Heterogeneous Data Relevance Ranking Method

Ronald A. Poell, CTO Semantic Network Technologies
Netherlands Organisation for Applied Scientific Research (TNO), Poell@fel.tno.nl

Date 2003-12-15

Abstract

The integration of various search-results from different sources poses the problem of merging the results of different ranking mechanisms. Furthermore, the topics in the results are not necessary of the same kind. Some searches will provide only references to documents whilst others will return heterogeneous kind of data.
This paper gives a short overview of the problems in this domain and proposes a theoretical solution in which the knowledge about the ranking mechanisms used by the various search engines is not necessary.
The solution consists of applying 'the best of two methods' amongst the "calibration of the result space" and the "calibration of the ranking space". Under certain circumstances previous calibrations can be reused.

Introduction

The total information space available to human users (and software agents) becomes more and more heterogeneous in kinds of information.
The information space consists of the following kinds of sources:

- Databases (specific for a domain – e.g. financial, employees, projects)
- Unstructured text documents (e.g. various formats, restricted or public)
- Multimedia documents (e.g. pictures, animated pictures, sound)
- Structured text documents (e.g. XML)
- Software agents interfaces (e.g. providing the cheapest fares)
- Other structured sources like a semantic network (in e.g. RDF, OWL, XTM and other formats)
- …

Each of these kinds of sources has one or more search facilities (probably in the form of applications or services) providing a list of search results with some kind of classification or ranking of the results.

This paper provides material for the integration of search results from various search agents[1] having different ranking methods and covering different kinds of sources. The merging is realised by the "Search Integration Agent" (SIA)[2].
We restrict ourselves to a user's context free, relative comparison mechanisms[3].
This paper is divided into two main sections. In the "Problem space" we provide a description of the major elements that might interfere when integrating data from various sources. The "Solution space" contains the recommended methods, their conditions and arguments for solving the problem.
Previous work has been released in this domain like in [Mendelzon], [Sugiyama] and [Gionis] but they are applied to only a part of the situation we handle here.

Problem space

The total problem space is divided into several areas, each with their own characteristics, axioms and other important aspects. The elements mentioned are descriptive of the area and targeted at a mind setting, not as preliminary to a possible solution.

[1] Search Agent (SA): generic term used in this paper to indicate a facility that provides search capabilities even if the service is not properly spoken a software agent.

[2] Search Integration Agent (SIA): term used in this paper for a service or software agent that combines several search results into one consistent result set.

[3] The use of the context of the user *does* interfere in the final relevance ranking of the results. But this context is identical for all the separated result sets and has no influence on the relative comparison of them. Some search agents though do take the user's context into account and integrate it into their individual ranking. In these cases there is a "hidden" influence of the user's context in the merged results. An upcoming paper addresses the application of the user's context for relevance ranking of heterogeneous information spaces (be it a merged result set or not).

User space
When a user requests information from a search agent what is he (she) really looking for? In some cases the user looks for a (specific kind of) document (e.g. a request addressed at image SA) or a specific service (e.g. request for a flight from Amsterdam to Boston).
But in many cases the ultimate aim of the request is to obtain some (probably particular) information about the subject specified by the request and not the documents that contain information about the subject. Even if somebody is looking for a particular document (providing the title e.g.) there is still some underlying reason why this document is requested.
Unfortunately a huge part of the available information space consists of documents without any semantic annotation about the subject(s) it is handling and even less about the information it is carrying. In spite of the upcoming use of semantic annotations (e.g. using RDF) the challenge remains of using the installed document base by adding this necessary meta data automatically, external to the source.
This aspect is not handled in this context and is supposed to be realised by the search agent or its underlying services (e.g. smart indexing services).

Query space
The query space is the area in which the user can express his information need in a form that can be handled by the SA or SIA he is addressing. It can be expressed either directly in an acceptable form for the SA (e.g. a SQL select statement), but more and more through query space transformation processes.
A consideration that should be taken into account is the information on which the search is realised. It is obvious that a full text index of a document will provide fundamentally different hits than an indexed record set in a database. This difference has a direct impact on the way a user exposes its request. A request to a full text search engine has often a different form (for the user) then a request for structured source as database.
Although transformation processes (broker function) are certainly issues, in particular in the case of an SIA with an input user interface, this aspect is out of scope.

