Soft Operators for Exploring Information Granules of Web Search Results

Gloria Bordogna
Institute for the Dynamics of Environmental Processes
National Research Council of Italy
Via Pasubio 5, 24044 Dalmine (BG), Italy
gloria.bordogna@idpa.cnr.it

Giuseppe Psaila
Faculty of Engineering
University of Bergamo
Viale Marconi 5, 24044 Dalmine (BG), Italy
psaila@unibg.it

***Abstract* - The paper defines some soft aggregation operators for combining results of Web searches, organized into information granules of distinct resolution. We propose to use them to perform personalized explorations of the contents retrieved by distinct queries to possibly distinct search engines on the Internet. These operators exploit the contents of the result lists retrieved by search engines to discover shared and correlated contents between web pages. We present their application within the meta-search system *Matrioshka* and discuss their semantics and utility.**

I. INTRODUCTION

The motivation of this work is to optimize the search processes on the Internet, actually consisting of multiple query reformulations [9], by providing some means for analyzing the contents already retrieved by a query or a set of queries. The objective is improving the potential exploitation and comprehension of the contents retrieved by multiple Web searches to classic search engines. This is pursued by offering consultation indications alternative to the usual ranked list.

It is well known that one drawbacks of the way users search information on the Internet by submitting queries to search engines is that they formulate vague requests, consisting of at most three terms [9] [10], and as a result they are overloaded with huge amounts of web pages dealing with diversified topics, that they rarely analyze below the second page of results. As a result of multiple searches with the same target, possibly submitted to several search engines, one needs to filter out the relevant documents among those already retrieved. Specifically, we provide users with some practical means for exploring the overall set of results of several web searches, to filter out their shared documents, shared contents, correlated documents and correlated contents. The highlighting of hidden relationships between documents retrieved by distinct queries, can help understanding the topics dealt with in the documents text, and thus gives new hints on their relevance [13][14][17]. In order to make this task feasible, it is not possible to exploit the full content of the documents retrieved by the query (that would require an HTTP access to the web pages in the reported result lists); on the contrary, our solution extracts the necessary information from within the texts (titles and snippets) reported in the result lists provided by the search engines [5][13][15]. The operators that we define are derived from relational algebra and are defined based on fuzzy set theory [18].

In the paper, we first introduce the information granules that are the objects that we combine by the operators. Then, we define the operators used to combine the information granules to identify their shared documents and contents. Finally, an example of application of the operators within the meta-search system *Matrioshka* and its results are discussed [2][3].

II. MULTI-GRANULAR WEB SEARCH RESULTS

In this section, we summarize the data model, first introduced in [1], i.e., the multi-granular organization of the results of web searches, which constitute the basic bricks of information that the soft operators can combine.

We start by considering a query q submitted to a search engine; its result is a ranked list of web pages; an item represents a retrieved web page referred in the ranked list.

Definition 1: Item

An item i is the finest granule of information that the operators can manipulate. It represents (an instance of) a document retrieved by a web search. It is described by the following tuple of attributes:

$$i : <Uri_i, Title_i, Snippet_i, Bag_i, Irank_i>$$

Uri_i is the Uniform Resource Identifier of the ranked web document;

$Title_i$ and $Snippet_i$ are, respectively, the document title and snippet;

Bag_i is a bag of strings (single terms), each one weighted with a score in [0,1], expressing the significance of the string in representing the contents of the item; in Section 3, the procedure for the generation of Bag_i is detailed;

finally, $Irank_i$ is a score (in the range [0, 1]) that expresses the estimated relevance of the retrieved document w.r.t. the query and is computed as detailed in Section 3 (this is the reason why sometimes an item is named ranked item).

Definition 2: Cluster

A cluster c corresponds to an information granule composed of items, having a rank. It is defined by the tuple:

$$c : <Label_c, Content_c, Rankings_c, Crank_c>.$$

$Label_c$ is a set of terms that semantically synthesize the main content of the cluster;

$Content_c$ is the set of items associated with the cluster;

$Rankings_c$ is a pool of values that measure (in the range [0, 1]) different properties of clusters, while $Crank_c$ is a user-defined combination of measures in $Rankings_c$ (measures in $Rankings_c$ and the way $Crank_c$ is computed was presented in [3]).
Notice that $Label_c$, $Rankings_c$ and $Crank_c$ are computed as functions of $Content_c$. This ensures that whatever the cluster c was generated, if two clusters have the same content, they also have the same label, and rankings.
A cluster can be generated by applying an operator combining two other clusters or by a clustering operation applied to a coarser information granule, a group, defined hereafter.

Definition 3: Group
A group g is the main element of the data model. It is the coarsest information granule composed of ordered clusters. It is described by the pair:

$$g : <Label_g, Clusters_g>$$

$Clusters_g$ is the set of clusters belonging to the group.
A group can be obtained from the ranked list of documents retrieved by a search engine, or can be generated by an operator working on groups.
$Label_g$ is the label of the group: it is a set of terms that semantically synthesize the main contents of the group.
$Label_g$ is generated based on a function of all the items in all clusters of the group. Specifically, for a group generated by a query to a search engine, it is the text of the query submitted to the search engine; otherwise it is the title of the ranked item most representative of the group, as defined in [1].

III. Definition of the Soft Operators for Combining Information Granules of Search Results

Before we define the soft operators, we need to introduce some preliminary operations necessary to generate the information granules introduced in the previous section.

A. Generation of items

An item i is generated by parsing the list of ranked results returned by a search engine that evaluated a query.
Notice that the same web page retrieved by different search engines (or by different queries) may be represented by distinct items in distinct result lists. In fact, in this case the document is uniquely identified by the same Uri_i, while it may have distinct $Title_i$, $Snippet_i$, Bag_i and $Irank_i$.
We compute $Irank_i$ as a function of the position $Pos(i)$ of the item in the query result list, defined as follows:

$$\mathrm{Irank}_i = \frac{N - Pos(i) + 1}{N} \quad (1)$$

Where N is the number of ranked items in the result list. Thus, it is independent of the actual relevance score computed by the search engine.
On the other side, distinct web pages have distinct Uris but may share the same or similar titles and snippets, because they are indeed duplicated documents at distinct web sites retrieved by the same query.
A bag of weighted strings is defined by the fuzzy set:

$$Bag_i = \{ s_1/w_{s1}, \cdots, s_n/w_{sn} \},$$

with s being a string and $w_s \in [0,1]$. Bag_i represents the main contents of an item i.
The strings in Bag_i are obtained by performing lexicographic analysis of the Uri_i, $Title_i$ and $Snippet_i$ of item i by applying *Lucene* functions [12]: stop-words are removed, words stemming is applied, single terms are expanded with associated terms by using *Wordnet* [6]; finally, all the selected single terms are included in Bag_i. Each string s in Bag_i is weighted by its relative frequency w_s: an occurrence in the Uri and title is considered as twice occurrences in the snippet, and the total number of occurrences of a string is then normalized w.r.t. the sum of all weights of all strings in Bag_i so that it is $w_s \in [0,1]$.

