Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE @ DIRECT®

INTERNATIONAL JOURNAL OF APPROXIMATE REASONING

ELSEVIER International Journal of Approximate Reasoning 34 (2003) 105–120
www.elsevier.com/locate/ijar

# Soft approaches to distributed information retrieval

Gloria Bordogna [a], Gabriella Pasi [b,*], R.R. Yager [c]

[a] *CNR-IDPA Piazza Cittadella 4, 24129 Bergamo, Italy*
[b] *CNR-ITC via Bassini 15, 20133 Milano, Italy*
[c] *Department of Machine Intelligence, IONA College, New Rochelle, NY 10801, USA*

Received 1 January 2003; accepted 1 July 2003

## Abstract

In this paper we consider the context of information retrieval in a network. Two distinct situations of distributed retrieval are considered: the case in which we have distinct databases residing on distinct servers having their own IRSs, and the case in which we have a unique, huge and distributed information repository (like the WWW), and distinct IRSs (search engines), which can be used to inquiry the same collection. In both situations we have the problem of merging the results produced by the distinct IRS into a unique ordered list, which constitutes the unified answer to the user query. In solving the problem of list fusion two main issues are involved. The first one concerns the decision of how many retrieved documents to select from each individual documents list. The second aspect is to define a procedure to fuse the individual lists of retrieved documents into an overall ordered list. In this paper some fuzzy approaches to solve the two above-mentioned problems are presented.
© 2003 Elsevier Inc. All rights reserved.

*Keywords:* Distributed information retrieval; Metasearch engines; List fusion; Fuzzy quantifiers

[*] Corresponding author. Tel.: +39-0223699549.
*E-mail addresses:* gloria.bordogna@idpa.cnr.it (G. Bordogna), gabriella.pasi@itc.cnr.it (G. Pasi), ryager@iona.edu (R.R. Yager).

0888-613X/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/j.ijar.2003.07.003

## 1. Introduction

In this paper, the issue of searching for information distributed on a wide area network is considered. As outlined in [1], there are two main models describing the problem of retrieving such information: in the first model the information is considered as belonging to a unique, huge database which is centrally indexed for retrieval purposes. This is the model adopted by search engines on the WWW. A second model is based on the distribution of the information on distinct databases, independently indexed, and thus constituting distinct sources of information. This last model gives rise to the so called distributed or multi-source information retrieval problem. In this second case the databases reside on distinct servers each of which can be provided with its own search engine (IRS). The distributed information retrieval paradigm is more complex than the centralized model as it presents additional problems, such as the selection of an appropriate information source for a given information need. A common problem which can be identified with both models is the problem of list fusion. In the case in which we have a unique, huge and distributed information repository (like in the WWW), and distinct IRSs (search engines), which can be used to inquiry overlapping collections, metasearch engines have been used to improve the effectiveness of the individual search engines. The main aim of a metasearch engine is to submit the same query to distinct search engines and to fuse the individual resulting lists into an overall ranked list of documents that is presented to the user. In this case we typically have overlapping individual lists since a document may be retrieved by more than a single search engine. The fusion method thus has to be able to handle situations in which a document may appear in more than one list and in different positions within the lists. In the case of multi-source information retrieval the problem is to merge the lists resulting from the processing of the same query by (generally distinct) search engines on the distinct databases residing on distinct servers. However, in this second case we generally do not have overlapping lists as a result of the same query evaluation. Typically a document will be retrieved by just one single search engine, and thus the fusion problem is simplified with respect to the previous case. Recently in the literature several papers have addressed the problem of defining effective solutions to the problem of retrieving information on a network. In [2–8] some approaches to the definition of metasearch engines are presented, while in [1,9–11] some solutions to the problem of multi-source information retrieval are described.

In this paper we consider both situations of information retrieval on a network, and present some possible approaches to the above-mentioned problems. The uniqueness of these approaches is that they are based on soft computing techniques to more flexibly model the resource selection problem (in distributed information retrieval), and the list fusion problem.

In Sections 2 and 3 the steps involved in a multi-source information retrieval activity and in a metasearch activity are outlined and some soft approaches to deal with them are introduced.

## 2. The multi-source information retrieval activity

An important contemporary problem in the area of information retrieval concerns the retrieval of information distributed over multiple sources, each of which resides on a server. An example of document search from multiple information sources is offered by the inquiry of archives of newspapers: a user looking for the market trend in USA could be interested in inquiring the archives of the articles maintained for example by the New York Times, the Wall Street Journal and the Chicago Tribune. In this multi-source information retrieval activity it is assumed that we have a number of servers each containing its own unique collection of documents and each provided with their own information retrieval system (IRS): a user query is independently evaluated by each IRS which produces a list of documents ordered on the basis of a score indicating the estimated relevance of the document with respect to the query (this score is called the retrieval status value (RSV)). We can expect that, since the documents repositories have unique collections, as a result of a query evaluation we obtain disjoint ordered lists. The computation of the RSVs depends on the model on which the IRS is based. What the user would expect is an overall list of documents, which must be obtained by the fusion of the individual lists produced by each server. This is not an easy task, as the retrieval engines may be based on different retrieval models, which apply different matching mechanisms, with the consequence of producing incomparable RSVs. The main problem related to list fusion is then to compare the ordered lists and construct an overall list which preserves the retrieval results produced by the individual servers and which approximates the effectiveness of searching the entire set of documents as a single collection.