Information space
The information space consists of all *the kinds of content* on which the SA's will operate.
The basic aspect of this area is the variation in kinds of topics returned by a search agent. Some will return the titles and URLs of documents, others will return a database record (or a view on a combination of different records in different tables). As different as these appear at first glance, they appear rather similar when studying the problem more closely.
All the search agents provide a list of information items (title-URL, family name-first name) about a specific kind of topic (document, person).
The most important aspect that needs to be handled is the potential *variation* of the kinds of topics for *each* of the SA's. Some SA's will always provide the same kind of topics (Internet search agents will always return documents). Others may return either different homogenous sets of topics (different database searches each on one view) or heterogeneous sets of topics (an Intranet search agent allowing to find something among persons, projects, processes et cetera, or a search in a semantic network).

For the Internet (and some older forms of Intranets) a specific aspect should be taken into account. We can distinguish two different types of "documents". In the *static* information sub space there are static pages (mostly html, shtml or xhtml) and more and more generated pages (jsp, asp, cgi, php, et cetera) in the *dynamic* information sub space. Although the contents of the dynamic pages is often generated from an underlying database the output is often, as it is for static pages, unstructured or poorly structured text.
As the content of these documents changes over time, one should be careful to use results of this kind. Some search agents handle dynamic pages as if they are static, which gives undesirable results.
In the case the dynamic pages are result pages of other search agents and they are reinterpreted in terms of result sets, they can be used without any problem.

Domain space
This area covers all the available content.
In the domain space there is a fundamental difference between SA operating on an open space and those operating on a closed space[4].

[4] An open space might have a closed set of kinds of topics (the Internet SA's provide only documents) but an open set of subjects. In a closed space there are often only few topics (persons, projects) and often also a limited set of subjects.

A typical example of an open space is the Internet. Documents that occur in this space are of different quality. In the static sub space it varies from a high quality scientific article to a little child's home page with its hobby's mentioned. And in the dynamic sub space we encounter simple lists but also semantically annotated sets of information and more and more full-blown applications. In particular in the static sub space the applied vocabulary is non-controlled, often ambiguous and multi-lingual. Until more semantic annotations are available (e.g. with RDF) the semantics remain highly ambiguous.
Also, in the static sub space, the "absolute" relevance of the information itself is uncontrolled. Old obsolete information remains available for a long time while the new information is also available.
Within a specific closed space the quality of the information can vary between the closed spaces but the variation within the space in general less. The applied vocabulary is more restricted (some times fully controlled) and often monolingual. The structure of the information is implicitly available (e.g. through the database schema) although not often correctly exposed externally. The associated semantics can be expressed in the database's meta data but in most cases resides in the designers head, the associated documentation (e.g. UML diagrams) or even only in the applications using the software[5].
The "absolute" relevance of information[6] is often higher because old information is on a regular basis replaced by new one.

Ranking space

We call the ranking space the area of ranking values (relevance, matching quality) including the ranking methods and algorithms.
Many of the search agents do not expose the ranking method(s) they apply. For document based search engines (like the Internet search engines) the basic method is sometimes described ([Brin], [Ridings], [Cormack], [Muscat]) for others it isn't. In most cases the actually *applied* ranking method is not described in detail (for technical or commercial reasons) and even less discoverable by the Search Integration Agent.
In the case of database search the ranking (if any) is highly dependent on the level of service that is used. On the lowest level, SQL querying, the "ranking" is probably the database's natural ordering or an ordering requested in the query. This is a form of classification but not properly spoken a ranking. Higher levels of database search agents may combine the results of various queries into one result set with some (probably non-exposed) logic.
Independent of the above, the SA can or cannot apply various matching algorithms (exact, partial, word boundary based, stemmed et cetera) and integrate these in the final search result with a specific combined ranking algorithm.
To complicate matters a little more, some product lines authorise implementations to personalise the ranking methods, either through a plug-in possibility for new ranking algorithms, or through parameterisation of existing algorithms. So we need to make the distinction between the ranking method (general form) and the *applied* ranking method (parameterisation of the method).
Furthermore, for maintenance reasons of the SIA, it seems unreasonable to assume that a particular SA will have a stable (applied) ranking method over time. In the dynamic IT world, applied ranking methods change at any time without notification.
Unless standardisation efforts are realised for the exposure of the applied ranking methods, this will remain so for the foreseeable future.

Approach

It seems difficult at this moment to have a clear view on the applied ranking methods of each of the SA's and, if available, to maintain this information accurate over time. For this reason the proposed solution doesn't *need* any

[5] At TNO-FEL a first study has been realized in 2002 [VanWieringen] to see whether semantic equivalence can be (semi) automatically discovered between different data models based on content comparison. We planned to continue to work on this item in the next years.