B. Generation of Groups

An immediate way to generate a group of ranked items is to perform a Web search by submitting a query q to a search engine SE through a call to the *CQuery* operator as follows: $CQuery(q, SE)$. Further, the N top ranked items in the list of retrieved results are clustered, by applying a clustering algorithm as the one described hereafter. A group of ranked clusters containing the items is thus generated.

C. Generation of Clusters

The second operation that we introduce is the clustering of the results of each query. To this end, in the implemented prototypal system *Matrioshka* [3], the *Lingo* clustering algorithm [9] is used, which performs an efficient flat crisp clustering of the retrieved documents on the basis of the titles and of the snippets, expanded based on Wordnet.
Since the label of a cluster must help the user to understand the main topic of the cluster, and since clusters are built in such a way clustered documents are similar as far as the content of title and snippet is concerned, the most relevant document w.r.t. the query is the one that best represents the main topic. For this reason, we decided that $Label_c = Title_i$, i.e., the title of item $i \in Content_c$ which is the most relevant item in the cluster. Notice that we do not need to access the text of the documents for extracting the features necessary to cluster them. We parse the result list provided by the search engine, containing the first N results, and extract all the information which constitutes the representation of a ranked item.

D. Comparing Items

Two functions are defined to compare pairs of ranked items, that serve distinct purposes.
The first one, named *match* performs and exact matching of the Uris of two items i and j:

$$match(i,j) = 1 \text{ if } Uri_i = Uri_j \text{ else } 0 \quad (2)$$

This function is used to verify if the two items i and j refer indeed the same web page, assuming as unique identifier of the page its Uri. The rationale of this assumption is the fact that the same document, retrieved by two different search

engines, may have different title and snippet, but have the same Uri.

Nevertheless, it can happen that the same web page is duplicated at distinct sites, or its contents are near duplicates, so two web pages may differ just for their Uris while they may have very similar contents. This motivates the introduction of other matching functions which do not identify an item by its Uri but by its bag of string Bag_i, regarded as a fuzzy set. Then, the comparison of two items is formalized by partial matching functions defined as fuzzy relations between fuzzy sets.

The first function is a weak fuzzy inclusion, indicated by $\subseteq^{I}_{F}$ computing a degree of subsethood of the first argument *i* in the second argument *j*, respectively [11] [14]:

$$\subseteq^{I}_{F}(i,j) = \frac{\sum_{k \in Bag_i} min(Bag_i[w_k], Bag_j[w_k])}{\sum_{k \in Bag_i} Bag_i[w_k]} \quad (3)$$

$\subseteq^{I}_{F}(i,j) \in [0,1]$ and we assume that, when a string *s* of Bag_i is not present in Bag_j its weight is zero. This subsethood measure gets the value zero when the bags of strings Bag_i and Bag_j do not have any common string; it gets the maximum value 1 when all the strings in Bag_i are also present with a greater-equal weight in Bag_j; it gets intermediate values in (0,1) when the two bags have some common entry. This function satisfies these properties [11]:

$\subseteq^{I}_{F}(i, i) = 1$;
$\subseteq^{I}_{F}(i, k) = 1$ iff $Bag_i \subseteq Bag_k$
$\subseteq^{I}_{F}(i, k) = 0$ iff $Bag_i \cap Bag_k = \varnothing$
if $Bag_i \subseteq Bag_k$
it is $\subseteq^{I}_{F}(s, i) \leq \subseteq^{I}_{F}(s, k) \wedge \subseteq^{I}_{F}(i, s) \geq \subseteq^{I}_{F}(k, s)$

Notice that this fuzzy inclusion is not T_Z transitive, i.e. it does not satisfy the following:

$$T_Z(\subseteq^{I}_{F}(i, j), \subseteq^{I}_{F}(j, k)) \leq \subseteq^{I}_{F}(i, k)$$

in which T_Z is the Zadeh T-norm defined as the *min* [6].

This function is used to evaluate how much the contents of an item *i* are also dealt by another item *j*. The degree it computes is interpreted as the extent of the relative specificity of the contents dealt with by *i* w.r.t. the contents dealt with by *j*.

Another matching function is the similarity between two items, computed based on the generalized Jaccard coefficient:

$$sim(i, j) = \frac{\sum_{k \in Bag_i} \min(Bag_i[w_k], Bag_j[w_k])}{\sum_{k \in Bag_i} \max(Bag_i[w_k], Bag_j[w_k])} \quad (4)$$

It was proved that this function is reflexive, symmetric, and T_W-transitive (where T_W is the Luckasiewicz T-norm $T_W(x, y)=max(x + y - 1, 0)$ [5]:

$sim(i, i) = 1$;
$sim(i, j) = sim(j, i)$;
$T_W(sim(i, j), sim(j, k)) \leq sim(i, k)$

This function is used to estimate the percentage of shared contents between two items.

E. Operations between Clusters

In this context, we regard a cluster as a fuzzy set of web pages; *Irank* is the membership degree of the web page identified by its Uri. A ranked item represents the contents of a web page.

We define the intersection and union operations between clusters in two different forms.

The ranked intersection, *RIntersection*, and the ranked union, *RUnion*, are defined as classic operators between fuzzy sets because they perform an exact matching between the Uris of the items that uniquely identify the web pages.

The soft intersection, *SIntersection*, and the soft union, *SUnion*, are named soft operators because they identify the ranked items through their bags of weighted strings, which represent the contents of the web pages and that are fuzzy subsets of strings. These soft operators are defined to combine fuzzy sets of fuzzy sets.