In [11–13] a three steps strategy for solving the problem of retrieval from multiple sources of information has been proposed. In the first step a query $Q$ is submitted to the IRSs of the $m$ different servers, denoted by $S_i$, with $i = 1, \dots, m$, which evaluate it; the result of this step is $m$ ordered lists of documents, one for each server. It is assumed that in the $m$ lists each document has an RSV associated. In the second step a procedure is applied to establish the number of documents that each server should contribute. The third step concerns the problem of fusing the individual ordered lists into an overall list which is presented to the user. In the next subsections we discuss some approaches based on soft computing to address the last two steps of this process: determining the number of documents to select from each individual list and the fusion of the individual ordered lists. The first step, while of great importance, is not considered here.

### *2.1. Determination of the number of documents to select from each list in multi-source information retrieval*

In this section we illustrate some approaches to establish the number $N_i$ of documents that each server $S_i$ is allocated to contribute. If $N$ is the total number of documents to be presented to the user we have that $\sum_{i=1}^{m} N_i = N$. Once having $N_i$ from each server we can obtain the sublist, $L_i$, which contains the $N_i$ documents with the highest score.

The process of determining the numbers $N_i$ depends on many factors, among which the type and the nature of the information contained in the database residing on server $S_i$, the topics investigated by the user and formalized in her/his query, the appropriateness of the server as collector of documents relevant to the considered query. This is then a knowledge intensive problem, as such some meta information about the content of the server is needed, as well as some information about the nature of the query being considered. In the literature some approaches have been proposed to face this problem [11–15]. In [11–13] the authors have proposed an approach to determine these numbers, the $N_i$, based on the use of training queries to build both the content and search behaviour of each collection.

In [16] a knowledge based approach has been proposed which is based on the computation of a score for each server indicating its *fitness* with respect to the considered query. A fitness score indicates how much a server is a good collector of the relevant documents for a given query. In order to compute the fitness scores for the present query the authors propose a rule based approach. First a set of query types is defined; associated with each query type is a set of fitness scores (one per server). A query type is formalized by means of a rule in which the antecedent is the description of the query in terms of a set of properties, and the consequent specifies the fitness of each server with respect to this query type. Formally, the properties useful to characterize the subject matters of queries are denoted as $U_j$ $(j = 1, \dots, s)$; the fitness scores are the values associated with variables $V_i$ $(i = 1, \dots, m)$, one for each server $S_i$. The fuzzy rules identifying query types are formalized as

$$\text{if } U_1 \text{ is } A_{k1} \text{ and } \dots \text{ and } U_s \text{ is } A_{ks} \text{ then } \mathbf{V} \text{ is } \mathbf{B}_k$$

in which $k$ is the index of the rule, the $A_{kj}$ are fuzzy subsets constituting the values of variables $U_j$, $\mathbf{V}$ is an $m$-dimensional vector whose components are the $V_i$, and $\mathbf{B}_k$ is also an $m$-dimensional vector whose components, $b_{ki}$, correspond to the fitness of server $S_i$ as an appropriate source of documents for the type of query described in the antecedent of rule $k$. Once a user query has been evaluated by the IRSs on the servers, a "matching" between it and the query type rules is performed, in order to compute the fitness scores for the considered query on the basis of the scores of the query types. The first step is to obtain a unique fitness score per server for the considered query. To this aim the user query is also described in terms of properties $U_j$; a value $a_j$ is associated with each $U_j$ and for each rule $k$ a firing level $\tau_k$ is computed: $\tau_k = \min_j[A_{kj}(a_j)]$. To combine the fitness scores in the distinct rules, the obtained information is employed as follows:

$$\mathbf{B}^* = \frac{\sum_{k=1}^{p} \tau_k \mathbf{B}_k}{\sum_{k=1}^{p} \tau_k} \tag{1}$$

in which $p$ is the number of query types.

The components $b_i^*$ of vector $\mathbf{B}^*$ denote the fitness values of the servers for the considered query.

Finally, to obtain the values $N_i$ of documents to be contributed by each server, first the fitness values are normalized:

$$\alpha_i = \frac{b_i^*}{\sum_{i=1}^{m} b_i^*} \tag{2}$$

At this point the normalized values are used to compute the number of documents to be retrieved from each server:

$$N_i = \alpha_i N \tag{3}$$

The number of documents to be selected from each server is proportional to the fitness of that server to the query.

In the next subsection we present some soft approaches to compute the fitness score for each server.

### *2.2. Soft approaches to rank the information sources in multi-source information retrieval*

In this subsection we synthesize what has been proposed in [11] for estimating the fitness scores of the information sources so as to serve the computation of the number of documents to be contributed by each server.