[6] The "absolute" relevance of information represents the relevance at a specific moment in time. It can be compared to the "absolute" relevance of the same subject at some other time. The "relative" relevance is the relevance in a specific context also at a specific moment in time. For example the absolute relevance of "X is author of Y" is maximal and remains so over time. Its relative relevance in the *global world context* might be high or low depending on the importance of X and Y at a specific moment in time but will probably decrease quite rapidly (world news doesn't remain *news* for very long). Whereas it's relative relevance might be much higher in a *specific domain* and remain so for a longer period (a good scientific paper will remain important until the next (r)evolution on the subject). See also [Poell2001] relevance decay models.

knowledge about the internal ranking methods of individual SA's.
The approach consists of two different calibration methods, applied according to the possibilities of the separated result spaces.

Calibration of the result spaces
The calibration of two result spaces consists of searching identical elements in both spaces. Within homogenous result spaces[7] (e.g. only documents) calibration can occur if three conditions are fulfilled:

- the result spaces must partially overlap
- identification of elements in a result space is possible
- the identifying properties must be semantically equivalent in both spaces

Overlap
When result spaces don't have any result in common there can be no calibration of the result spaces. The same holds if there are in fact identical elements in the result sets but they cannot be identified correctly (see below).

Identification
The identification condition has some pitfalls. In the example of static documents the URL's can be used for identification. If URL's are identical the documents are identical. But if the URL's are different the documents can still be the same. The same content can be represented in different formats (.ps, .pdf, .html) and the same documents can be present in several places (mirrors, backups et cetera). For this example clever algorithms must be able to go beyond the basic URL matching.
What we really look for (in most cases) is the abstract concept behind the document not one of its physical representations [Lagoze].
Similar arguments hold for other kinds of topics. Identification should be realised through a *set* of properties allowing the identification of the abstract concept and for obvious practical reasons also its physical representations.
These sets are called *identifying properties*. The same set will be valid for homogenous result spaces, but in heterogeneous results spaces there will be several sets (one set per kind of topic).

Semantically equivalent identifying properties
When there is a set of identifying properties in one result set, the semantics of it must be mapped to properties in another result set. The various standardisation efforts and the real use of their results will facilitate the mapping efforts. Although some work has been done to automate these mappings [VanWieringen] there are actually no reliable methods to discover the necessary semantic mapping.

Calibration is not possible between two homogenous result spaces with different topics (you cannot find a project with the same identification as a document or at least you shouldn't).

Homogenous result spaces where the topic kind is an abstraction of various heterogeneous topics can be used, under some conditions, in the calibration process. Examples are semantic networks (nodes), Topic Maps (topics) and RDF (resources).
The SA providing such a result space must also provide with the sets of identifying properties the topic kind and these kinds must semantically match topics kinds in other result spaces (e.g. document, person)
When these conditions are fulfilled, these result spaces are extremely useful because they allow result spaces with different topics to *be cross-calibrated*, enabling a *complete* relevance ranking across heterogeneous data.
At the top level the abstract concepts are ranked. The topic kind property allows the creation of ranked homogenous sub spaces that are calibrated with each other (same ranking space). Each of these sub spaces can then be calibrated with the other result spaces.

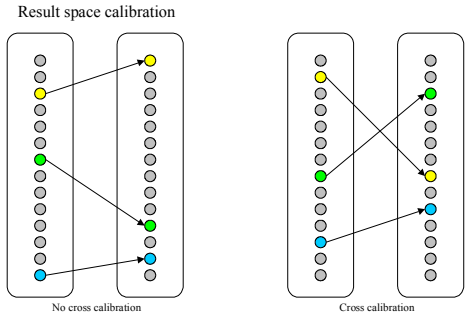

Figure 1. Result space calibration with and without cross calibration

[7] A homogenous result space contains only topics of the same nature: documents or persons (but not both). A heterogeneous result space contains more than one kind of topic (documents, persons and knowledge areas). A result space that contains a homogenous abstraction of heterogeneous topics is also called heterogeneous.
Two homogenous result spaces can be heterogeneous when they are compared to each other: one with only documents, the other with only persons.

When calibration can occur then we have to face different situations (Figure 1).
In the most straightforward situation there will be no cross calibration (all the calibration points occur in the same order in the result sets).

The most natural solution for integration of two result sets with no cross calibration, is a linear normalisation of the results between two calibration points and merging of the sub sets between two calibration points (Figure 2 and 3).

