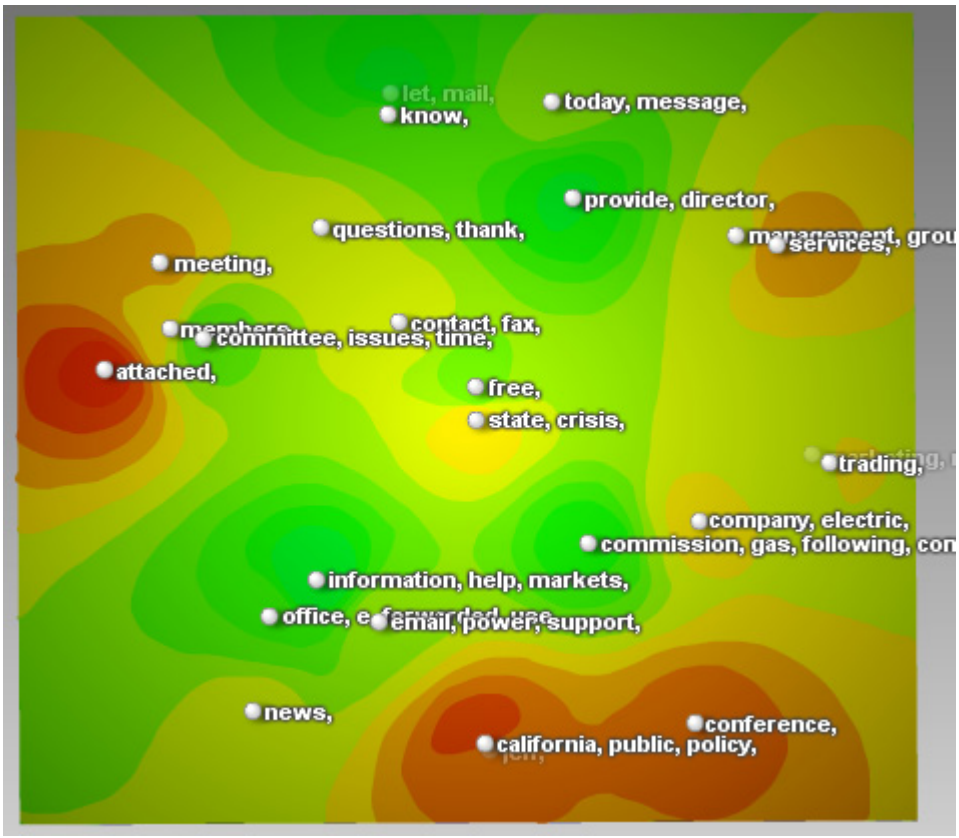
Search string	energy
Clustering algorithm	k-means
Distance function	cosine
Height function	logarithmic
Clusters calculation	4s

Picture 14 shows resulting WIND map of search result with search string “energy” (Enron was an energy company). It is quite obvious that common words are usually the most ones used. This means that word stop list used by Lucene.Net is insufficient and therefore several custom words have been added to the Lucene.Net’s default English word stop list. List of these words may be found in Word Stop List. After applying extended word stop list the same search string gives result which is displayed by WIND map as displayed in Picture 15.

Picture 15

Search string	energy
Clustering algorithm	k-means
Distance function	cosine
Height function	logarithmic
Clusters calculation	4s

Lucene uses *TfIdf* document evaluation when sorting the result set. This makes the rank of the document within the search result set quite important, because the documents with higher rank contain usually the more common words (lower *Idf*) and the ones more specific for the search string (with higher *Idf*) have lower rank. This observation is illustrated in Picture 16.



Picture 16

Search string	energy
Clustering algorithm	k-means
Distance function	cosine
Height function	rank
Clusters calculation	4s

The most relevant keywords in documents relevant to search string "energy" are the ones on the top of the search result. We may identify them using comparison of Picture 15 and Picture 16. The most important keyword cluster is "attached" (probably because e-mail communication was used to send documents – e.g. "see attached document").

5.2.2 Live! Search

Live! Search, search service run by Microsoft, is a web search service with API accessible via web services (runs on HTTP protocol). Maximum of 50 document links can be downloaded within a single request and maximum of first 1000 documents may be downloaded totally. This means that if all the document links provided by Live! Search should be downloaded 20 HTTP requests have to be performed and therefore the whole reduced DVM creation lasts several seconds. The experiments were performed on the

maximum number of documents (approximately 1000). Since some of the documents appeared more than 1 time in the result set the resulting number was less than 1000 (usually approximately 960).

Since search above dynamic document space is performed *Idf*s of keywords are not known and therefore they are considered to be 1. *Tf*s of the words are number of their occurrences within the snippet which the link provided by Live! Search is accompanied by. Word stop list (words which don't appear in keyword clusters and therefore on the map generated based on these keyword cluster) is the one in Word Stop List. Dynamic document space search results may change (and sometimes really change) in time and therefore approximate time of the search performance is noted as a property of all maps generated based on result set of dynamic document space search.

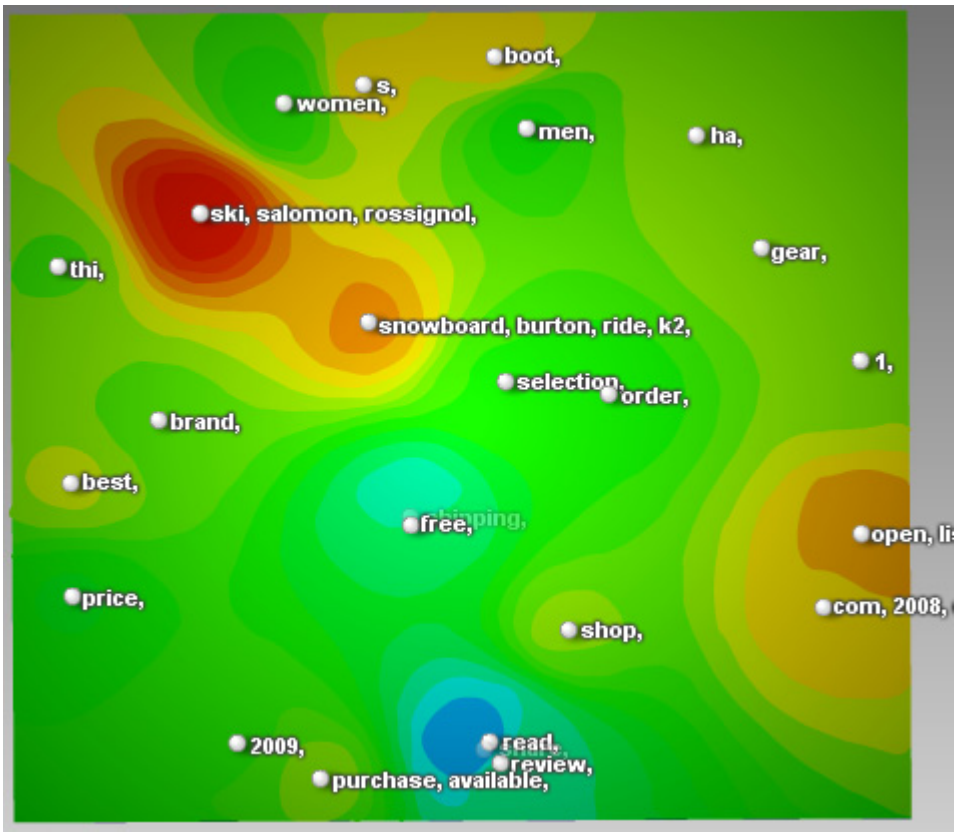
This document space was tested using the search string "bindings". This word is related mostly to ski, snowboard and wakeboard, but it also has occurrences e.g. in IT articles and on-line mailing lists. In Picture 17 WIND map of this word search result is displayed. Properties of this search are presented below Picture 17.

Picture 17

Search string	bindings
Date and time of the search	13th March 2009, 10:30
Clustering algorithm	k-means
Distance function	cosine
Height function	logarithmic
Clusters calculation	5.5s

As expected most of sites relevant to search string “bindings” is related to ski and snowboard (2 clusters on the north side of the map. There are several IT-related sites (south-eastern corner) and the sites in green area in south-western corner are probably web shop, magazines etc.

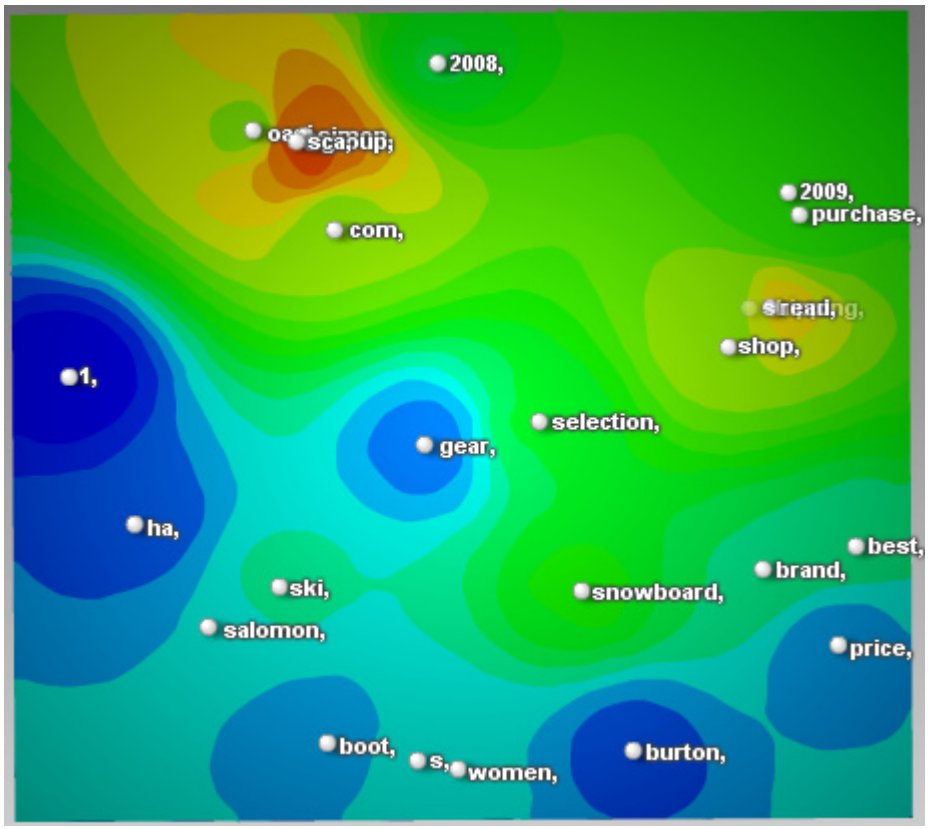
The same search result clustered by optimized k-means gives the result displayed in Picture 18 and its properties below Picture 18.