Definion 4: Ranked Intersection

Consider the ranked intersection of two clusters *c1* and *c2*, denoted by :

$$c = RIntersection(c1, c2) = \cap^{R}(c1, c2) \quad (6)$$

$i \in Content_{\cap R(c1,c2)}$
iff $\exists\, i1 \in Content_{c1} \wedge \exists\, i2 \in Content_{c2} \mid match(i1, i2)$, for which
if $(Irank_{i1} \neq Irank_{i2})$ then
(in this case *i1* and *i2* have distinct *Irank* but identify the same web page retrieved by two distinct searches or search engines)
$Irank_i = min(Irank_{i1}, Irank_{i2})$
$Uri_i = Uri_{i1}$
$Title_i = \text{Argmin}_{Title^k \in \{Title^{i1}, Title^{i2}\}}(Irank_k)$
$Snippet_i = \text{Argmin}_{Snippet^k \in \{Snippet^{i1}, Snippet^{i2}\}}(Irank_k)$
$Bag_i = \text{Argmin}_{Bag^k \in \{Bag^{i1}, Bag^{i2}\}}(Irank_k)$
else
(in the case *i1* and *i2* have the same *Irank* but identify the same web page too)
$Irank_i = Irank_{i1}$
$Uri_i = Uri_1$
$Title_i = shortest(Title_{i1}, Title_{i2})$
$Snippet_i = shortest(Snippet_{i1}, Snippet_{i2})$
$Bag_i = Bag_{i1} \cap Bag_{i2}$

where function *shortest* returns the shortest argument and *match* is defined as in (2) and returns 1 when $Uri_{i1} = Uri_{i2}$.

This definition is consistent when we regard a cluster as a fuzzy set of items, and consider the *Irank* of an item as its membership degree to the cluster [12]. $Label_c$ $Rankings_c$ and $Crank_c$ are computed as functions of $Content_c$.

Definion 5: Ranked Union

The ranked union, denoted by *RUnion*, is defined as follows:

$$c = RUnion(c1, c2) = \cup^{R}(c1, c2): \quad (7)$$

$i \in Content_{\cup R(c1,c2)}$
if $\exists\, i1 \in Content_{c1} \wedge \neg\exists\, i2 \in Content_{c2} \mid$ $match(i1,i2)$, then $i=i1$
else (there exists a *i2* with the same Uri)
if $(Irank_{i1} \neq Irank_{i2})$ then
$Irank_i = \max(Irank_{i1}, Irank_{i2})$
$Uri_i = \text{Argmax}_{Urik \in \{Urii1, Urii2\}}(Irank_k)$
$Title_i = \text{Argmax}_{Title^k \in \{Title^{i1}, Title^{i2}\}}(Irank_k)$
$Snippet_i = \text{Argmax}_{Snippet^k \in \{Snippet^{i1}, Snippet^{i2}\}}(Irank_k)$
$Bag_i = \text{Argmax}_{Bag^k \in \{Bag^{i1}, Bag^{i2}\}}(Irank_k)$
else $(Irank_{i1} = Irank_{i2})$
$Irank_i = Irank_{i1}$

$Uri_i = Uri_{i1}$
$Title_i = longest(Title_{i1}, Title_{i2})$
$Snippet_i = longest(Snippet_{i1}, Snippet_{i2})$
$Bag_i = Bag_{i1} \cup Bag_{i2}$
if $i2 \in Content_{c2} \mid \neg match(i1, i2), \forall i1 \in Content_{c1}$
then $i=i2$

where function *longest* returns the longest argument and *match* is defined in (2).

$Label_c$ $Rankings_c$ and $Crank_c$ are computed as functions of $Content_c$.

The $Irank_i$ of $i \in c$ is the maximum *Irank* value of *i1* and *i2*, assuming that $Irank_{i1} = 0$ (resp. $Irank_{i2} = 0$) when no item with that Uri belongs to *c1* (resp. *c2*).

To obtain the *Title*, the *Snippet* and the *Bag* of the items belonging to the resulting cluster, we select, as resulting *Title*, *Snippet* and *Bag*, those belonging to the document having the smallest (in the case of *Ranked Intersection*) or the greatest (in the case of *Ranked Union*) value of *Irank*, without making any change. The rationale of this choice is the fact that, in the aggregation based on the intersection (resp. union), we want to represent the document by its worst (resp. best) representative, in accordance with the modeling of the AND and the OR within fuzzy set theory [12]. When their *Uris* and *Iranks* are equal, which can occur when the same web page is retrieved in the same position of two ranked lists by distinct queries, we take the shortest (longest) titles and snippets in the case of intersection (union) and generate a bag that is the intersection (union) of the two bags.

Since an item ultimately identifies a web page, a cluster can be regarded as a fuzzy set of web pages, and thus $\cap^R$ and $\cup^R$ are indeed the intersection and union of fuzzy sets. It follows from the properties of the intersection and union of fuzzy sets based on the *min* and *max* that are idempotent, that also $\cap^R$ and $\cup^R$ satisfy the idempotency, commutativity, associativity, distributivity properties:

$\cap^R(c, c) = c$ $\cup^R(c, c) = c$
$\cap^R(c1, c2) = \cap^R(c2, c1)$ $\cup^R(c1, c2) = \cup^R(c2, c1)$
$\cap^R(\cap^R(c1, c2), c3) = \cap^R(c1, \cap^R(c2, c3))$
$\cup^R(\cup^R(c1, c2), c3) = \cup^R(c1, \cup^R(c2, c3))$
$\cap^R(\cup^R(c1, c2), \cup^R(c1, c3)) = \cup^R(c1, \cap^R(c2, c3))$
$\cup^R(\cap^R(c1, c2), \cap^R(c1, c3)) = \cap^R(c1, \cup^R(c2, c3))$
Further $\cap^R(c, c^c) \neq \varnothing$ $\cap^R(c, c^c) = (\cup^R(c, c^c)^c$
where c^c is the complement of cluster c defined so as to contain the same items of *c* but with a membership degree:
$\mu_{c^c}(i) = 1 - \mu_c(i)$

Let us make an example considering the items in clusters *c1* and *c2* reported in Table 1.

$\cap^R(c1,c2)$={u2/0,7} where u2.*title*="*Italian costal tourist centers*" and $Bag = Bag_{c1.u2} \cap Bag_{c2.u2}$ = {Venice/0,9} since u2 is the only item having the same *Uri* and *Irank* in *c1* and *c2*.

$\cup^R(c1, c2)$ ={u1/0,8; u3/0,9; u2/0,7} where u2.*title*="*Italian costal tourist centers*" an $Bag = Bag_{c1.u2} \cup Bag_{c2.u2}$={ Venice/1; laguna/0,8; Genoa/0,8; Rome/0,5 Capri/0,9 }

With such a strict definition of the intersection between clusters, the ranked items that represent duplicated web pages are filtered out from the result since their Uris are different (in the example items u1 and u3). In particular situations, this could be a limitation, since one would like to identify ranked items dealing with similar and duplicated topics. Let us consider, for example, the page of *Expedia* of the same hotel but retrieved in two different searches with two different dates of booking. They refer to the same hotel in the same Web site, but they have different Uris. With the *RIntersection* operator, these web pages are considered distinct, even if their semantics is the same.