Each approach proposes a distinct way of computing fitness scores. Starting from the obtained fitness scores the numbers of document to be selected from each server is determined by applying formulas (2) and (3) in Section 2.1.

#### *2.2.1. Approach based on the association of a fuzzy prototypal set with each server*

The idea at the heart of this approach is to associate with each server a fuzzy set of the key terms which best represent the topics in the considered archive. Their associated membership values are interpreted as their significance as descriptors of the server. In other words the weights represent the fitness of the server with respect to the topic expressed by the associated term. The fuzzy set can be interpreted as a "prototypal document" or "document type" constituting a representative of the documents in the archive of the server. The terms and their "fitness" weights can be either manually defined or automatically computed. An automatic selection of the key terms can be obtained on the basis of their normalized occurrences in the archive on the considered server; the computed occurrences can then be used as terms membership values. Let us denote by $\mathrm{TOTOCC}(t)_i$ the total number of occurrences of a given term $t$ in the archive of a server $i$, and by $\mathrm{TOTOCC}_i$ the highest number of occurrences among all the terms in the archive. For each term $t$ the following "importance" weight can then be evaluated:

$$\mathrm{IMP}_i(t) = \mathrm{TOTOCC}(t)_i/\mathrm{TOTOCC}_i \tag{4}$$

The terms considered as the key concepts are selected as the ones with an importance weight over a given threshold $\tau$. For a given server $S_i$ then a prototypal fuzzy set is defined:

$$\mathrm{PROT}(S_i) = \sum \mathrm{IMP}(t)_i \Big/ t \tag{5}$$

When a query has to be evaluated, the process of estimate of the appropriateness of each server as the collector of relevant information for the query can be done on the basis of a procedure which performs a matching between the query and the "prototypal" fuzzy set of terms. The matching procedure of a Boolean query against the fuzzy prototypal term set proceeds in a bottom up way: first, the membership of each query term to the prototypal fuzzy set is computed. The obtained values (indicating the fitness of the server for each query term) are then aggregated by applying the connectives specified in the query (the AND is interpreted as a min, the OR as a max, and the NOT as a complement).

The obtained value represents the fitness $b_i^*$ of the server $S_i$ with respect to the considered query.

Let us consider as an example the query $q =$ "$t_1$ AND $t_2$" and three servers $S_i$ with $i = 1, \ldots, 3$ represented by the following prototypal fuzzy sets:

$$\mathrm{PROT}(S_1) = \{0.3/t_1 + 0.9/t_2 + 0.6/t_3\}$$
$$\mathrm{PROT}(S_2) = \{1/t_1 + 0.3/t_3\}$$
$$\mathrm{PROT}(S_3) = \{0.5/t_1 + 1/t_2\}$$

The fitness scores $b_i^*$ with $i = 1, \ldots, 3$ of the servers with respect to the query $q$ are then estimated as follows:

$$b_1^* = \min(0.3, 0.9) = 0.3 \quad b_2^* = \min(1, 0) = 0 \quad b_3^* = \min(0.5, 1) = 0.5$$

### *2.2.2. Approach based on the specification of a set of "query type" for each server*

This approach is based on the definition of a set of "prototypal queries" for each server. A "prototypal query" is conceived as a high level description of the key concepts summarizing the information in the archive on the server. A prototypal query is simply defined by means of two sets of terms; the first set (denoted as NNT) contains the key terms that the server deals with, while the second set (denoted as NT) contains the key terms that the server does not deal with. If for example the documents in the archive deal with the applications of computer science to medicine, but not with the applications related with image processing, the two following sets could be defined: NNT = {computer science, medicine, medical applications, expert systems} and NT = {image, image processing, image analysis}. To make more refined the role of prototypal queries as carriers of the information contained in the server, a fitness score is associated with each of them. This score expresses the degree of concern of the considered server with the topics highlighted in the prototypal query. As described before, the fitness score is defined as a numeric value in [0,1].

We consider queries formulated by means of the Boolean query language. When a given query is considered, to evaluate the fitness of server $S_i$ with respect to it the following process is applied. First two sets are defined: the set NNT which contains the terms not negated in the query, and the set NT which contains the terms negated in the query. As a second step, for each prototypal query $\mathrm{P}_{qk}$ in the server the intersection of NT with the set of negated terms in the prototypal query ($\mathrm{PQNT}_k$) is computed, thus obtaining the set $\mathrm{INT}_k$, and the intersection of NNT with the set of not negated terms in the prototypal query ($\mathrm{PQNNT}_k$) is also computed, thus obtaining the set $\mathrm{INNT}_k$. To obtain a fitness score for the considered query with respect to a prototypal query the following rule is applied:

$$\text{If } most(|\mathrm{INNT}_k|/|\mathrm{PQNNT}_k|) \text{ AND } most(|\mathrm{INT}_k|/|\mathrm{PQNT}_k|) \text{ then } V_{ik} = b_{ik}$$

in which $|S|$ is the cardinality of the set $S$, and $b_{ik}$ is the fitness score of server $S_i$ for the prototypal query $P_{qk}$. This rule verifies if most of the query terms are contained in the prototypal query. *Most* is a proportional increasing linguistic quantifier formally defined as a fuzzy set on the interval [0,1] [17]. The evaluation of the antecedent of a rule produces a firing level $\tau_{ik}$. To obtain the overall fitness score $b_i^*$ of the server $S_i$ for the considered query, the following formula is applied:

$$b_i^* = \frac{\sum_{k=1}^{p} \tau_{ik} b_{ik}}{\sum_{k=1}^{p} \tau_{ik}} \tag{6}$$

in which $p$ is the number of prototypal queries defined for a given server $S_i$.