Picture 18

Search string	bindings
Date and time of the search	13th March 2009, 10:30
Clustering algorithm	optimized k-means
Distance function	cosine
Height function	logarithmic
Clusters calculation	4s

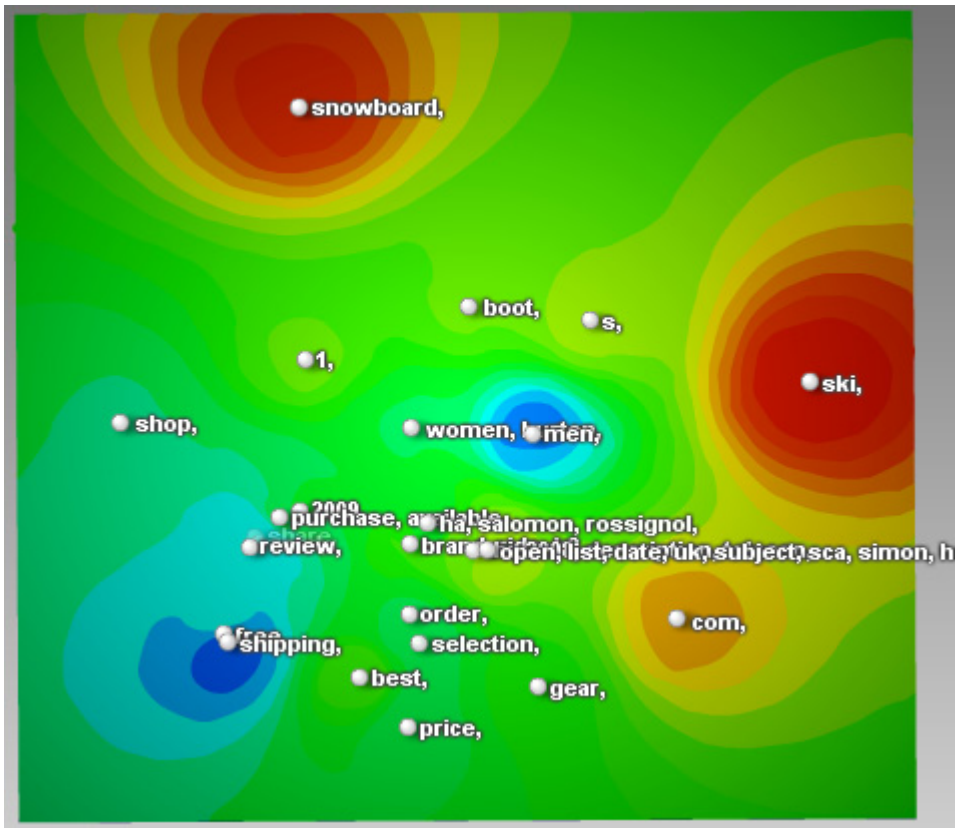
It is quite obvious that the results are very similar but the time needed to calculate clusters was approximately 30% shorter than in k-means case. For illustration the same search result has been cluster using c-means (Picture 19).



Picture 19

Search string	bindings
Date and time of the search	13th March 2009, 10:30
Clustering algorithm	c-means
Distance function	cosine
Height function	logarithmic
Clusters calculation	70s

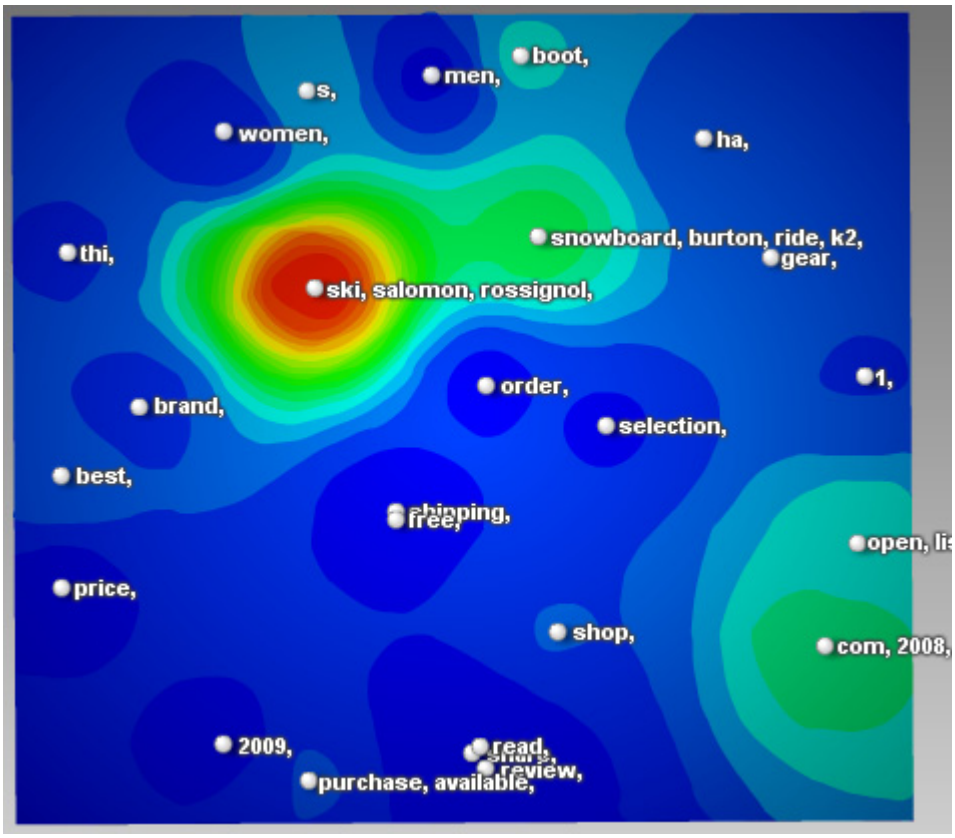
Very long calculation time (with epsilon = 0.1) and inability to interpret the results in the real world makes this method unusable. Until now the best results were results of optimized k-means algorithm. Changing distance function for k-means algorithm leads to the result in Picture 20



Picture 20

Search string	bindings
Date and time of the search	13th March 2009, 10:30
Clustering algorithm	k-means
Distance function	Euclidean
Height function	logarithmic
Clusters calculation	4s

Since Euclidean distance function takes into account number of occurences of the keywords rather than ratio of these numbers 2 clusters with ski and snowboard were created, because most of the websites concerning ski and snowboard contain these 2 words much more often than other related words (e.g. brands – Salomon, Rossignol, Burton, K2 etc.). This fact makes cosine metric better than Euclidean. Changing height function to document count doesn't improve the map either (Picture 21).

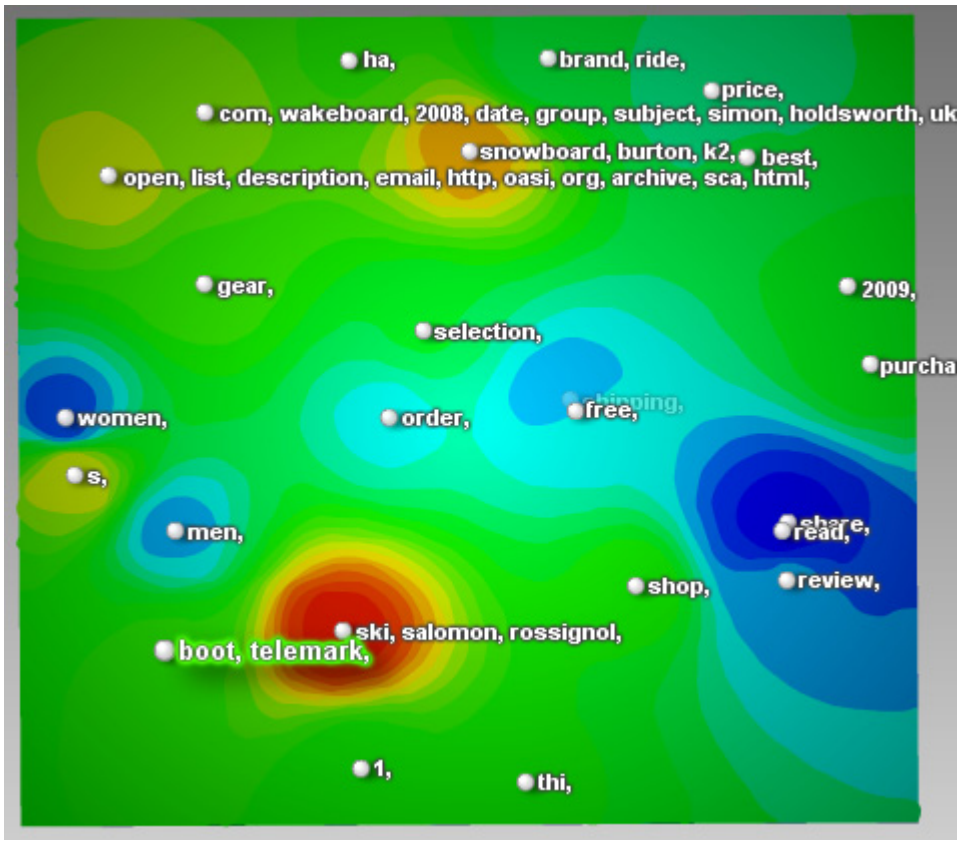


Picture 21

Search string	bindings
Date and time of the search	13th March 2009, 10:30
Clustering algorithm	k-means
Distance function	cosine
Height function	count
Clusters calculation	5.5s

Since most of clusters contain only several documents most of the map terrain's height is low (blue color).

In all of the previous experiments additional information provided by the search engine was totally disregarded – rank of the document (order within the result set). This information is taken into account in rank based cluster height (Picture 22).



Picture 22

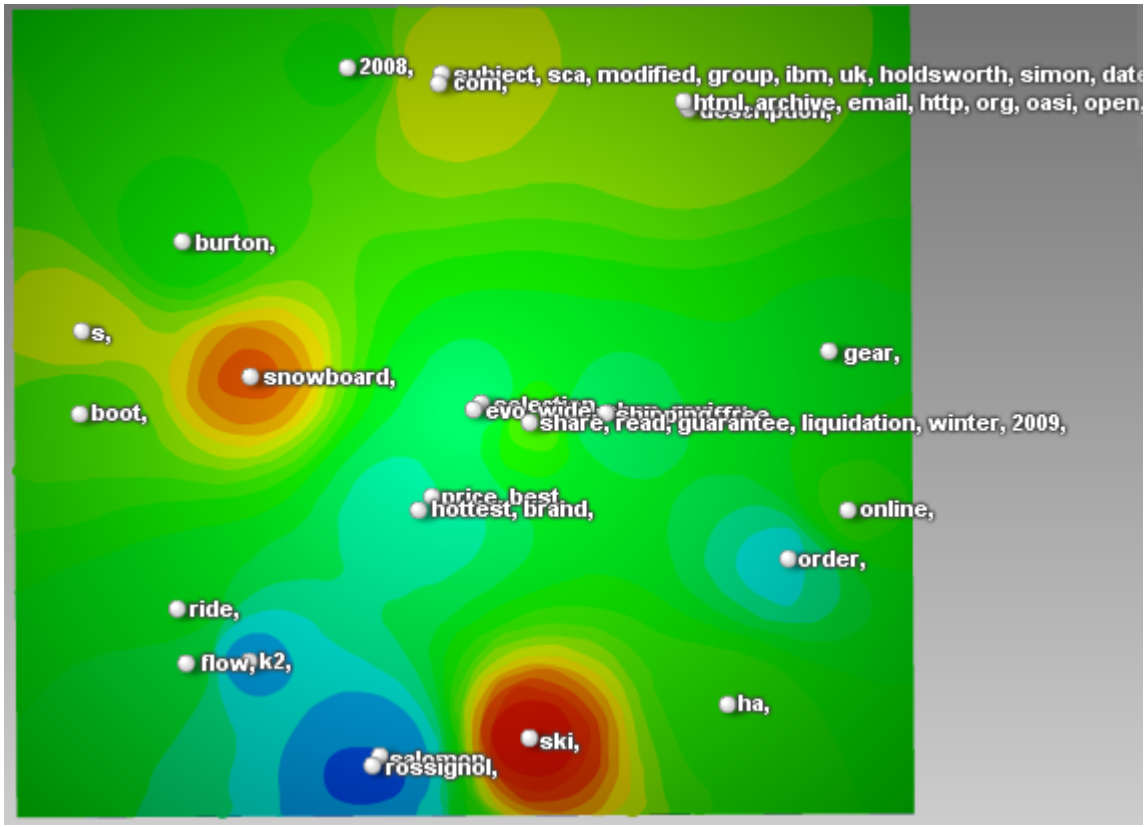
Search string	bindings
Date and time of the search	20th March 2009, 11:30
Clustering algorithm	k-means
Distance function	cosine
Height function	rank
Clusters calculation	6s