In order to overcome this limitation, we defined the *Soft Intersection* between clusters. As a result, it yields a cluster *c* generated considering the shared contents between the set of titles and snippets of the ranked items belonging to the input clusters. Then, it identifies topics that represent shared contents between clusters.

On the other side, *the ranked union duplicates items* that correspond to near duplicated web pages (in the example items u1 an u3) while in these cases it would be desirable *to obtain just one item*. To cope with this limitation of the ranked union, we defined the *Soft Union* operator.

Table 1: ranked items

cl	uri	title	bag	Irank
c1	u1	*Mediterranean tourist points of interest*	{ Athens/1; Zante/0,9; Creta/1; Capri/1; Portofino/0,8; Venice/1; Palma de Mallorca/1; Saint-Tropez/1; Monaco/0,8; Zara/1 }	0,8
	u2	*Italian costal tourist centers*	{ Venice/1; laguna/0,8;}	0,7
c2	u3	*Mediterranean tourist cities*	{ Athens/1; Venice/1; Monaco/0,8; Zara/1 }	0,8
	u2	*Italian costal tourist centers*	{ Venice/0,9; Genoa/0,8; Rome/0,5 Capri/0,9; }	0,7

Definition 6: Soft Intersection

The soft intersection operation *SIntersection*, denoted by $\cap^S$, performs the content intersection of two clusters *c1* and *c2* of ranked items and returns a new cluster *c* of ranked items.

To define it, we uniquely represent an item *i* by its bag of strings Bag_i, that is, by a fuzzy subset of all possible strings. Then, the definition of the *Soft Intersection* (and *Soft Union*) is the intersection (and union) of *fuzzy sets of fuzzy sets*. Notice that this is different from the intersection (and union) of type-2 fuzzy sets [4]. In our context, the items refer to web pages, and the operators must return references to web pages associated with the items in the input clusters. In order to do this, we first apply a *partial matching* between the *Bags* of any pairs of items in the two input clusters to identify items sharing similar contents. Then, once these items pairs are identified, from them we select those that are most specific w.r.t. the contents of their most similar items in the other cluster. $\cap^S$ is defined as follows:

$$c = SIntersection(c1, c2) = \cap^S(c1, c2) \quad (8)$$

$\forall j \in Content_{\cap^R(c1,c2)}\ \exists i \in Content_{\cap^S(c1,c2)} \mid i=j$

$\forall i1 \in Content_{c1} \wedge i2 \in Content_{c2} \mid i1 \notin Content_{\cap^R(c1,c2)} \wedge$
$i2 = \text{Argmax}_{ik \in Content_{c2} \mid (sim(i1,ik) \geq \varepsilon) \wedge (\neg match(i1,ik))} (sim(i1,ik))$,
$\exists!\ i \in Content_{\cap^S(c1,c2)} \mid Irank_i = \min(Irank_{i1}, Irank_{i2})$ for which
if $((\subseteq^I_F(i1, i2)) > (\subseteq^I_F(i2, i1)) \vee$
$(\subseteq^I_F(i1, i2) = \subseteq^I_F(i2, i1) \wedge (Irank_{i1} < Irank_{i2}))$, then
$Uri_i = Uri_{i1}$
$Title_i = Title_{i1}$
$Snippet_{ii} = Snippet_{i1}$
$Bag_i = Bag_{i1}$
else if $(\subseteq^I_F(i1, i2) = \subseteq^I_F(i2, i1) \wedge Irank_{i1}=Irank_{i2}))$, then
$Uri_i = Uri_{i1}$
$Title_i = shortest(Title_{i1}, Title_{i2})$
$Snippet_i = shortest\ (Snippet_{i1}, Snippet_{i2})$
$Bag_i = Bag_{i1} \cap Bag_{i2}$
else if $i2 \notin Content_{\cap^R(c1,c2)}$, then
$Uri_i = Uri_{i2}$
$Title_i = Title_{i2}$
$Snippet_i = Snippet_2$
$Bag_i = Bag_2$

$\forall i2 \in Content_{c2} \wedge i1 \in Content_{c1} \mid i2 \notin Content_{\cap^R(c1,c2)} \wedge$
$i1 = \text{Argmax}_{ik \in Contents_{c1} \mid (sim(ik,i2) \geq \varepsilon) \wedge (\neg match(ik,i2))} (\text{sim}(ik,i2))$
$\exists\ !\ i \in \text{Content}_{\cap^S(c1,c2)} \mid Irank_i = \min(Irank_{i1}, Irank_{i2})$ for which
if $((\subseteq^I_F(i1, i2)) < (\subseteq^I_F(i2, i1)) \vee$
$(\subseteq^I_F(i1,i2) = \subseteq^I_F(i2,i1) \wedge (Irank_{i2} < Irank_{i1}))$, then
$Uri_i = Uri_{i2}$
$Title_i = Title_{i2}$
$Snippet_i = Snippet_{i2}$
$Bag_i = Bag_{i2}$
else if $(\subseteq^I_F(i1, i2) = \subseteq^I_F(i2, i1) \wedge Irank_{i1}=Irank_{i2}))$, then
$Uri_i = Uri_{i2}$
$Title_i = shortest(Title_{i1}, Title_{i2})$
$Snippet_i = shortest\ (Snippet_{i1}, Snippet_{i2})$
$Bag_i = Bag_{i1} \cap Bag_{i2}$
else if $i1 \notin Content_{\cap^R(c1,c2)}$ then
$Uri_i = Uri_{i1}$
$Title_i = Title_{i1}$
$Snippet_i = Snippet_{i1}$
$Bag_i = Bag_{i1}$

in which $\varepsilon \in [0,1]$ is a minimum similarity degree and $\subseteq^I_F$ and *sim* are defined as in formulae (3) and (4) respectively.

$Label_c$ $Rankings_c$ and $Crank_c$ are computed as functions of $Content_c$.

Some properties can be proved.

$\cap^R(c1, c2) \subseteq \cap^S(c1, c2)\ \ \forall \varepsilon \in [0, 1]$ by definition.