### 2.2.3. Rule-based approach

The kernel of this approach is the identification of a set of variables useful to characterize a query in terms of “properties”. The definition of a prototypal query is obtained by instantiating each variable with a selected value. For each prototypal query each server has a corresponding fitness score. In the approach proposed by Yager and Rybalov, the score is selected "a priori" for each prototypal query, and encodes the appropriateness of the source as a collector of the relevant information with respect to the prototypal query. In this way the information role played by each information source is directly encoded in the numeric fitness score. A more flexible but complex approach could also model the preliminary phase of computation of these scores.

The main difficulty related to the knowledge-based approach is the identification of properties useful to characterize queries. A consideration useful to identify some properties is that they are strongly related with the adopted query language. In particular the properties should be selected in order to highlight the information content of the query. To this aim, it is necessary to consider the level of the "information carriers" in the query expressions; in the query languages of current IRSs this role is played by terms.

In [11–13] an example of application of the knowledge-based approach is discussed. In this example, by following the approach specified by Yager and Rybalov each prototypal query corresponds to a rule; in the antecedents of each rule the variables with their associated values are specified. An example of rule defined by considering the variable "term importance" is the following:

**If** $t_1$ is *important* and $t_2$ is *very important* and $t_3$ is *not very important* **then V** is $\mathbf{B}_k$ in which the $t_i$ are terms, $k$ is the index of the rule, **V** is an $m$-dimensional vector whose components are the $V_i$, and $\mathbf{B}_k$ is also an $m$-dimensional vector whose components, $b_{ki}$, correspond to the fitness of server $S_i$ as an appropriate source of documents for the type of query described in the antecedent of rule $k$.

To match the antecedent of a rule with a query, the compatibility between the numeric query term weights (of query terms appearing in the antecedent) and the linguistic weights is evaluated on the basis of the weight semantics.

To obtain a more flexible modelling the antecedents of rules could be specified by applying a linguistic quantifier to aggregate the single terms considered, like in the following example:

**If** *most* ($t_1$ is *important*, $t_2$ is *very important*, $t_3$ is *not very important*) **then V** is $\mathbf{B}_k$ in which *most* is defined as an ordered weighted averaging operator [18].

### 2.2.4. Flexible approaches to fuse distinct document lists

We have a collection of m ordered lists of distinct elements and we desire to fuse these desired lists into one ordered list $L$. Here we use $L_i$ to indicate the $i$th individual list and $N_i$ to indicate the number of elements in $L_i$ where

$$i = 1, \ldots, m; \quad \sum_{i=1,\ldots,m} N_i = N, \quad \text{the number of elements desired in } L.$$

In forming the combined list $L$ from the $L_i$ there is clearly one property that must be satisfied, *intra–list consistency*. This condition requires that if $x$ and $y$ are two documents appearing in list $L_i$ and if $x$ is higher then $y$ in this list, $x >_i y$, then it is required that in the fused list $L$ to $x$ is higher than $y$, $x >_L y$.

In order to combine these individual ordered lists we need some additional guiding imperative to form the aggregated list. In developing a method for combining the lists we note that there is only one distinction between the lists, the number of elements in each list $N_i$. In the following we shall discuss a general approach [13,16] to this list fusion problem which uses the number of elements in a contributing list.

In the following we let $x_{ij}$ indicate the $j$th document in $i$th list. We shall associate with each $x_{ij}$ a value $V_{ij}$ and place the documents in the fused list $L$ based upon this value, the higher the value of $V_{ij}$ the higher the position in the fused list. The formula we shall use for $V_{ij}$ is

$$V_{ij} = \alpha N_i + 1 - j, \quad \text{where } \alpha \geqslant 0 \tag{7}$$

As we subsequently see $\alpha$ is a parameter which determines the type of list fusion we are implementing. We note that documents from the same contributing list will always have distinct values, $V_{ij} \neq V_{ik}$ for $j \neq k$. In addition we note that the intra-list consistency property is always satisfied, if $j < k$ ($x_{ij}$ higher in list $i$ then $x_{ik}$) then $V_{ij} > V_{ik}$ and therefore $x_{ij}$ will be higher in the fused list then $x_{ik}$. It is possible to have ties between documents from different contributing list. In order to address this problem we shall use the following deterministic tie breaking procedure. We index the individual collections in descending order of the total number of documents they are going to contribute to the fused collection; thus if $S_i$ and $S_j$ are two servers, we shall assume $i < j$ if $N_i > N_j$.