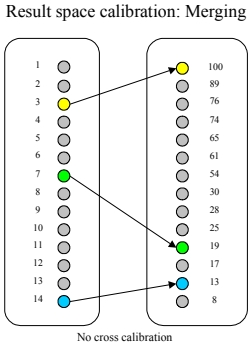

Figure 2. Result space calibration without cross calibration. The left set has an order ranking, the right set is in percentage.

Merging the two result sets from figure 2 is realised in 4 steps. The top results in the left set and the bottom result in the right set are simply added respectively at the top and bottom of the final result.

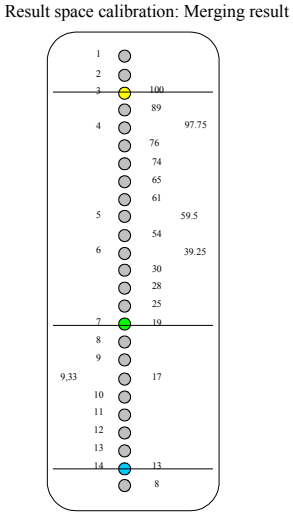

Figure 3. Result space calibration without cross calibration. Result set after merging. The merging process consists of 4 separated steps: top, middle (left into right), middle (right into left and bottom.

The two sections in the middle are each separately merged by linear normalisation of the rankings between the yellow and green results and the green and blue results respectively.[8]
In the case the remaining top and bottom sets are both not empty, should apply, if possible, for these parts the ranking space calibration (see below).
When ranking space calibration cannot be applied the results of the set with the smallest number of results can be evenly distributed in the ranking space of the area that has the most remaining results.

The situation with no cross calibration will probably occur when internal ranking algorithms are quite close and the sub information space is limited. In most cases you will face the situation where cross calibration does exist.
The solution we recommend for cross calibrations consists of the isolation of the sections in the result spaces that represent calibration crossings and handle them with the Ranking Inversion Method (RIM) (Figure 4).

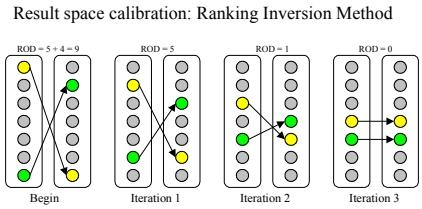

Figure 4 Ranking Inversion Method. The Relative Ordering Difference (ROD) of a crossing equals the sum of the number of results between the results that causes the crossing.

In this method the ranking (position) of the results that form the crossing are exchanged with the result just below or above (ranking inversion). Each moved result will get the ranking of the result it is exchanged with. This iterative process on both spaces terminates when the two spaces have no crossings any more (in the example above after 2.5 iterations).
When no crossings remain the two result spaces can be merged as described for spaces that didn't had any crossings at all.

When the exchange is done alternatively in the right and left sets the relative quality of each of the rankings is considered equal (both sets are

[8] For implementation reasons only the smallest sub set should be integrated in the biggest.

hashed in the same amount). The RIM application factor is 1:1. (the RIM is applied in each space the same number of times = standard application factor).[9]

If there are reasons to believe that one of the rankings is better than the second one, we can apply different factors. The space with the best ranking will completely dominate the final ranking when we use a factor of 1:0.
The typical use of an application factor of 1:0 occurs when you want to calibrate a result space with a particular ranking mechanism with another result space providing only an alphabetical sort order (but no ranking)[10].

As mentioned above, in order to define an application factor other than the standard one, you must dispose of information about the relative quality of the ranking provided by the SA. This will, in almost all cases, involve some user quality evaluation feed back loop. Be also aware that the IT is a dynamic world (cf the problem space) and search agents will change their ranking methods over time. The determination of an application factor other than the standard should also be a dynamic, preferably automated, process. But quality judgement implicates the definition of (standard) criteria and consensus about the usage of the results and we are still far away from that point for search agents.

The Relative Ordering Difference (ROD) of a crossing equals the sum of the number of results between the results that causes the crossing (figure 4).
If more than one crossing exist in two result spaces the RIM should be applied first to those crossings that represent the smallest ROD. Both sets will have the smallest amount of mutations when all the crossings are solved[11].

When more than 2 result spaces must be combined the RIM should be applied first according to the following selection criteria for the combinations of two result spaces:

1. sets of spaces that have the lowest number of crossings
2. sets of spaces that have the smallest total sum of ROD's

The selection algorithm should be applied after each merging of two result spaces. The previously merged spaces form one of the new spaces participating.