Idempotency

From the previous and since $\cap^R$ is idempotent it follows that $\cap^S(c, c) \supseteq c$ and, since there cannot be two items with the same Uri in a cluster $\cap^S(c, c) = c$.

Commutativity

Since *sim* and *match* functions and the two conditions in (8) are symmetric, we have that:

$\cap^S(c1, c2) = \cap^S(c2, c1)$

From the properties of $\cap^R$ it also follows that:

$\cap^S(c, c^c) \neq \varnothing$

Further, $\cap^S(c1, \cap^S(c2, c3)) \neq \cap^S(\cap^S(c1, c2), c3)$ for $\varepsilon \in (0,1]$.

The associativity property is not satisfied, since the selection condition based on the minimum (not null) similarity ε and the inclusion may filter out from the intermediate results elements that cannot be recovered any more. For example, let us consider the case in which $i1 \in c1$ which is more specific than any other similar item $j \in \cap^S(c2, c3)$: it follows that $i1 \in \cap^S(c1, \cap^S(c2, c3))$. At the same time, it can happen that $\exists$ an item $i2 \in c2$ similar to *i1* and most specific so that $i1 \notin \cap^S(c1, c2)$, and at the same time not enough similar to any $i3 \in c3$ so that $i2 \notin \cap^S(c2, c3)$: it follows that $i1 \notin \cap^S(\cap^S(c1, c2), c3)$.

On the other side, when ε=0, the condition on the minimum similarity is not imposed. Nevertheless, also in this case associativity is not satisfied, due to the fact that $\subseteq^I_F$ is not transitive. Consider the previous counter example: assuming $i1 \in c1$ is more specific than any other item $j \in \cap^S(c2, c3)$ so that $i1 \in \cap^S(c1, \cap^S(c2, c3))$. If there exists an item $i2 \in c2$ most specific then *i1* so that $i1 \notin \cap^S(c1, c2)$, there must be an item $i3 \in c3$ so that $i3 \in \cap^S(c2, c3)$, i.e., *i3* is most specific than any other item in *c2*. If this happens, in order to guarantee associativity *i3* should be also more specific than *i1*, and this cannot be stated since $\subseteq^I_F$ is not transitive.

The *Soft Intersection* relaxes the constraint of the *Ranked Intersection*. It generates a cluster *c* that contains both the results of the *Ranked Intersection* of the two input clusters *c1* and *c2*, plus the items of the input clusters that have the most specific contents w.r.t. those dealt with in the most similar item of the other cluster, as it can be guessed from their *Bags*.

Let us explain the rationale of this definition with a simple example. Given two documents, one dealing with "*Mediterranean Tourist Points of Interest*", and the second with "*Mediterranean Tourist cities*", they probably share most of the places listed in the second document, since the Mediterranean Tourist cities are indeed Mediterranean Points of Interest, but the vice versa is unlikely to occur, because the first document contains also names of islands, and picturesque villages, such as Capri, Portofino, Saint-Tropez and so on. So, the *Soft Intersection* retains only the shared contents, i.e., the second document on Mediterranean cities.

Consider the example in Table 1, by setting ε=0 we obtain:

$\cap^S(c1, c2)$={u2/0,7; u3/0,8}

u2 is obtained since it belongs to $\cap^R(c1,c2)$ and u3 is obtained since it is more specific than u1: $\subseteq^I_F(u3,u1) > \subseteq^I_F(u1,u3)$.

Definition 7: Soft Union

The operation *SUnion*, denoted by $\cup^S$, performs the content union of two clusters *c1* and *c2* of ranked items.

To define it, we uniquely represent an item *i* by its bag of strings Bag_i and we evaluate a partial matching of any pairs of *Bags* of the items in the two input clusters.

$\cup^S$ is defined as follows:

$$c = SUnion(c1, c2) = \cup^S(c1, c2) \qquad (9)$$

$\forall j \in Content_{\cap^R(c1,c2)}\ \exists i \in Content_{\cup^S(c1,c2)} \mid i=j$

$\forall i1 \in Content_{c1} \mid i1 \notin \text{Content}_{\cap^R(c1,c2)}$
if $\neg\exists\ i2 \in Content_{c2} \mid sim(i1,i2) < \varepsilon$ then

$\exists!\ i \in Content_{\cup S(c1,c2)} \mid i=i1$

else ($\exists\ i2 \in Content_{c2} \mid sim(i1,i2) \geq \varepsilon$) for which

if ($\subseteq^{I}_{F}(i1,i2) = \subseteq^{I}_{F}(i2,i1) \wedge Irank_{i1} = Irank_{i2}$), then

$\exists!\ i \in Contents_{\cup S(c1,c2)} \mid i=i1 \wedge \exists!\ j \in Contents_{\cup S(c1,c2)} \mid i=i2$

else if ($\subseteq^{I}_{F}(i1,i2) < \subseteq^{I}_{F}(i2,i1)$)

$\vee$ ($\subseteq^{I}_{F}(i1,i2) = \subseteq^{I}_{F}(i2,i1) \wedge Irank_{i1} > Irank_{i2}$) then

$\exists!\ i \in Content_{\cup S(c1,c2)} \mid$

$Irank_i$=max($Irank_{i1}$, $Irank_{i2}$)

$Uri_i = Uri_{i1}$

$Title_i = Title_{i1}$

$Snippet_i = Snippet_{i1}$

$Bag_i = Bag_{i1}$

else if ($\subseteq^{I}_{F}(i1,i2) > \subseteq^{I}_{F}(i2,i1)$

$\vee$ ($\subseteq^{I}_{F}(i1,i2) = \subseteq^{I}_{F}(i2,i1) \wedge Irank_{i1} < Irank_{i2}$) then

$\exists!i \in Content_{\cup S(c1,c2)} \mid$

$Irank_i$=max($Irank_{i1}$, $Irank_{i2}$)

$Uri_i = Uri_{i2}$

$Title_i = Title_{i2}$

$Snippet_i = Snippet_{i2}$

$Bag_i = Bag_{i2}$

$\forall i2 \in Content_{c2} \mid i2 \notin Content_{\cap R(c1,c2)}$

if $\neg\exists\ i1 \in Content_{c1} \mid sim(i1,i2) < \varepsilon$ then

$\exists!i \in Content_{\cup S(c1,c2)} \mid i=i2$

else ($\exists\ i1 \in Content_{c1} \mid sim(i1,i2) \geq \varepsilon$) for which

if ($\subseteq^{I}_{F}(i2,i1) < \subseteq^{I}_{F}(i1,i2)$)