Furthermore, if there are any ties with respect to the number of documents individual collections are going to contribute we shall distinguish these by alphabetically ordering the tied collections; thus if $S_i$ and $S_j$ are such that $N_i = N_j$ we shall assume $i < j$ if the name of $S_i$ appears higher in the alphabet then $S_j$. Using this indexing if at some point in the fusion process there is a tie with respect to the $V$ values we select the next document from the collection with the smallest index. Using this tie breaking procedure results in our fusion process always being deterministic for a given query. Let us look in more detail at the fusion function and try to understand the role of the parameter $\alpha$. As we subsequently see $\alpha$ essentially determines the type of fusion procedure we are using. We shall first consider the case in which $\alpha = 0$. In this case our fusion function becomes $V_{ij} = 1 - j$. Here fusing the lists by treating the lists so that its top elements are equivalent. Consider now the case in which $\alpha = 1$, here our fusion function becomes $V_{ij} = N_i + 1 - j$. In this approach we essentially push all of the individual orderings up so that their bottom elements are equal. Essentially this approach can be seen to be one in which the potential of a list $L_i$ providing the next element for the fused list is determined by the number of documents remaining in it.

In the two approaches previously discussed one consisted of a moving down of collections so that the worst elements are on the same level ($\alpha = 1$) while the other is one in which all collections are equalized by moving the smaller ones up so that all top elements are equal ($\alpha = 0$). With $\alpha = 0.5$ we essentially center the collections and fuse them.

Next is the case in which $\alpha$ is very large, we shall denote its value as **Big**. In this case our fusion function becomes $V_{ij} = \mathbf{Big}N_i + 1 - j$. Here we see that we first take all the documents from the contributing list with the largest number of elements, then we take all the documents from the contributing list with the next number of documents etc. Essentially the imperative used here is one that says if $N_i > N_j$ then all the documents in $S_i$ are more relevant than those in $S_j$.

## 3. Metasearch retrieval

The huge amount of documents distributed over different sites and accessible on the Internet can be retrieved by searching through one of the many available search engines such as AltaVista, Excite, Lycos, HotBot, Infoseek, Yahoo, Google, just to cite some of them. It often happens that, if one submits the same request to different search engines, one retrieves distinct ranked lists of documents which differ in both the number of retrieved items, and in the order in which these items are judged relevant. A document can then appear in more then one of the retrieved lists and in a different position within each list. This is due to many factors peculiar to the search engine among which the most common ones are the different indexing, updating and retrieval criteria adopted to represent the accessible document collection and to compute the retrieval status value (RSV). Then, the RSVs of a document produced by distinct search engines are not comparable one another since they are computed based on different indexing and matching criteria [5,19,20].

The retrieval result, which would be much desirable to a user, is an overall ranked list of documents, obtained by the fusion of the single lists produced by each search engine.

Based on the consideration that more experts are better than one, the idea is that by using more search engines, regarded as experts in finding information, one achieves better results [19,21]. To this aim in the literature several approaches have been proposed [2,3,5–7,13,22,23]. In the following subsection a soft approach to perform the fusion of overlapped ranked list of documents yielded by distinct search engines proposed in [5] and successively refined in [6] is presented.

### 3.1. Soft fusion of overlapping ranked lists

In this subsection a soft fusion of overlapping ordered lists into an overall ordered list is regarded as a group decision making activity in which the search engines play the role of the experts, the documents are the alternatives that are evaluated based on a set of criteria expressed in a user query, and the decision function is a soft aggregation operator modeling a specific user retrieval attitude. Let us formalize the entities involved in this decision activity. Let us indicate by $q$ a request for information expressed by a single word or by a Boolean expression on words (words or phrases connected by the AND, and the OR and negated by the NOT operators). It constitutes the set of criteria by which the alternatives must be evaluated.

Let us indicate by $E_1, \dots, E_K$, the set of the $K$ search engines producing the lists $L_{1q}, \dots, L_{Kq}$ of documents as a result of the query $q$ evaluation, with $|L_{iq}|$ the number of elements in the $L_{iq}$ list. If $|L_{iq}| = 0$ it means that the $L_{iq}$ list is empty, that is the $i$th search engine did not found any document satisfying the query $q$.

The set of the alternatives is constituted by the set of documents $D$, uniquely identified by their www addresses hereafter indicated by $d_1, \dots, d_N$, appearing in at least one of the retrieved lists:

$$D := \{\mathrm{d} | \mathrm{d} \in \cup_{j=1,\dots,K} L_{jq}\} \tag{8}$$

Each document $d_j$ has a position $p_{ji} = l_{iq}(d_j)$ in the $i$th list $L_{iq}$ produced by the evaluation of a query $q$. We assume that $p_{ji} = l_{iq}(d_j) = 0$ when document $d_j$ is not present in the $i$th list or when the $i$ list is empty:

$$\text{if } |L_{iq}| = 0 \text{ then } l_{iq}(d_j) = 0 \quad \forall d_j \in D$$

Since the RSVs computed by distinct search engines are not comparable with one another, we define the criteria performance judgements of the documents evaluated by the search engines with respect to the criteria in the query just through the ranks of the retrieved documents in the lists. Specifically, we define as criteria performance judgements of each alternative $d_j$ expressed by the $K$ search engines $K$ scores $C_{j1}, \dots, C_{jK}$ in the range [0,1] obtained as follows:

$$C_{ji} = (|L_{iq}| - p_{ji} + 1) \tag{9}$$

The performance judgement of a document with respect to a search engine decreases as the position of the document in the list goes from the top to the bottom of the list. $C_{ji} = 1$ if $d_j$ is the first document in the list $L_{iq}$.