Rank based doesn't change the map terrain much. Clusters with commonly used keywords like "women", "men", "free" and "shipping" are usually presented in the documents with higher rank but more specifically used keywords like "ski", "Salomon", "Rossignol"[1], "snowboard", "Burton", "K2"[2] are presented in the documents with lower rank which makes these clusters higher. In some cases rank height function may be better but it fully depends on the user.

[1] Salomon and Rossignol are ski brands

[2] K2 is both ski and snowboard brand

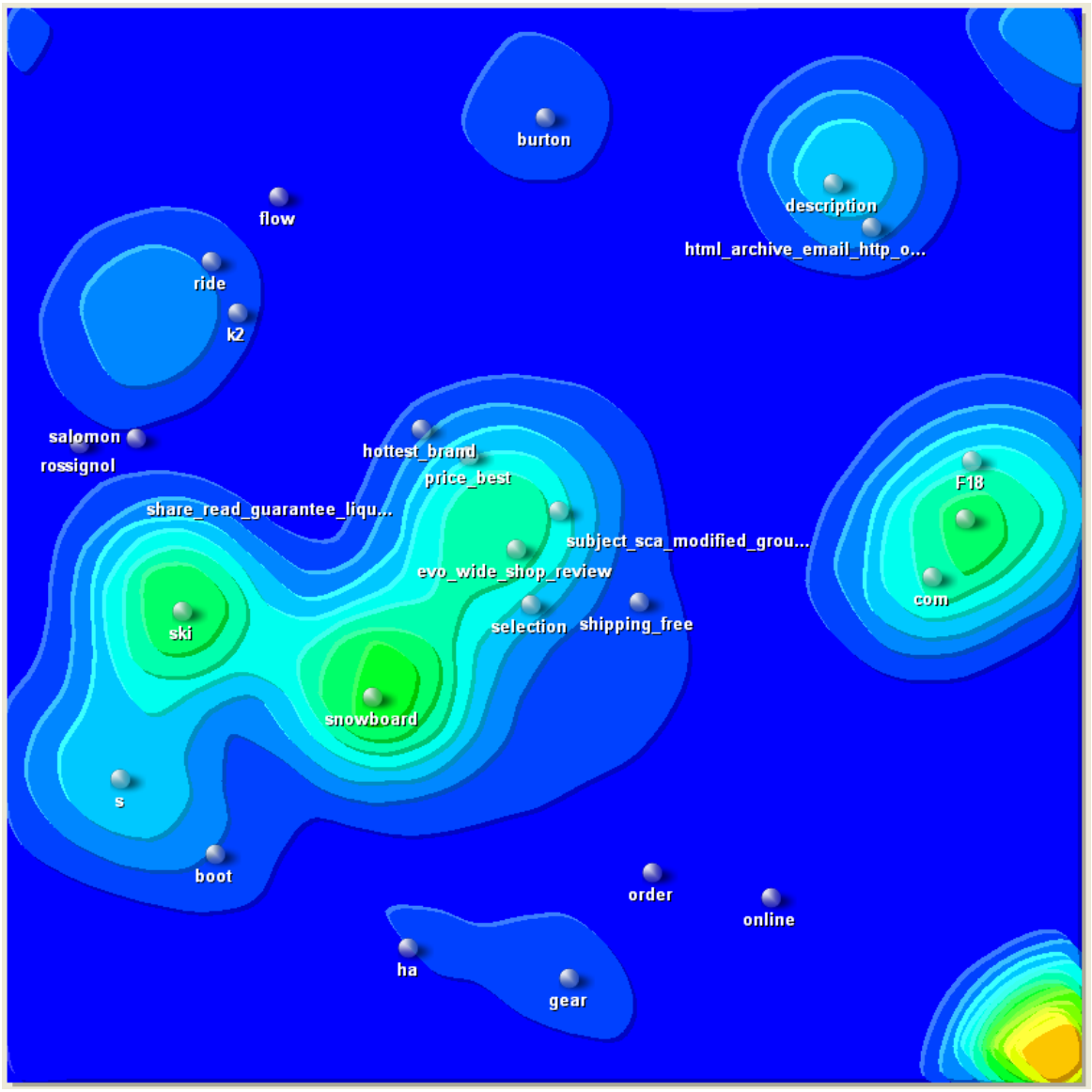
The last WIND map based on Live! Search data is the one computed using hierarchical clustering (Picture 23).



Picture 23

Search string	bindings
Date and time of the search	4th March 2009, 16:30
Clustering algorithm	hierarchical
Distance function	cosine
Height function	log
Clusters calculation	0.2s

As seen on Picture 23 there are still 2 main clusters (ski and snowboard) around which there are related clusters. In the north-eastern corner there are IT-related clusters. STORM map of the sam data set is in Picture 24.



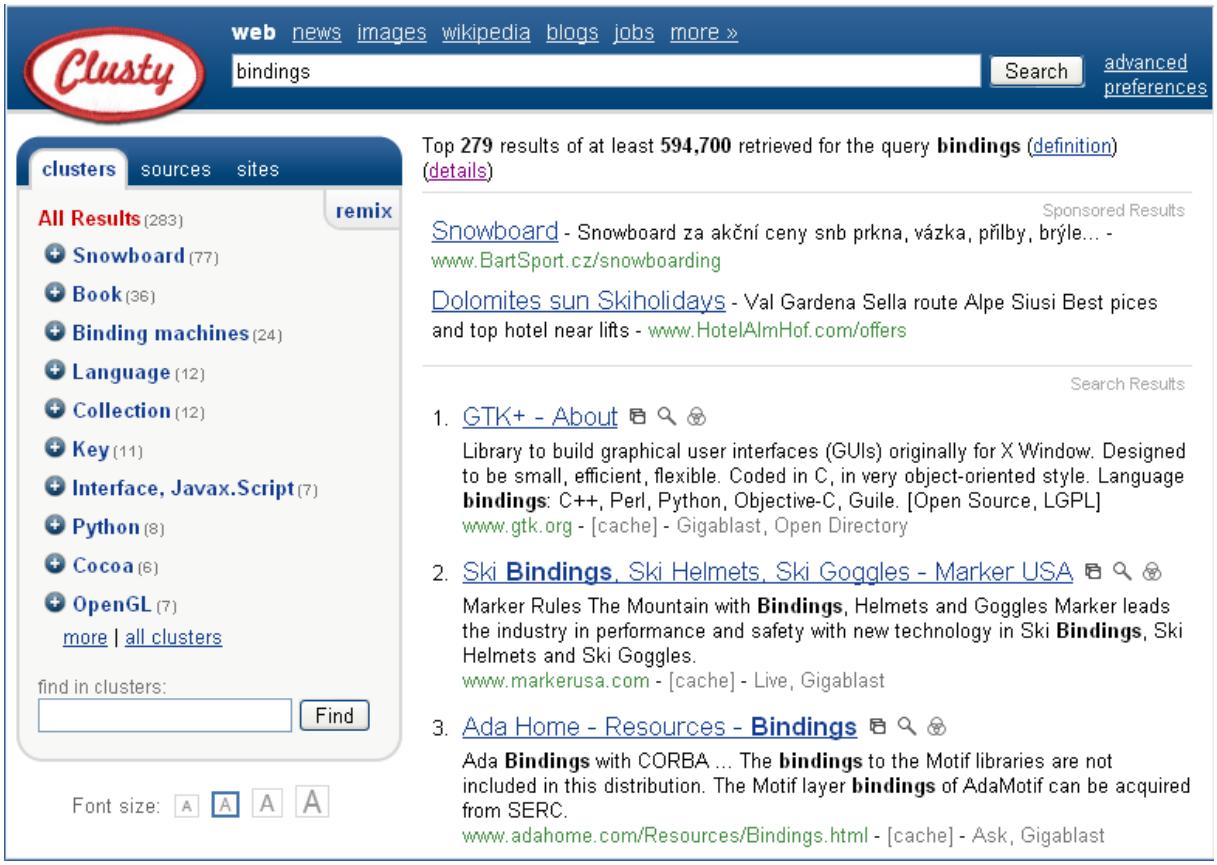
Picture 24

Search string	bindings
Date and time of the search	4th March 2009, 16:30
Clustering algorithm	hierarchical
Distance function	cosine
Clusters calculation	0.2s

Since in case of STORM maps the documents are placed into the map according to the relevancies to the keyword clusters there's problem with documents irrelevant which are not relevant to any keyword cluster (doesn't contain any of the most commonly used keywords in the result set). In Picture 24 there is a noname cluster in south-eastern corner (corner which has the greatest distance from the nearest keyword cluster) containing such documents. The same problem may be observed in Picture 27 (south-western corner). The