$\vee$ ($\subseteq^{I}_{F}(i1,i2) = \subseteq^{I}_{F}(i2,i1) \wedge \text{Irank}_{i2} > Irank_{i1}$) then

$\exists!\ i \in Content_{\cup S(c1,c2)} \mid$

$Irank_i$=max($Irank_{i1}$, $Irank_{i2}$)

$Uri_i = Uri_{i2}$

$Title_i = Title_{i2}$

$Snippet_i = Snippet_{i2}$

$Bag_i = Bag_{i2}$

else if ($\subseteq^{I}_{F}(i2,i1) > \subseteq^{I}_{F}(i1,i2)$

$\vee$ ($\subseteq^{I}_{F}(i1,i2) = \subseteq^{I}_{F}(i2,i1) \wedge Irank_{i2} < Irank_{i1}$) then

$\exists!i \in Content_{\cup S(c1,c2)} \mid$

$Irank_i$=max($Irank_{i1}$, $Irank_{i2}$)

$Uri_i = Uri_{i1}$

$Title_i = Title_{i1}$

$Snippet_i = Snippet_{i1}$

$Bag_i = Bag_{i1}$

in which $\varepsilon \in [0,1]$ is a minimum similarity degree and $\subseteq^{I}_{F}$ and *sim* are defined as in (3) and (4), respectively. The notation "$\exists!\ i$" stands for "there exists one and only one *i*.

$Label_c$ $Rankings_c$ and $Crank_c$ are computed as functions of $Content_c$.

Some properties can be proved: $\cup^S$ is idempotent, commutative, and monotonic not decreasing.

Idempotency

$\cup^S(c, c) = c$

Since duplicated items with the same Uri are not allowed in a cluster, we have that $\cup^S(c, c) \subseteq \cup^R(c, c)$;

Further $\neg\exists\ i \notin \cup^S(c, c) \wedge i \in \cup^R(c, c)$;

Assuming that $i \notin \cup^S(c, c) \wedge i \in \cup^R(c, c)$ would mean that $\exists j \in c \mid sim(i,j) \geq \varepsilon$ that is more general than *i* or that has a smaller *Irank*; nevertheless, this condition would be satisfied:

($\subseteq^{I}_{F}(i1,i2) = \subseteq^{I}_{F}(i2,i1) \wedge Irank_{i1} = Irank_{i2}$), so $i \in \cup^S(c, c)$.

Commutativity

$\cup^S(c1, c2) = \cup^S(c2, c1)$ since *sim* and the two conditions in (9) are symmetric.

Further, $\cup^R(c1, c2) \supseteq \cup^S(c1, c2)$

If $\exists\ i \in \cup^S(c1, c2)$ and $i \notin \cup^R(c1, c2)$, it means that $i \notin c1$ and $i \notin c2$, thus *i* does not exists, which contradicts the assumption.

$\cup^S(\cup^S(c1, c2), c3) \neq \cup^S(c1, \cup^S(c2, c3))$

The associativity property is not satisfied due to the condition on the similarity and the intransitivity of the weak fuzzy inclusion.

The *Soft Union* restricts the ranked union by eliminating, from its results, the most specific items having a similar item in the other input cluster. Let us give an example of utility. Assume that we want to have a panoramic overview of the Mediterranean Tourist information by eliminating redundant contents; having two documents, one dealing with "*Mediterranean tourist points of interest*", and the second with "*Mediterranean Tourist cities*", we want to eliminate the second document from the results and keep the first one that is more general: to achieve this, we apply a *Soft Union*. Consider the example in Table 1, by setting ε=0 we obtain: $\cup^S(c1, c2)$={u2/0,7, u1/0,8}

u2 is obtained since it belongs to $\cap^R(c1,c2)$ and u1 is obtained since it is more general than u3: $\subseteq^{I}_{F}(u3,u1) > \subseteq^{I}_{F}(u1,u3)$.

The distributivity property of $\cap^S$ w.r.t. $\cup^S$ and viceversa do not hold due to the not associativity of $\cap^S$ and $\cup^S$:

$\cap^S(\cup^S(c1, c2), \cup^S(c1, c3)) \neq \cup^S(c1, \cap^S(c2, c3))$

$\cup^S(\cap^S(c1, c2), \cap^S(c1, c3)) \neq \cap^S(c1, \cup^S(c2, c3))$

F. Operators between Groups

There are several operators taking Groups, i.e., the coarsest granules of information that we can manipulate, as arguments and generating a new group. They are defined based on the cluster operations previously introduced. Here, we just define the basic ones used to identify shared documents and contents and correlated documents and contents.

Definition 8: Group Intersection Operators : $\cap^{GR}$ and $\cap^{GS}$

The *Group Ranked Intersection* operator $\cap^{GR}$ and the *Group Soft Intersection* operator $\cap^{GS}$ are defined so as to exploit the *Ranked Intersection* $\cap^R$ and the *Soft Intersection* $\cap^S$ between all the pairs of clusters belonging to the two input groups.

Given two groups of clusters *g1* and *g2*, both $\cap^{GR}$ and $\cap^{GS}$ hereafter indicated simply by $\cap$ are defined as follows:

$g = \cap(g1, g2) \mid$

$\forall\ (c1, c2) \mid c1 \in Clusters_{g1} \wedge c2 \in Clusters_{g2}$

$\exists c \in Clusters_g$ if $\cap^*(c1, c2) \neq \varnothing \wedge$

$c = \cap^*(c1, c2)$ in which $\cap^* \equiv \cap^R$ in the case of $\cap^{GR}$ while $\cap^* \equiv \cap^S$ in the case of $\cap^{GS}$.

$Label_c$ $Rankings_c$ and $Crank_c$ are computed as functions of $Content_c$.

$Label_g$ is defined as a function of the items in all clusters of g.

It can be proved that $\cap^{GR}$ and $\cap^{GS}$ are idempotent, commutative. Further $\cap^{GR}$ is also associative while $\cap^{GS}$ is not.
Example of application of the group intersection operators are provided in the next section.