If $|L_{aq}| < |L_{bq}|$ and $p_{ja} = p_{jb}$ then $C_{ja} < C_{jb}$: getting a good position in a long list of candidate relevant items is considered more valuable than a good position in a short list [6].

Given $K$ search engines, for each query a matrix $C = [C_{ji}]$, with $j = 1, \dots, |D|$, and $i = 1, \dots, k$, can be computed in which the columns corresponds with the search engines and the rows with the set of documents $D$. Once the performance judgements of the criteria have been computed by all the experts, an aggregation phase takes place aimed at computing the overall performance of each alternative reflecting the judgement of a fuzzy majority of the experts. In our case this phase corresponds with the soft fusion of the overlapping ranked lists. The overall performance judgement $C_j$ of document $d_j$ is computed by aggregating the values in the $j$th row of the $C$ matrix; it expresses the consensual RSV of a fuzzy majority of the search engines on document $d_j$. This value is used to rank the document in the fused list, or consensual list. Like in the fuzzy multi-agent decision making model defined in [24] we adopt a quantifier guided aggregation function to define the concept of a fuzzy majority expressed by a linguistic quantifier $Q$ such as *all, most, at least a few, at least one*.

These relative monotone non-decreasing quantifiers specify weaker and weaker interpretations of a fuzzy majority and in this context model distinct retrieval attitudes.

*All* corresponds with the strictest majority that requires all search engines to retrieve a document for it to appear in the fused list. This aggregation criterion expresses the need for a precise (focalized) answer since the set that the user wants to consider is obtained by the intersection of the search engines lists. In this case we can say that the user is pessimistic on the ability of the individual search engines to select only relevant items and then, moved by the aim of increasing the retrieval precision, he/she specifies the most selective decision function as fusion criterion. *Most* expresses a fuzzy majority that does not impose the unanimous retrieval of a document for it to be considered in the fused list; for instance, it is sufficient that it is present in at least 80% of the lists. *At least a few* corresponds with a weaker aggregator; it is sufficient that a document is retrieved by at least a minority of the search engines, a few of them (for instance at least 20%), for the document to appear in the fused list, and finally *at least one* is the weakest fusion criterion that specifies the union of the lists. This last fusion criterion expresses the need for an exhaustive answer; the user does not want to disregard any potentially relevant document, he/she is then moved by a recall oriented retrieval attitude. In this last situation we can say that the user is optimistic on the ability of the individual search engines to select only relevant items, and then, moved by the aim of increasing the recall, he/she specifies the least selective decision function as fusion criterion.

We define a quantifier guided aggregation function associated with a linguistic quantifier by means of the IOWA operators [25]. These operators allow to distinguish the order used to take into account the contributions of the arguments (the reorder vector) from the arguments values to be aggregated (the arguments vector). What we want to determine is the order in which the contributions of the search engines performance judgements are taken into account depending on a fitness score evaluated for each search engine. By this approach we consider a semantics and consequently a computation of the fitness score different from the one proposed in Section 2.2. The fitness score is in this case associated with a search engine and is computed by a relevance feedback mechanism that will be described in the following subsection. The search engines that exhibit a high fitness score must contribute more heavily to determine the overall performance of a document.

Formally, a quantifier guided aggregation function can be defined by means of an IOWA operator associated with a monotone non decreasing linguistic quantifier $Q$: the weighting vector $W$ is obtained from the fuzzy set defining $Q$ by computing $w_i = Q(i/K) - Q(i-1)/K$ for $i = 1, \ldots, K$ and $K$ IOWA dimension, in this case the number of the search engines [13]. Then, the IOWA operator is defined as

$$\mathrm{IOWA}(\langle u_1, C_{j1}\rangle, \ldots, \langle u_k, C_{jk}\rangle) = W^{\mathrm{T}} B_U \tag{10}$$

in which $W^{\mathrm{T}}$ is the transpose vector of $W$; $U = [u_1, \ldots, u_k]$ is the reorder vector: in this case $u_h$ is defined based on the fitness score of the $h$th search engine and the performance judgement of document $d_j$ by the $h$th search engine $C_{jh}$; $B_U$ is the arguments vector $[C_{j1}, \ldots, C_{jk}]$ reordered according to the $U$ vector so that $b_i$ is the $C_{jh}$ having associated with it the $i$th largest $u$ value. For example, given the IOWA with $W = [0, 0.5, 0.5, 0]$ and the argument vector $[\langle 3, .1\rangle, \langle 8, .3\rangle, \langle 6, .8\rangle, \langle 2, 1\rangle]$ we obtain: $\mathrm{IOWA}(\langle 3, .1\rangle, \langle 8, .3\rangle, \langle 6, .8\rangle, \langle 2, 1\rangle) = 0 * 0.3 + 0.5 * 0.8 + 0.5 * 0.1 + 0 * 1 = 0.45$.