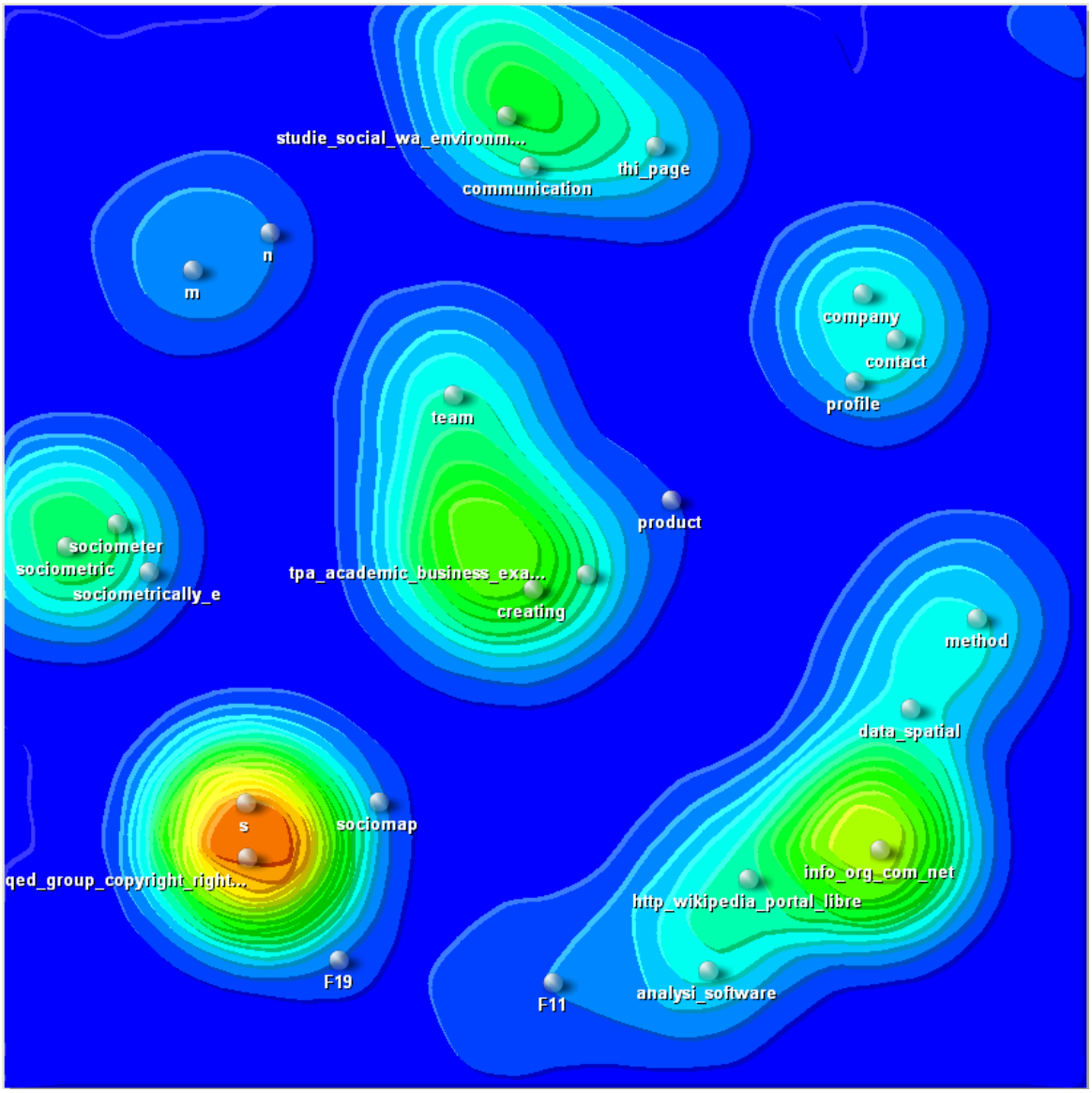
remedy of this problem would be to take into account more keywords in order to eliminate the number of documents with no relevance to any keyword cluster.

In order to compare the results with more traditional search engine clusty result set is in Picture 25. Apparently Clusty takes into account another metric when determining the relevance of keyword to a document (Google page rank). There are much more IT-related documents and e.g. cluster about skiing is missing at all. This fact tells that these 2 methods are different but doesn't tell which one is better.



Picture 25

Search result of search string "sociomapping" from Live! Search is visualized using STORM map in Picture 26.



Picture 26

Search string	sociomapping
Date and time of the search	7th April 2009, 22:30
Clustering algorithm	k-means
Distance function	cosine
Clusters calculation	0.5s

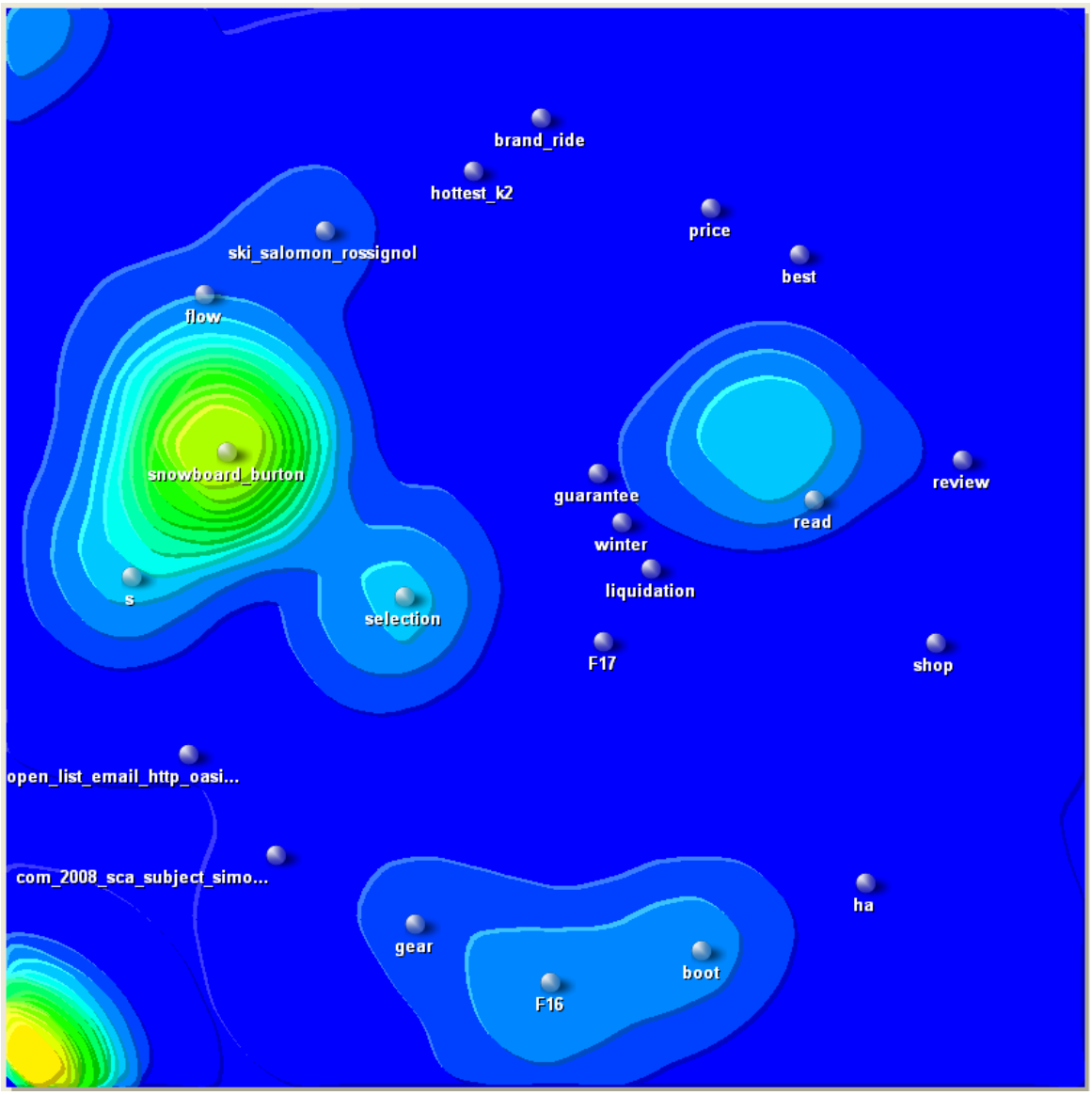
Although there are 25 keyword clusters the documents have been gathered into 7 storm clusters[1]. If we perform statistical testing on the south-western storm cluster (qed, group, copyright etc.) we obtain above average concentration of words: sociomap, qed, group,

[1] Hereafter storm cluster is used as denotation of "hill" on STORM map terrain

copyright, right, webmaster, registered, product, social. This cluster may therefore be interpreted as cluster of QED Group's[1] internal websites. The center storm cluster (team, tpa etc.) contains higher than average concentration of words: sociometrically, contact, software, info etc. This is a cluster of websites related to Team Profile Analyzer (TPA). It is software with purpose of sociometric analysis which uses sociomapping. Storm cluster in north eastern part of the map (company, contact, profile) contains above standard concentration of documents with words product, contact, company which tells that this cluster contains presentation of QED Group. Other storm clusters' interpretation is not so straightful.

STORM map displayed in Picture 27 of search result of search string "bindings" contains less storm clusters than Picture 26. This allows to assign a meaning to each storm cluster.

[1] QED Group, a.s. is company which developed sociomapping



Picture 27

Search string	bindings
Date and time of the search	8th April 2009, 19:00
Clustering algorithm	k-means
Distance function	cosine
Clusters calculation	0.5s

The high storm cluster in south western corner contains IT related documents toghether with snowboard-related ones. Unfortunatelly IT-related documents have too different words to create above standard concentration of their words and therefore only snowboard-related keywords are abnormally concentrated here. Snowboard-related storm cluster is also the one in western part of the map (burton, snowbord) but it is fluently transformed to the skiing-related storm cluster (northern part of this storm cluster with

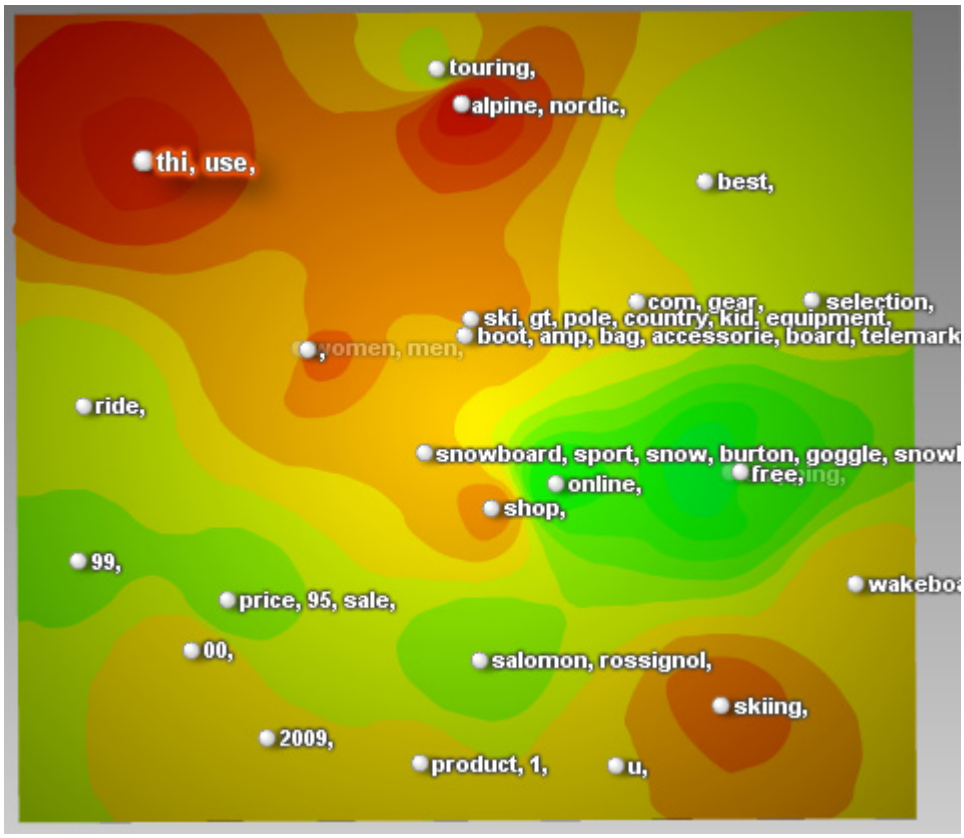
ski, rossignol and salomon keywords). A webshop selling only “selection” of all snowboard product creates the “selection” storm cluster almost in the center of the map. A different webshop selling snowboard equipment creates the storm cluster in the eastern part of the map. Mostly common documents about ski and snowboard bindings and boots (reviews, Wikipedia documents etc.) create the storm cluster in southern part of the map.