Definition 9: Group Union Operators $\cup^{GR}$ and $\cup^{GS}$
The group ranked union operator $\cup^{GR}$ and the group soft union operator $\cup^{GS}$ are defined so as to exploit the ranked union $\cup^{R}$ and the soft union $\cup^{S}$ between all the pairs of clusters originated from the two input groups.
Given two groups of clusters *g1* and *g2*, both $\cup^{GR}$ and $\cup^{GS}$ hereafter indicated simply by $\cup$ are defined as follows:
$g = \cup(g1, g2)$
$\forall(c1, c2) \mid c1 \in Clusters_{g1} \wedge c2 \in Clusters_{g2}$
$c \in Clusters_g \mid c = \cup^{+}(c1, c2)$, in which $\cup^{+} \equiv \cup^{R}$ in the case of $\cup^{GR}$ while $\cup^{+} \equiv \cup^{S}$ in the case of $\cup^{GS}$.
$Label_c$ $Rankings_c$ and $Crank_c$ are computed as functions of $Content_c$.
$Label_g$ is defined as a function of the items in all clusters of *g*.
It can be proved that these operators are idempotent, commutative, and monotonic.
Further $\cup^{GR}$ is also associative while $\cup^{GS}$ is not.
This operator allows generating clusters of all (non redundant) contents dealt with in pairs of input clusters.

Definition 10: Group Join Operators $><^{GR}$ and $><^{GS}$
The *Group Ranked Join* operator $><^{GR}$ and the *Group Soft Join* operator $><^{GS}$ are defined so as to exploit the crisp and soft operators between all the pairs of clusters belonging to the two input groups.
Given two groups of clusters *g1* and *g2*, both $><^{GR}$ and $><^{GS}$ hereafter indicated simply by $><$ are defined as follows:

$g = ><(g1, g2)$
$\forall (c1, c2) \mid c1 \in Clusters_{g1} \wedge c2 \in Clusters_{g2}$
$\exists c \in Clusters_g$ if $\cap^{*}(c1, c2) \neq \varnothing \wedge c = \cup^{+}(c1, c2)$
in which $\cap^{*} \equiv \cap^{R}$ and $\cup^{+} \equiv \cup^{R}$ in the case of $><^{GR}$ while $\cap^{*} \equiv \cap^{S}$ and $\cup^{+} \equiv \cup^{S}$ in the case of $><^{GS}$.
$Label_c$ $Rankings_c$ and $Crank_c$ are computed as functions of $Content_c$.
$Label_g$ is defined as a function of the items in the clusters of *g*.
It can be proved that both $><^{GR}$ and $><^{GS}$ are reflexive, commutative, and monotonic, and $><^{GR}$ is also associative.
These operators allow filtering clusters of correlated topics, when they share some topic. Example of application of the group join operators are provided in the next section.

IV. EXAMPLE OF PERSONALIZED EXPLORATORY ACTIVITY BY THE AID OF THE SOFT OPERATORS

Matrioshka is a meta-search system designed and implemented to perform personalized explorations of the results retrieved in a Web search process [1][2]. Among its functionalities, it allows: submitting queries to four search engines (*Google, Google Scholar, Yahoo!* and *Bing*). It allows clustering the list of results. Finally, it makes available the operators for manipulating groups, that the user can apply for combining pairs of lists to explore their shared contents and documents. Notice that, since this manual application of operators can be uneasy for inexperienced users, *Matrioshka* provides an alternative way for exploring the shared contents between distinct groups. The *graph* utility is made available that displays, in the form of labeled multi-granular graphs, the results in the selected groups, and their shared documents obtained by applying their group ranked intersection and group soft intersection [3].
To show an example of exploratory analysis by the use of the soft operators, we submitted the two queries "*Proceedings SIGIR*" and "*Proceeding ECIR*" to *Google Scholar* through *Matrioshka*; then, we executed a *Ranked Intersection* with the results of the previous queries. Observing that we got an empty group, we tried a *Soft Intersection* by setting the minimum similarity threshold to ε=0.5. This time we obtained the group entitled "*Proceeding information ACM Retrieval Conference*" (indeed both SIGIR and ECIR are ACM conferences on IR themes) containing 13 clusters with documents dealing with shared topics in the input groups (see Figure 1)

g1="Proceedings information ACM retrieval conference"
C1 Scores distribution in Information retrieval
C2 Interactive visualization of multiple query results
C3 Query expansion using random walk models
C4 Empirical studies of information visualization: a meta-analysis
C5 Retrieval constraints and words frequency distributions: a log logistic model for IR
C6 Categorizing paper documents…
C7 Language models for Information Retrieval
C8 Methods and apparatus for distributed indexing and retrieval
C9 Apparatus and Methods for collaboratively searching knowledge databases
C10 Where to start reading a textual XML documents
C11 Advances in Information Retrieval
C12 Hierarchical clustering with real time updating
C13 Automatic construction of known item finding test-bed

Fig 1: clusters in the group g1 are obtained by executing "*Proceeding SIGIR*" $\cap^{GS}_{0.5}$ "*Proceeding ECIR*"

Finally, we applied the *Group Soft Join* operator to generate a group containing documents dealing with correlated topics.
At first, we set ε=0.5 to require a strong correlation between the clusters. One group g2 containing two clusters g2.*C1* and g2.*C2* of correlated topics was generated (see Figure 2). One could question why the number of shared topics between two input groups, i.e., the number of clusters in g1, is greater than the number of correlated topics between the same two input groups, i.e., the number of clusters in group g2. The reason is that correlation is a less strict relationship than sharing. Notice that two homonymous clusters exists in g1 and g2: in fact g1.*C2* and g2.*C1*, as well as g1.*C3* and g2.*C2*, have the same label, but different content since g2.*C1* and g2.*C2* contain additional documents w.r.t. g1.*C2* and g1.*C3* respectively.
We reapplied the *Group Soft Join* operator by decreasing the minimum correlation ε=0.2: this time, we obtained the group g3 with 11 clusters of correlated topics (see Figure 2). This greater number of clusters in g3 w.r.t. the number of clusters in g2 is due to the fact that, by decreasing the correlation

threshold, we get a not empty soft intersection for more than two pairs of input clusters. Notice that also in g3 we have the two clusters g3.*C4* and g3.*C6* homonymous of g1.*C2* and g1.*C3* respectively.