Since we want that the search engines with highest fitness scores have a greater chance to determine the result than search engines with lowest fitness, given the orness of the aggregator operator we compute the reorder vector $U$ as follows: for the IOWA operators with $\mathrm{orness}(W) > 0.5$, i.e. with a semantics closer to the OR (the IOWA operators associated with the quantifiers *at least 1* and *at least a few*), we compute the reorder vector as

$$u_{jh} = 1 - dist\Big((C_{jh} * f_h)/(\max L_q)^2, \mathrm{orness}(W)\Big) \tag{11}$$

in which $f_h \in [0, \max L_q]$ is the fitness of the $h$th search engine, $\max L_q$ is the maximum cardinality among the $|L_{hq}|$, i.e., $\max_{h=1,\ldots,k} |L_{hq}|$, and *dist* is a distance function (ex $dist(a, b) = |a - b|$). In this way, the overall relevance judgement of a document is computed based on an optimistic decision function that considers the best relevance evaluations of the search engines with highest fitness to be the most determining.

For the IOWA operators with $\mathrm{orness(IOWA)} \leqslant 0.5$, i.e. with a semantics closer to the AND (the IOWA operators associated with the quantifiers *all* and *most*) we compute the reorder vector as

$$u_{jh} = dist\Big((C_{jh} * (\max L_q - f_h))/(\max L_q)^2, \mathrm{orness}(W)\Big) \tag{12}$$

In this way, the overall relevance judgement of a document is computed based on a pessimistic decision function that considers the worst relevance judgements of the search engines with highest fitness to be the most determining.

### 3.2. Computation of the search engines fitness scores

We have seen that in multi-source information retrieval the notion of fitness score is associated with a source of information, and denotes the appropriateness of the source information contents with respect to a user query. The computation of such a score may depend on both the type of the query and the characteristics of information source itself. This score estimates how much the content of the information source is potentially relevant to the topics represented by the query. In the context of metasearch engines the fitness scores are generally computed to serve two different purposes: either to have a priority criterion for using a search engine instead of another one, or to determine the number of documents to be selected from each ranked list. In the soft approach proposed in Section 3.1 the fitness score has a different meaning. It is intended as a preference accorded to a search engine by the user on the basis of the engine's successful behaviour in retrieving documents relevant to his/her information needs. The basic idea is that it should serve a twofold purpose: the most preferred engines, those with highest fitness scores, should have a greater chance to determine the final judgement. Further, the fitness of a search engine increases as it retrieves more documents relevant to the user in best positions over a period of time, say after a query or a retrieval session. This allows to compute the fitness score of a search engine based on a long term dynamic user profile.

When the user connects for the first time, at time $t_0$, to the metasearch engine it is possible either to rank the search engines in decreasing order of their preference, this information is used to set specific values for the fitness scores $f_h(t_0) \in [0, \max L_q]$ $\forall h = 1, \ldots, k$. In the case of complete uncertainty, all the fitness scores can be fixed equal to the average cardinality of the lists $f_h(t_0) = \sum_{h=1,\ldots,k}(|L_{hq}|)/k$. Then, after analyzing the results produced after a first query evaluation, the user can identify those candidate documents that he/she considers relevant to his/her needs. These documents are assigned a utility factor $s \in (0, 1)$, hereafter interpreted as a learning rate, that is used to modify the fitness of the $h$ search engine as follows:

$$f_h(t) = \min[f_h(t-1) + s * C_{rh}(t-1), \max L_q] \tag{13}$$

in which $f_h(t)$ is the fitness score at time $t$, $\max L_q = \max_{h=1,\ldots,k}(|L_{hq}|)$ and $C_{rh}(t-1)$ is the relevance judgement of the relevant document identified by the user in the last run. In this way, the fitness score increases with the user satisfaction proportionally to the relevance judgement of document $r$ really relevant to the user. The greater the $s$ learning rate, the faster is the increasing of the fitness score to the maximum allowed equal to the maximum cardinality of the lists $\max L_q$. On the other side, the user can also indicate among the retrieved documents those that are not relevant at all; this information is used to decrease the fitness scores of the search engines proportionally to the relevance judgements of the irrelevant document down to a minimum bounding value of zero as follow:

$$f_h(t) = \max[f_h(t-1) - s * C_{rh}(t-1), 0] \tag{14}$$

## 4. Conclusions

In this paper some approaches to deal with the problem of distributed information retrieval have been analysed. In particular two situations have been considered. The problem of querying multiple disjoint collections through a distributed information retrieval system and the problem of metasearch retrieval, i.e., the querying of documents distributed on a network by means of distinct search engines indexing the whole collection. The peculiarities of the two situations are outlined and some soft approaches to deal with the processes they imply are described.