Since Live! Search returned the most suitable results for the maps creation most of the clustering algorithms and height functions were shown based on Live! Search results. In order to compare these results with other sources’ results the same combinations of search strings, clustering algorithms and height functions will be presented in Yahoo! And Google chapters.

5.2.3 Yahoo!

Yahoo!, one of the traditional search engines, provides API via AJAX (XML returned via HTTP based on an HTTP request). It can return all the documents in the result set but usually the result set is quite small (in comparison e.g. with Google result set), so the experiments were performed on approximately the same number of documents in the search result as Live! Search (approximately 1000).

The same search string (bindings) has been tried with Yahoo! as well. The result is displayed in Picture 28.



Picture 28

Search string	bindings
Date and time of the search	13th March 2009, 15:00
Clustering algorithm	k-means
Distance function	cosine
Height function	logarithmic
Clusters calculation	5.5s

It may be seen that snowboard cluster (center) is situated on the north from skiing area (clusters "salomon, rossignol" and "skiing"). Unfortunately northern cluster "alpine, nordic" is the first cluster in the list (see K-means) and the length of the snippets are not long enough, so the DVM is a sparse matrix (usually there are only few documents relevant to the keyword within cluster in the reduced DVM). This is the reason why this cluster contains so many documents (approximately 130 out of 1000) and contains IT-related websites as well as skiing-related websites. The solution to this problem would be providing additional information (e.g. *Idf* of the words which would distinguish the relevancy and therefore cluster membership).

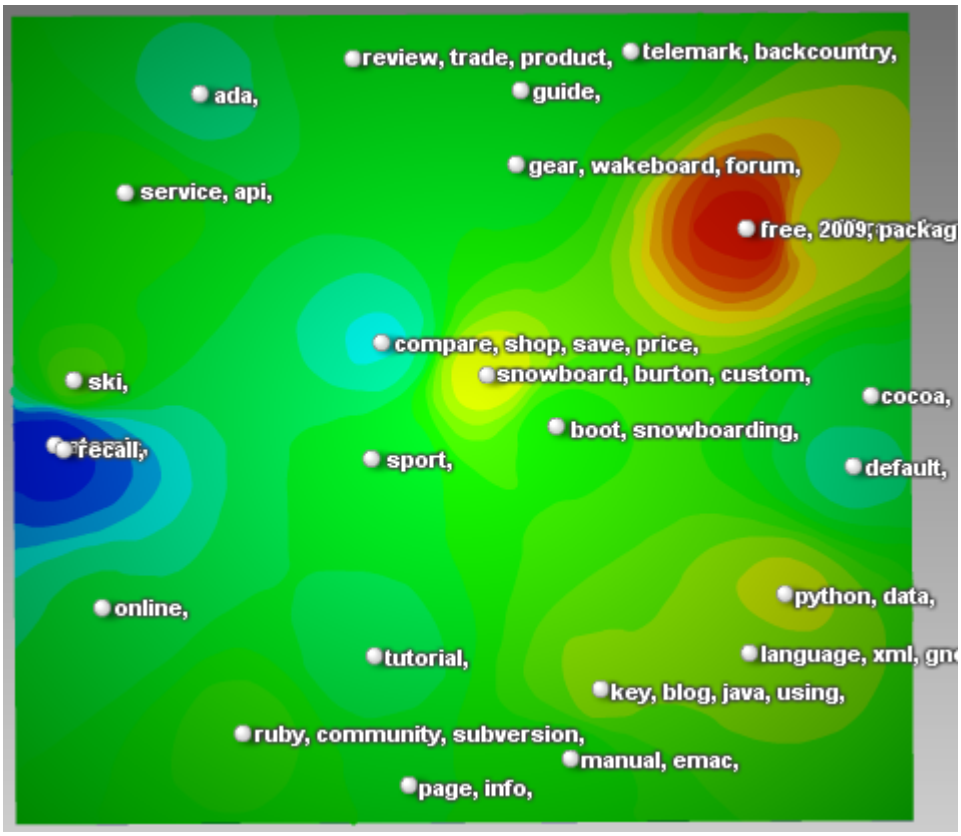
5.2.4 Google

The most widely used search engine doesn't provide any API for global search. It

provides search API for a single website only. In order to test search results from Google it was necessary to write a parser which decomposes the HTML site containing search result to items containing target URL, description and snippet. Snippets are (similarly to Live! and Yahoo!) used to construct DVM. Unlike Live! and Yahoo! Google inserts links to his own documents into the search result (e.g. from http://books.google.com). Since these documents don't have any snippet they weren't included into the resulting DVM.

Since Google has notion of such attempts to write a parser replacing API it bans IP addresses which perform several similar searches per a short time unit. Therefore a short pause between downloading the parts of the result set provided by Google has to be performed and loading search result from Google is the slowest of all data sources.

In order to compare WIND map based on data from Live! search, Yahoo! and Google the same search string ("bindings"). Since the search result is similar to the one Clusty is based on the keyword clusters should be similar to the ones in Picture 25.



Picture 29

Search string	bindings
Date and time of the search	25th March 2009, 14:30
Clustering algorithm	k-means
Distance function	cosine
Height function	logarithmic
Clusters calculation	1.5s

As displayed in Picture 29 the cluster with highest number of documents is the cluster with no specific keywords (2009, free, shipping ...). Problem of too low number of relevant keywords and absence of keywords' *Idf* is faced in this search result (the cluster with highest number of documents is the first one). This problem is described in experiment displayed in Picture 28. If we disregard this problem the clusters were placed in WIND map in a way that may be interpreted quite easily. In the north-east corner of the map there are clusters related to telemark and cross-country skiing ("telemark" or "guide"), in the center of the map there are clusters related to snowboarding ("snowboarding", "boot", "Forum"[1], "Burton", "Custom"[1]) are in the center of the map.

[1] Forum is a snowboard brand

Google returns only few skiing-related website (as displayed in Picture 25) there is only 1 skiing-related cluster with an average height ("ski") by the western side of the map. The IT-related documents are in the southern and south-eastern part of the map.

5.2.5 Reuters Database

As described in [1] Reuters Corpus Volume 1 (hereafter RCV1) is a collection of about 800,000 articles published by Reuters during years 1996 and 1997 manually categorized into more than 100 hierarchical categories where a single article belongs to 1 or more categories. RCV1 is used in this work as a benchmark for clustering algorithms used in this work. Articles from RCV1 CD1 (more than 470,000) categorized into classes from 8/20/1996 to 3/31/1997 were inserted into MySQL database with Lucene.NET fulltext index built above it. The queries were performed using Lucene.NET index while data stored in MySQL were used to obtain classes categorization and text of the article. Since this data source is static the value of *Idf* is known and it is calculated by Lucene.NET.

The clustering results were compared using entropy and F-Measure [12, 18]. Entropy of result of clustering algorithm executed on documents categorized into classes is defined by Formula 22.

$$E = \sum_j \frac{n_j}{n} E_j$$

Formula 22

In Formula 22, Formula 23 the following notation is used:

n	Number of all documents
n_j	Number of documents in j^{th} cluster
n_{ij}	Number of documents which belong to i^{th} class and are in j^{th} cluster
n_i	Number of documents in i^{th} class

[1] Burton Custom is quite popular snowboard binding

E_j Entropy of j^{th} cluster defined by Formula 23

p_{ij} Precision $p_{ij} = n_{ij} / n_i$

r_{ij} Recall $r_{ij} = n_{ij} / n_j$

$$E_j = -\sum_i p_{ij} \log_2 p_{ij}$$

Formula 23

Entropy is a degree to which each cluster contains documents of single class (less is better). Since one document usually belongs to more classes and it is impossible (based on the data contained in RCV1) to distinguish the most important category the document was consiedered to be part of results the same number of times as number of categories where it belongs to. E.g. if 2 documents were in j^{th} cluster and each of them belonged to 3 classes (classes 1, 2 and 3) then $n_j = 6$ and $n_{ij} = 2$ ($i = 1 .. 3$).

F-Measure is defined by Formula 24 and Formula 25.

$$F - measure = \sum_i \frac{n_i}{n} \max\{F(i, j)\}$$

Formula 24

$$F(i, j) = \frac{2 r_{ij} p_{ij}}{r_{ij} + p_{ij}}$$

Formula 25

F-Measure is an extent of which clusters contain documents of only one class and contain all documents of that class. This means that F-Measure unlike entropy takes into account whether all documents from one class are part of only one cluster.

Several search result sets have been used to create keyword (and document) clusters using different algorithms and keyword vector similarities. Entropies and F-Measures of these clustering results have been calculated and the result may be seen in the following charts. Vertical axis is evaluated by the corresponding values (F-Measure, or entropy) and horizontal axis is evaluated by search strings according to which the search result has been retrieved.

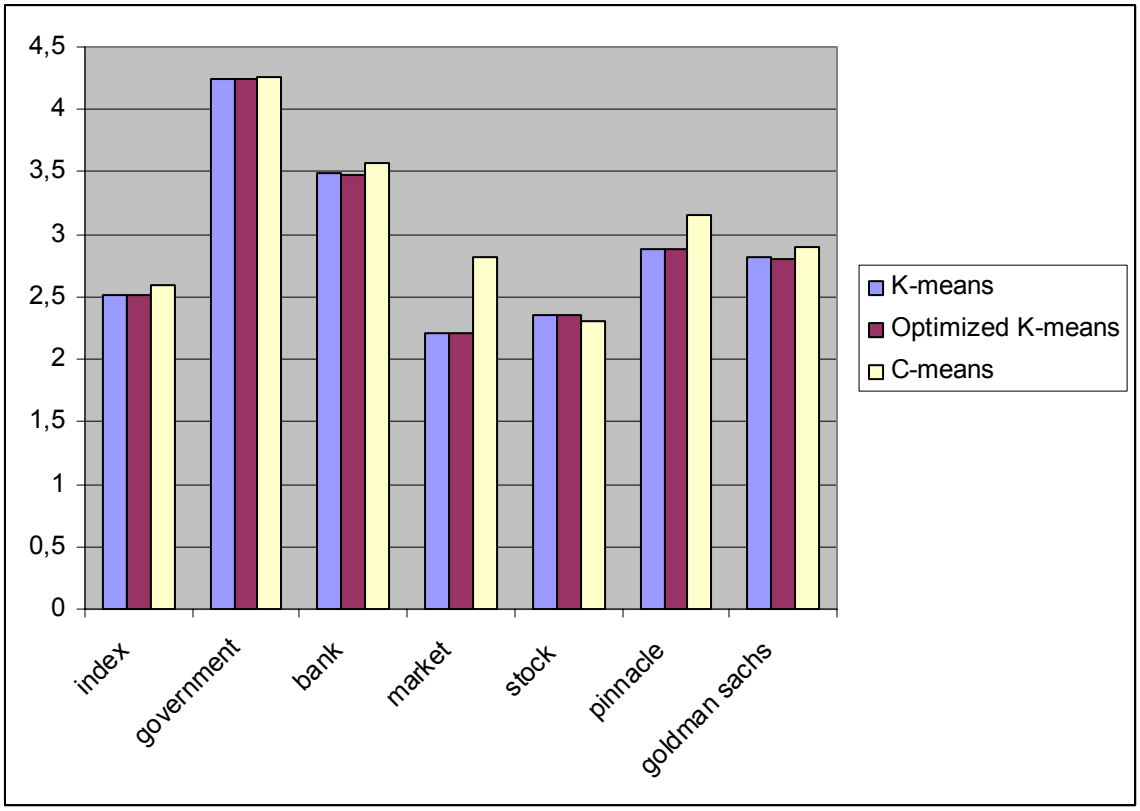


Chart 1

Chart 1 contains entropies of clustering result sets created using Euclidean keyword vector similarity. K-means and optimized K-means algorithms produce very similar results and therefore their entropies are usually very similar. C-means algorithm is a fuzzy based algorithm and therefore the result is usually different from the 2 previous algorithms. Entropies of C-means results are usually higher. According to this comparison metric K-means is considered to be better than C-means.