Figure 3 illustrates another example in which we submitted the same query "*visit Greece*" to *Yahoo!* and *Bing* and further, in order to filter the most relevant results, we applied a *Ranked Intersection* and finally a *Soft Intersection*. It can be observed that the Group resulting from the *Soft Intersection* contains additional documents w.r.t. the group resulting from the *Ranked Intersection*.

g2="Proceedings information ACM retrieval conference"= "Proceeding SIGIR">$<^{GS}_{0.5}$ "Proceeding ECIR"
C1 Interactive visualization of multiple query results
C2 Query expansion using random walk models
g3="Proceedings retrieval information ACM conference"= "Proceeding SIGIR">$<^{GS}_{0.2}$ "Proceeding ECIR"
C1 Methods and apparatus for extracting data from data sources on a network
C2 Categorizing paper documents
C3 Systems and methods for querying multiple, distributed databases
C4 Query expansion using random walk models
C5 Facilitating WWW search utilizing a multiple search engine query clustering fusion
C6 Interactive visualization of multiple query results
C7 The effects of topic familiarity on information search behaviour
C8 Cha-cha: a system for organizing intranet search results
C9 Where to start reading a textual XML documents
C10 Advances in Information Retrieval
C11 Report on the 25th European conference on information retrieval research (ECIR-03)

Fig 2: Two output Groups obtained by the execution of "*Proceeding SIGIR* " $><^{SG}$ "*Proceeding ECIR*" with correlation ε=0.5 and ε=0.2 respectively

g4="visit Greece" to Yahoo!	**g5="visit Greece" to Bing**
C1=Visit Greece Answerbag	*C1=Week*
C2=Greece Tourism	*C2=Ancient Greece*
C3=Greek Islands	*C3=Time to visit Greece*
C4=Greece Holiday Hotels Flights	*C4=Northeastern Aegean Islands*
C5=Time to visit Greece	*C5=Places to visit*
C6=Destination	*C6=American Jewish Leaders*
C7=Top Reasons	*C7=Getting*
C8=Athens Greece Holiday Package	*C8=Reasons to visit Greece*
C9=Greece Honeymoon	
C10=Services	
C11=Ancient Greece	
C12=Visit Greece events	
g6= g1 $\cap^R$ g2 = "best Greece travel visit guide"	**g7 = g1 $\cap^S$ g2 ="Greece travel visit best"**
C1 =Greece travel *Visit Greece* *Greece Tourism | Best Places..* *The best of Greece*	*C1=Greece travel* *Visit Greece* *Greece Tourism | Best Places..* *Holidays in Greece, Go to ..* *The best of Greece* *C2=Places to visit* *Greece Tourism | Best Places..* *Greece Places to visit*

Fig 3: Groups g4 and g5 are obtained by submitting "*visit Greece*" to *Yahoo*! and *Bing* respectively, groups g6 and g7 are obtained by the *Rintersection* and *SIntersection* of g4 and g5

V. CONCLUSIONS

In the paper we defined some soft operators for combining information granules of distinct resolution which represent web pages retrieved by distinct searches submitted to possibly distinct search engines. We discussed their properties and their application for web exploration. We figure out that the main use of these operators is for exploring the contents relationships between the results of distinct queries to search engines. We further figure out that their repetitive nested application does not make too much sense and will be rarely needed. Thus, the fact that the soft operators are not associative is not a big concern in practical use. Nevertheless, further studies are needed to assess their potential utility.

ACKNOWLEDGMENT

We thank Simone Fidanza and Valentina Taramelli, students at University of Bergamo for the implementation.

REFERENCES

[1] Bordogna G., Campi A., Psaila, G., & Ronchi S. (2008) A language for manipulating groups of clustered web documents results, In ACM CIKM '08: Proceedings of the: the 17th ACM conference on Information and knowledge management (pp 23-32). ACM Press.

[2] Bordogna, G., Campi, A., Psaila, G., & Ronchi, S. (2008). An interaction framework for mobile web search. In MoMM-2009 6th International Conference on Advances in Mobile Computing and Multimedia (pp. 183–191). Linz, Austria:ACM.

[3] Bordogna, G., Campi, A., Psaila, G., & Ronchi, S. (2010). Web Search Results Discovery by Multigranular Graphs, submitted for publication in Ramon Brena, Adolfo Guzman eds , Quantitative Semantics and Soft Computing Methods for the Web: Perspectives and Applications", to be published in 2011 by IGI Global.

[4] Castillo O., Melin P. (2008), Type-2 Fuzzy Logic: theory and Applications, Studies in Fuzziness and Soft Computing Series, Springer Verlag.

[5] De Baets B., De Meyer H., Naessens H., A class of rational cardinality-based similarity measures, Journal of Computers and Applied Mathematics (2001) 51–69.5.

[6] De Baets B., De Meyer H., Naessens H., On rational cardinality-based inclusion measures, Fuzzy Sets and Systems 128 (2002) 169 – 183.

[7] De Graaf E., Kok J., & Kosters W.. (2007). Clustering improves the exploration of graph mining results. In IFIP2007: Proceedings of the Articial Intelligence and Innovations 2007: from Theory to Applications, volume 247 of IFIP International Federation for Information Processing, (pp 13-20). Springer Verlag.

[8] Fellbaum, C. (Ed.) (1998). WordNet An Electronic Lexical Database. Cambridge, MA ; London: The MIT Press.

[9] Jansen, B. J., & Spink, A. (2006). How are we searching the world wide web? a comparison of nine search engine transaction logs. Information Processing and Management, 42, 248263.

[10] Jansen, B. J., Spink, A., & Saracevic, T. (2000). Real life, real users and real needs: A study and analysis of users queries on the web. Information Processing and Management, 36, 207–227.

[11] Kosko B., (1992) Neural Networks and Fuzzy Systems: a Dynamical Systems Approach to Machine Intelligence (Prentice-Hall, Englewood Cliffs).

[12] Lucene(Web Site). Lucene java documentation. http://lucene.apache.org.

[13] Markov A., Last M., & Kandel A. (2007). Fast categorization of web documents represented by graphs. In WEBKDD: Proceedings of Advances in Web Mining and Web Usage Analysis. LNCS 4811, (pp 56-71).Springer Verlag.

[14] Miyamoto S. (1990) Fuzzy sets in information retrieval and clustering analysis, Kluwer Academic Press.

[15] Osinski, S., &Weiss, D. (2005). A concept-driven algorithm for clustering search results. IEEE Intelligent Systems, 20, 48–54.

[16] Pan J., Yang H., Faloutos C., & Duygulu P. (2007). Crossmodal correlation mining using graph algorithms, In Zhu X., Zhu X.; Davidson I. (Eds) Knowledge Discovery and Data Mining: Challenges and Realities with Real World Data. (pp. 274-294) IGI Global.

[17] Roussinov, D. G., & Chen, H. (2001). Information navigation on the web by clustering and summarizing query results. Information Processing and Management, 37, 789 – 816.

[18] Zadeh, L.A. (1965) Fuzzy sets. Information and control, 8, 338-353.