## References

[1] J. Callan, Distributed information retrieval, in: W.B. Croft (Ed.), Advances in Information Retrieval, Kluwer Academic Pub., 2000, pp. 127–150.

[2] J. Baker, Meta-Search Engines, UC Berkley Teaching Library Internet Workshop. Available from <http://www.lib.berkeley.edu/Teaching>Lib/Guides/Internet/MetaSearch.html>.

[3] B.T. Bartell, G.W. Cottrell, R.K. Belew, Automatic Combination of Multiple Ranked Retrieval Systems, SIGIR 1994, Dublin, abstract.txtsigir94/sig, 1994.

[4] N.J. Belkin, P. Kantor, C. Cool, R. Quatrain, Query combination and data fusion for information retrieval, Proc. of the Second Text Retrieval Conf. (TREC-2), 1993, pp. 35–44.

[5] G. Bordogna, SOft Fusion of Information Accesses, Proc. of the IEEE Int. Conf. On Fuzzy Systems 2002, Honolulu, 24–28 May 2002.

[6] G. Bordogna, G. Pasi, A Model for a SOft Fusion of Information Accesses on the Web, Fuzzy Sets Syst., submitted for publication.

[7] E. Selberg, O. Etzoni, The MetaCrawler architecture for resource aggregation on the Web, IEEE Expert 12 (1) (2000) 8–14.

[8] T. Uben, J. Delgado, Advanced Metasearch of News in the Web, Proc. of the Int. Conf. On Electronic Publishing, 2002.

[9] J.P. Callan, Z. Lu, W.B. Croft, Searching distributed collections with inference networks, Proc. 18th Annual Int. ACM SIGIR, Seattle, 1995, pp. 21–28.

[10] O. de Kretser, A. Shimmin, J. Zobel, Methodologies for Distributed Information Retrieval, ICDCS 1998, 1998, pp. 66–73.

[11] G. Pasi, R.R. Yager, Document retrieval from multiple sources of information, in: B. Bouchon-Meunier, R.R. Yager, L. Zadeh (Eds.), Uncertainty in Intelligent and Information Systems, World Scientific, 2000.

[12] R.R. Yager, A. Rybalov, Retrieving documents from multiple information sources, in: D. Mancini, M. Squillante, A. Ventre (Eds.), New Trends in Fuzzy Systems, World Scientific Pub., Singapore, 1998, pp. 3–22.

[13] R.R. Yager, Merging lists of documents retrieved from multiple web sources, Proc. of the International Conference on Fuzzy Information Processing, Beijing, 2003.

[14] M. Delgado, F. Herrera, E. Herrera-Viedma, M.J. Martin-Bautista, M.A. Vila, Combining linguistic information in a distributed intelligent agent model for information gathering on the Internet, in: P.P. Wang (Ed.), Computing with Words, John Wiley & Son, 2001, pp. 251–276.

[15] M. Delgado, F. Herrera, E. Herrera-Viedma, M.J. Martin-Bautista, L. Martinez, M.A. Vila, A communication model based on the 2-tuple fuzzy linguistic representation for a distributed intelligent agent system on Internet, Soft Comput. 6 (2002) 320–328.

[16] R.R. Yager, A. Rybalov, On the fusion of documents from multiple collection information retrieval systems, JASIS 49 (1998) 1177–1184.
[17] L.A. Zadeh, A computational approach to fuzzy quantifiers in natural languages, Comput. Math. Appl. 9 (1983) 149–184.
[18] R.R. Yager, On ordered weighted averaging aggregation operators in multi criteria decision making, IEEE Trans. Systems, Man Cybernet. 18 (1) (1988) 183–190.
[19] E.A. Fox, M.P. Koushik, J. Shaw, R. Modlin, D. Rao, Combining evidence from multiple searches, in: D.K. Harman (Ed.), Proc. of the First TREC conference, 1993, pp. 319–328.
[20] R.M. Loose, L.A.H. Paris, Measuring search-engine quality and query difficulty: ranking with target and freestyle, JASIS 50 (19) (1999) 882–889.
[21] P. Thompson, A combination of expert opinion approach to probabilistic information retrieval, part 1: The conceptual model, Inform. Process. Mgmt. 26 (3) (1990) 371–382.
[22] Z. Chen, X.M. Yarrow, A real-time client side metasearch learner, Proc. of the 2000 AAAAI work, On Artificial Intelligence for Web search (AAAAI00), 2000, pp. 12–17.
[23] J. Liu, Guide to meta-search engines. Available from <http://www.indiana.edu/librcsd/search/meta.html>.
[24] G. Bordogna, M. Fedrizzi, G. Pasi, A linguistic modeling of consensus for a fuzzy majority in group decision making, IEEE Trans. Systems, Man Cybernetics 27 (1) (1997) 126–132.
[25] R.R. Yager, D. Filev, Operations for Granular Computing: mixing words and numbers, FUZZIEEE 1998, Anchorage, 1998, pp. 123–128.