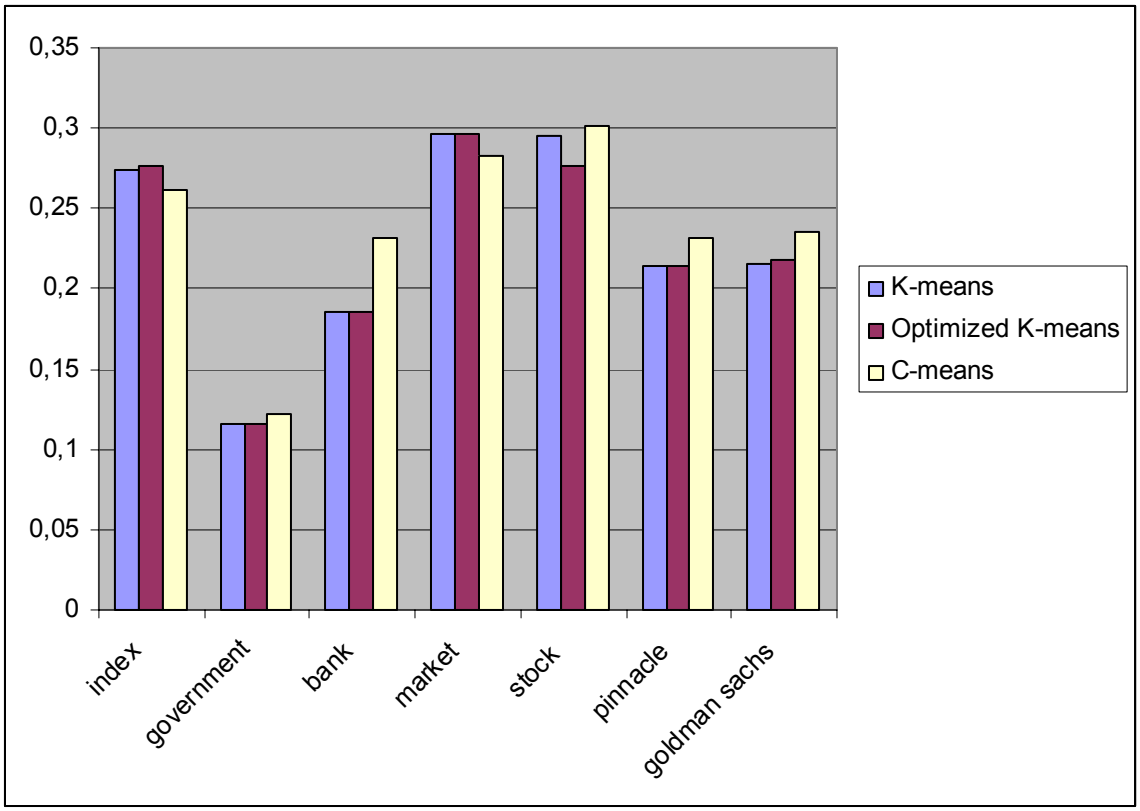


Chart 2

In Chart 2 we may see F-Measures of the same clustering results Chart 1. In most cases F-Measure of K-means result is similar to F-Measure of optimized K-means. Similarly to Entropy F-Measure confirms that C-means algorithm is worse than K-means or optimized K-means. After change of similarity metrics we obtain the following results (Entropy in Chart 3 and F-Measure in Chart 4).

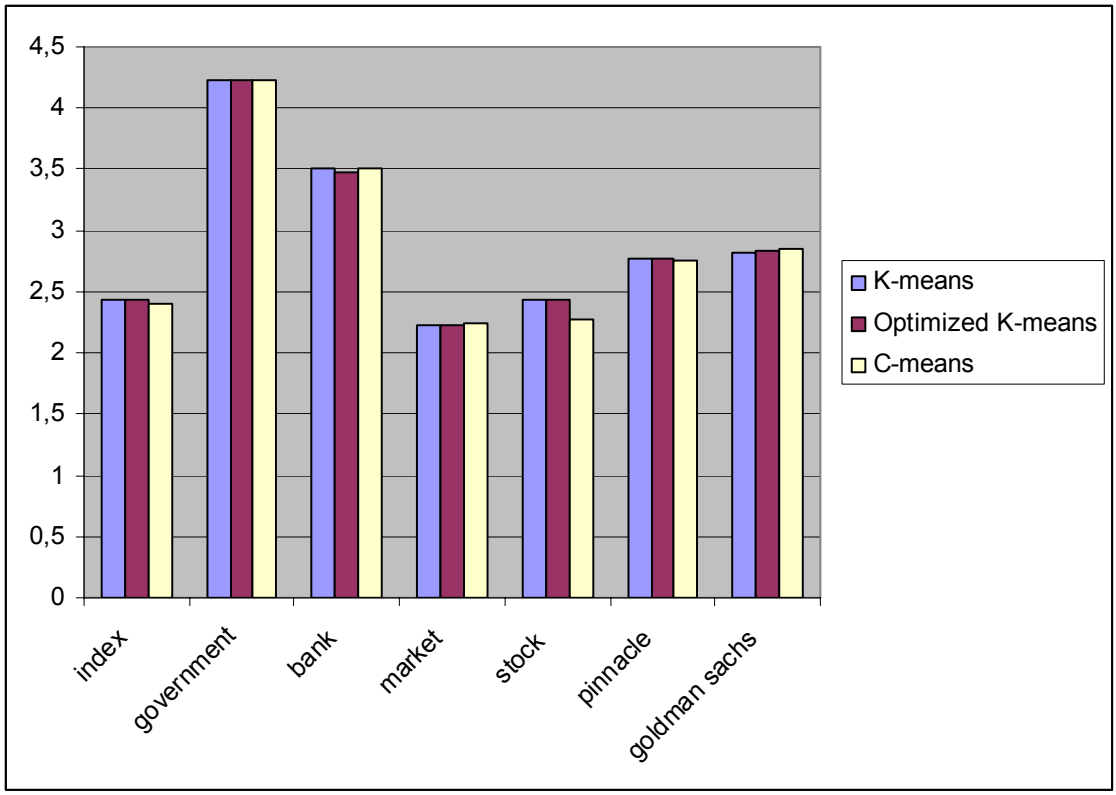


Chart 3

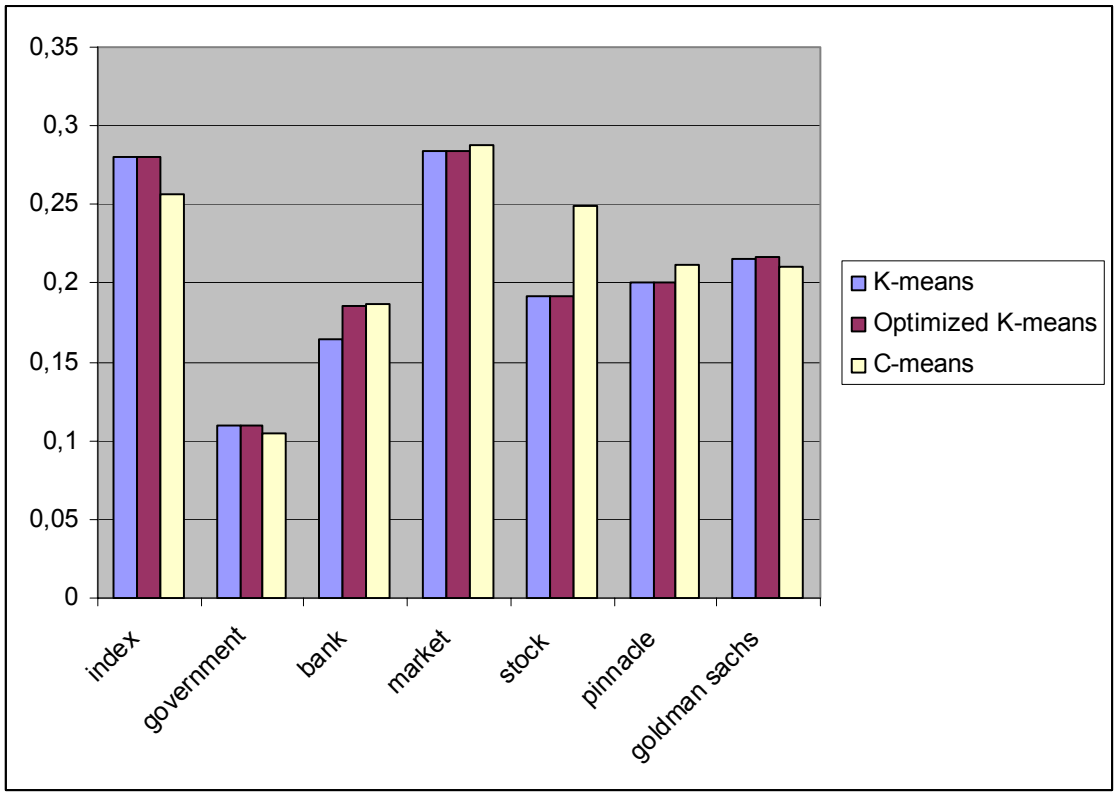


Chart 4

Differences among Entropy and F-Measure values for the algorithms are not as big as in case of Euclidean distance similarity. Since the user should get result not only precisely but fast the last value which was measured is time.

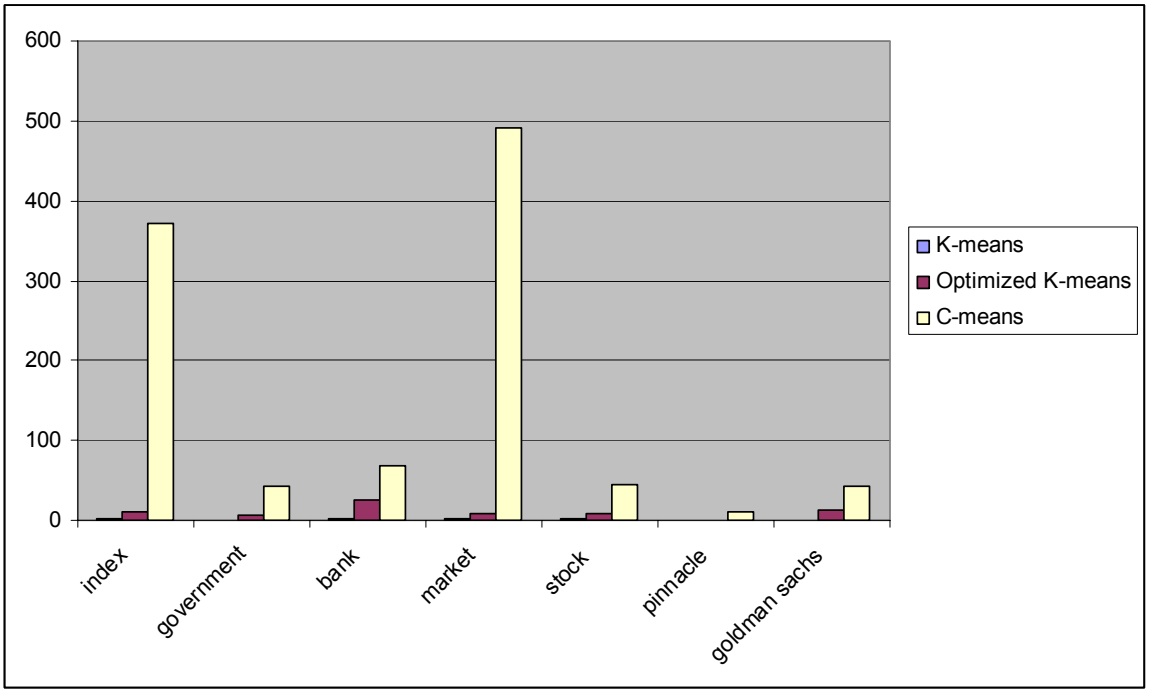


Chart 5

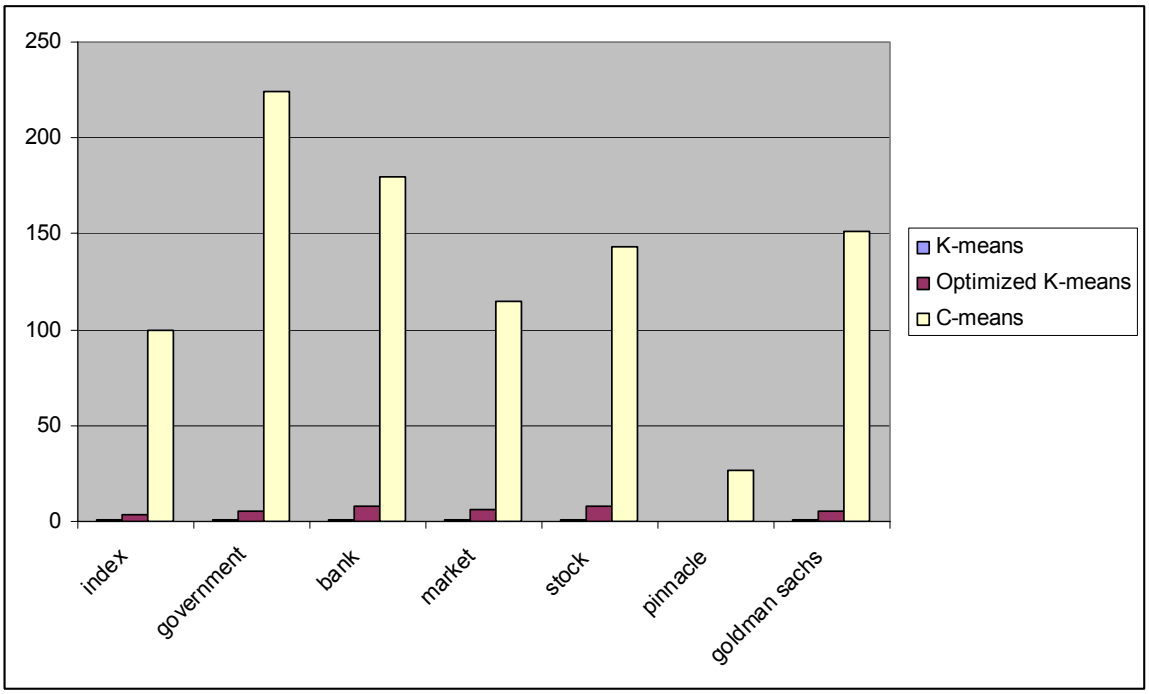
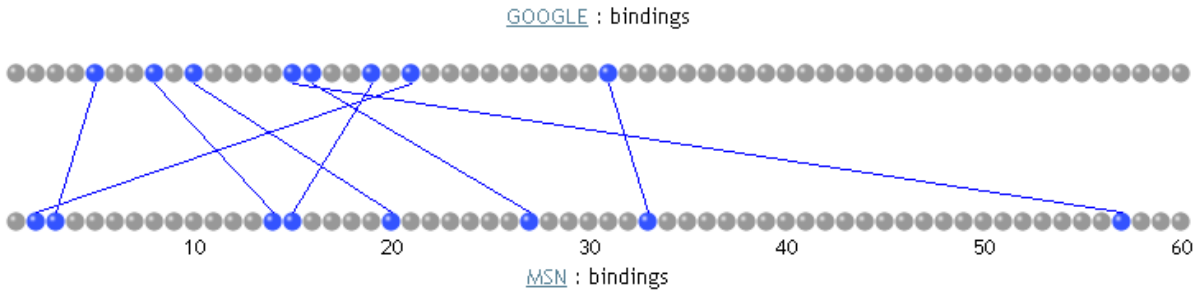


Chart 6

Chart 5 contains times needed to perform clustering using various clustering algorithms and Euclidean similarity function while Chart 6 contains time needed to perform clustering on the same result sets using cosine similarity function. Shortest times were achieved with K-means and longest with C-means. The experiments above presented that K-means is the most suitable algorithm for clustering keywords to clusters.

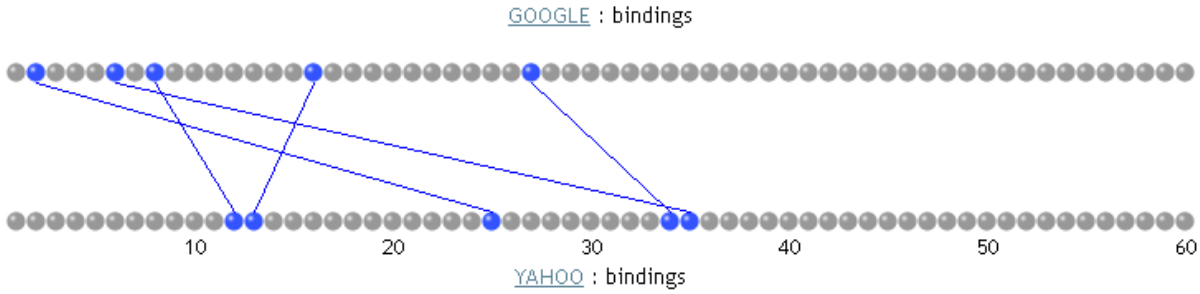
5.3 Search Results Comment

In order to illustrate differences in the engines which were used to search in web (dynamic document space) see Picture 30. Each bullet in this picture is one of the websites within the search result (first 60 documents) for word bindings provided by the corresponding search engine (Google is the upper row and Live! Search is the lower row). The documents which appear in both result sets are blue and are connected with blue line. There are only 8 documents which appear within both search results. This explains why the 2 maps based on Live! Search and Google are so different.



Picture 30

Results from Google and Yahoo! are even more dissimilar than the ones above as displayed in Picture 31.



Picture 31

6 Conclusion

The aim of this work was to find a suitable way of presentation of search result returned by various document search engines. We've used both locally stored document spaces (database of approximately 70,000 e-mail messages stored in Oracle database in chapter Enron Mail Database and database approximately of 450,000 Reuters articles categorized into about 100 categories in chapter Reuters Database) and web (Live! Search in chapter Live! Search, Yahoo! in chapter Yahoo! and Google search engine in chapter Google).

Search results of all of these search sources were transformed into document vector model and clustered to keyword clusters according to keyword occurrences in the documents (see DVM Reduction and Keyword Clustering). Keyword similarity was determined by 3 various keyword vector similarities: Euclidean, cosine and Manhattan (as defined in Keyword Clustering). According to Experiments the best results were achieved using cosine similarity. For purpose of clustering K-means, optimized K-means, C-means and hierarchical clustering algorithms were used (described in Keyword Clustering). The results of these clustering algorithms were compared visually (using the resulting WIND and in several cases STORM map) in chapter Experiments and (see Reuters Database) using F-Measure and Entropy.

The best results were achieved by K-means algorithm with cosine definition of similarity function. In most cases both WIND and STORM maps had a reasonable interpretation in real world.

7 Search Visualization Tool User's Manual

7.1 Initial configuration

In order Search Visualization Tool to work with static document spaces (Enron mail database and Reuters article database) database and connection to the database has to be configured properly. You don't need to go through any initial configuration steps if you want to use Search Visualization Tool only with web search engines. In such case only direct HTTP connection to the corresponding servers is required.

7.1.1 Database and Data Creation

MySQL database dump file is stored in archive \Data\rcv1.zip. In order to create database with data just execute the SQL script. In order to create Enron database and data in it use PlSql developer to restore file \Data\enron.pde into Oracle 11 database.

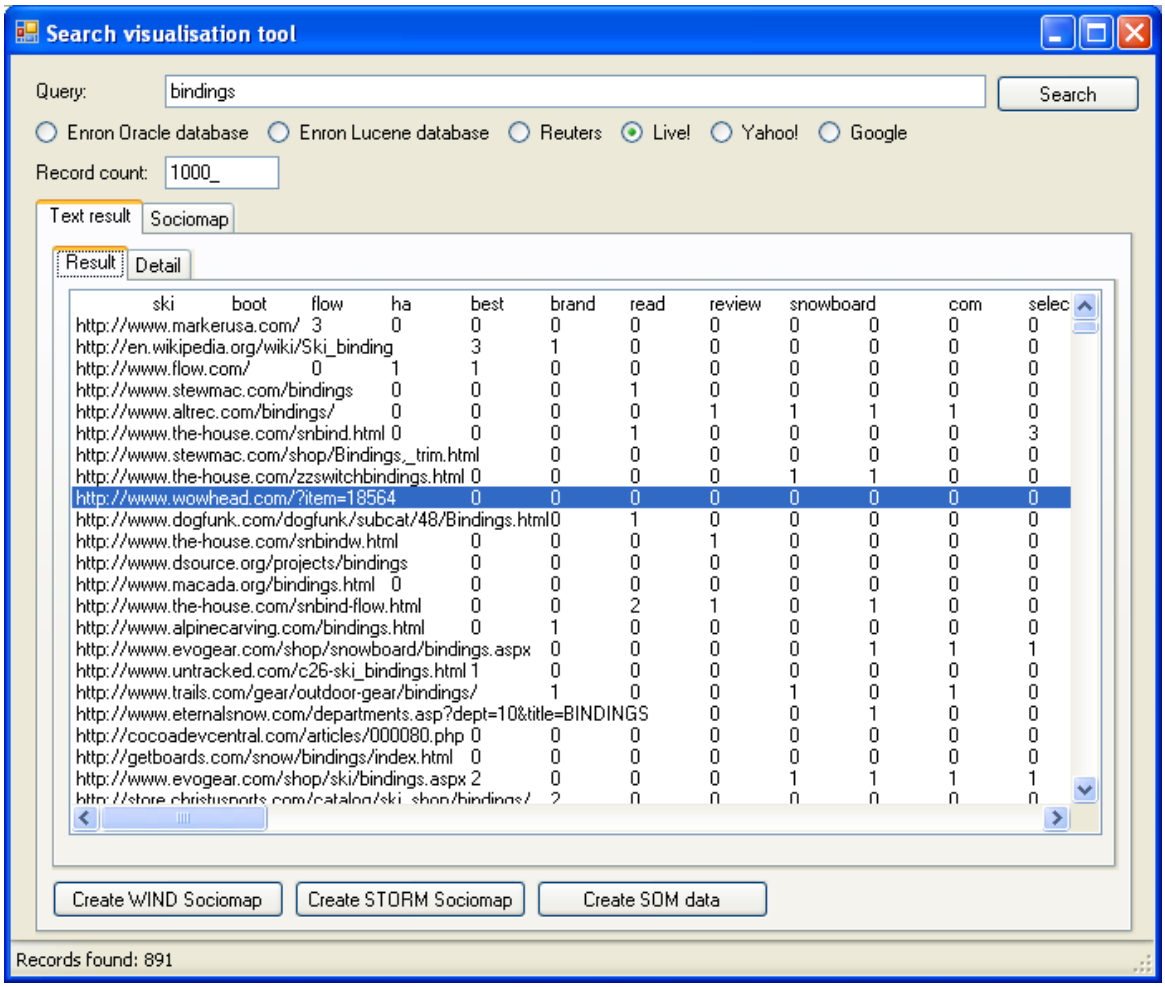
Connection strings to both databases have to be specified in \VisualSearch\VisualSearch.exe.config as values of corresponding configuration items ("ReutersMySqlConnection" for MySQL database with RCV stored and „EnronOracleConnection" for Oracle database with Enron mail database). Both connection strings are standard .NET connection strings. An example of such connection string is already written in the configuration file.

Lucene indexes for both databases have to be copied to a directory specified in \VisualSearch\VisualSearch.exe.config. Write the path to the directory where Enron index is stored (this index is stored in \data\enron\ on the attached media) to the value of "EnronIndexDirectory" and path to the directory where Reuters index is stored (\data\rcv1\ on the attached media).

7.2 Usage

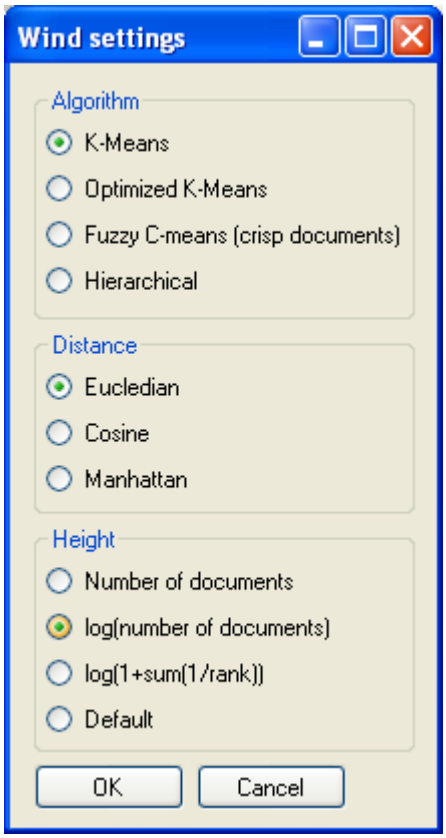
Usage of this tool is quite straightforward. After executing it the user sees the main (and in fact the only) window (Picture 32). User can select data source to search in using the radio buttons right under the Query text box Keep in mind that static document spaces (the first three ones) have to be configured in application settings file – see Initial configuration). Only direct internet connection is required in order the search in dynamic document spaces to be functional. Then the search string is to be written into the Query

text box. Since most all the dynamic document space search engines are limited to 1000 results 1000 is the default limit of documents returned (Record count text box). Obviously button Search causes the tool to initiate searchin using the selected source.



Picture 32

After the search is performed it is possible to create clusters using Create WIND Sociomap button which causes the clustering settings dialog to be displayed (Picture 33).



Picture 33

This dialog contains 3 radio groups allowing the user to select algorithm used for creating clusters, keyword distances (see Keyword Clustering) and WIND map terrain height (see WIND Map Creation). „Number of documents“ corresponds to height defined by Formula 15, “log(number of documents)” to Formula 16 and “log(1+sum(1/rank))” to Formula 17. “Default” height is the default height function given by WIND map which is calculated according to relation of the keyword cluster to other keyword clusters (the keyword cluster which is the most similar to the others will be on the highest point of the map).

Create STORM sociomap fills data needed for STORM sociomap creation into the text box on the Sociomap tab. These data may be pasted to Excel, saved and loaded by Rainbow Sociomapping software (not part of the attached media). Create SOM data fills data need for Self Organizing map into the text box on the Sociomap tab. These data may be posted to the input file of SOM software [9].

It is possible to display the full text of the document (if it is accessible) by doubleclicking on the corresponding line of Text result / Search text box. HTML documents are displayed in HTML.

Resources

[1] David D. Lewis, Yiming Yang, Tony G. Rose, Fan Li. RCV1: A New Benchmark Collection for Text Categorization Research The Journal of Machine Learning Research archive, Volume 5, 361 - 397, 2004 ISSN:1533-7928

[2] Cyril Höschl (2006): Sociomaps Visualisation, bachelor thesis (CZ) http://www.hoschl.cz/cyril/bakalarka/errata.pdf

[3] Cyril Höschl, Jan Hudeček, Petr Jenček, Jan Šebesta (2009): Mail Archiver, software project (CZ) http://urtax.ms.mff.cuni.cz/prk/zadani/mailar.pdf

[4] Dušan Húsek, Hana Řezanková, Václav Snášel: Shlukování a textové dokumenty (CZ) http://www.statspol.cz/robust/robust2004/husek.pdf

[5] Salton G., Buckley C. (1988). Term weighting approaches in automatic text retrieval. Information Processing and Management 24, 5, 513 – 523.

[6] Berry M.W. (editor). (2004). Survey of text mining: clustering, classification and retrieval. Springer-Verlag, New York.

[7] Lucene.Net documentation http://incubator.apache.org/lucene.net/

[8] Asghar Panahy (2005): Google search parser http://www.codeproject.com/KB/recipes/googleparser.aspx

[9] Ultsch, A., Moerchen, F.: ESOM-Maps: tools for clustering, visualization, and classification with Emergent SOM, Technical Report Dept. of Mathematics and Computer Science, University of Marburg, Germany, No. 46, (2005) http://databionic-esom.sourceforge.net/

[10] A. Hotho and S. Staab, A. Maedche: Ontology-based Text Clustering http://www-2.cs.cmu.edu/~mccallum/textbeyond/papers/hotho.pdf

[11] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, Prabhakar Raghavan (1998): Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications http://www.cs.cornell.edu/johannes/papers/1998/sigmod1998-clique.pdf

[12] Daling Wang, Ge Yu, Yubin Bao and Meng Zhang: An Optimized K-Means Algorithm of Reducing Cluster Intra-dissimilarity for Document of Clustering Advances in Web-Age Information Management 6th International Conference, WAIM 2005, Hangzhou, China, October 11 – 13, 785-790, 2005 ISBN:978-3-540-29227-2

[13] Oracle Data Provider for .NET documentation http://www.oracle.com/technology/tech/windows/odpnet/index.html [14] Tuevo Kohonen: Self-organizing Maps, 2001 ISBN: 3540679219

[14] G. Salton, A. Wong, C. S. Yang (1975): A Vector Space Model for Automatic Indexing http://www.cs.uiuc.edu/class/fa05/cs511/Spring05/other_papers/p613-salton.pdf

[15] Raul Valdes-Perez (2008): Searching To Surpass Web Searching? http://searchdoneright.com/2008/02/enterprise-searching-to-surpass-web-searching/

[16] Bryan Klimt, Yiming Yang (2004): Introducing the Enron Corpus http://www.cs.cmu.edu/~enron/

Appendices

Word Stop List

a, about, above, across, after, afterwards, again, against, all, almost, alone, along, already, also, although, always, am, among, amongst, amoungst, amount, an, and, another, any, anyhow, anyone, anything, anyway, anywhere, are, around, as, at, back, be, became, because, become, becomes, becoming, been, before, beforehand, behind, being, below, beside, besides, between, beyond, bill, both, bottom, but, by, call, can, cannot, cant, co, computer, con, could, couldnt, cry, de, describe, detail, do, done, down, due, during, each, eg, eight, either, eleven, else, elsewhere, empty, enough, etc, even, ever, every, everyone, everything, everywhere, except, few, fifteen, fify, fill, find, fire, first, five, for, former, formerly, forty, found, four, from, front, full, further, get, give, go, had, has, hasnt, have, he, hence, her, here, hereafter, hereby, herein, hereupon, hers, herself, him, himself, his, how, however, hundred, i, ie, if, in, inc, indeed, interest, into, is, it, its, itself, keep, last, latter, latterly, least, less, ltd, made, many, may, me, meanwhile, might, mill, mine, more, moreover, most, mostly move, much, must, my, myself, name, namely, neither, never, nevertheless, next, nine, no, nobody, none, noone, nor, not, nothing, now, nowhere, of, off, often, on, once, one, only, onto, or, other, others, otherwise, our, ours, ourselves, out, over, own, part, per, perhaps, please, put, rather, re, same, see, seem, seemed, seeming, seems, serious, several, she, should, show, side, since, sincere, six, sixty, so, some, somehow, someone, something, sometime, sometimes, somewhere, still, such, system, take, ten, than, that, the, their, them, themselves, then, thence, there, thereafter, thereby, therefore, therein, thereupon, these, they, thick, thin, third, this, those, though, three, through, throughout, thru, thus, to, together, too, top, toward, towards, twelve, twenty, two, un, under, until, up, upon, us, very, via, was, we, well, were, what, whatever, when, whence, whenever, where, whereafter, whereas, whereby, wherein, whereupon, wherever, whether, which, while, whither, who, whoever, whole, whom, whose, why, will, with, within, without, would, yet, you, your, yours, yourself, yourselves, new, com, thanks, cc, pm, http, www, subject, enron