When calibration of the result spaces has occurred, automatically the calibration of the ranking spaces has been realised (see below).

The context of this calibration is the query space originally provided by the user to the SIA (or to the different SA's). But what is the value of this calibration outside this specific query space?

When the exposed order is based on a real ranking mechanism (as opposed to a simple sorting mechanism) the calibration can be applied to other query spaces but there is a restrictive element to be considered.
When one of the ranking methods exposes only the order (even based on a real ranking mechanism), the calibration is highly dependent on the compared relative degree of domain coverage[12] of both SA's. The calibration of 10 versus 1.5 million documents is entirely different from a calibration between 1500 and 2000 documents. When both ranking methods expose their ranking values in absolute form it is far more likely that the application of the calibration in other query spaces provides useful results.

Calibration of the ranking space

There are three different forms of *ranges* of real ranking spaces: percentage, absolute value and order.
Three examples from Internet search engines:

- Lycos provides a relevance value in percentage.

[9] To be exact there might be a difference of 1 in the number of times the inversion is applied (like in the example of figure 4): $NumberInversion_{S1} <= NumberInversion_{S2} - 1$

[10] The integration of any result space with no ranking (e.g. a sorted result space) will always decrease the overall quality of the final result space.

[11] At an implementation level we tried out several variants braking this rule for performance reasons. The final results sets are as expected slightly different but we couldn't find any "hard" arguments to judge these differences.

[12] The degree of domain coverage is an indicator how complete the result space is. This is a value that can never be evaluated exactly. Some methods, like the one used to define biocenosis in ecology, might give though a good appreciation of the absolute values. The compared relative degree of domain coverage doesn't need the absolute values because it is defined as the absolute value of the difference in the sizes of both result spaces, divided by the biggest of both sizes.
$C_{ddc} = |(RS_1 - RS_2)/Max(RS_1,RS_2)|$

- Google gives a ranking in absolute inverse values (lowest value is the highest ranking)[13].
- Netscape, Fast and others expose only the order of the results without giving access to the underlying ranking mechanism.

Result spaces, in particular from low level SQL based SA's, may have an ordering that looks like a ranking but is in fact nothing more than a sorting (alphabetical, numeric).

When there is no calibration of result spaces, e.g. when the SA's cover completely different domains, there is some useful way to calibrate the ranking spaces instead.
The first necessary condition is that the SA's expose the *absolute ranking value*. The second condition is the capability by the SIA to discover the *ranges of these absolute values*. This discovering can be straightforward (i.e. percentages, or Google's ranking representation 1-10) or really discovered by looking at ranking values in the various result spaces over time (Ranking Space Range Calibration – RSRC).
A full ranking space calibration should also use evaluations in terms of quality of the other elements in the ranking space (ranking method, applied ranking method). As the evaluations are most of the time not available we only use the RSRC as a less perfect but pragmatic solution that has the advantage to be independent of knowledge about the ranking mechanisms of each individual search agent.

If no other information is available the Ranking Space Range Calibration uses a linear normalisation of the ranges. For search agents like Google the page rank (range 0.00000001 to infinity) will not provide the desired results because it is not a linear scale. It's page rank *representation* (range 1 to 10) will [Brin].

Calibrations based on the RSRC method are less accurate than the ones obtained through the calibration of result spaces because it uses only the *range* of the ranking space and really applied ranking algorithms are not taken into account.

The Ranking Space Range Calibration is also applied to homogenous result spaces with heterogeneous topics (when there is no calibration of result spaces possible), as each of the sub sets is in fact a separated homogenous result set.

[13] Google uses internally a page rank from 0.00000001 to infinity that is mapped to a page rank *representation* between 1 and 10 [Brin].

Conclusion

Despite the complex problem area and the lack of insight in the various ranking mechanisms applied by search agents, it remains possible under certain circumstances, to merge coherently several results into one ranked result set.
In overlapping homogenous result spaces, the *calibration of result spaces* will provide the best results.
Previous obtained result space calibrations can be reused on other result spaces from the same search agents if the ranking values are exposed.
If only the ranking order is exposed, reuse of previous calibrations is only useful when the compared relative degree of domain coverage by the search agents is high.
If calibration of result spaces is not available, *calibration of the ranking spaces* can be applied on result spaces that expose absolute ranking value. The *ranking space range calibration* can be applied if the ranking ranges can be discovered. The RSRC can be applied to non-overlapping homogenous result spaces and on homogenous result spaces with heterogeneous topics.
The calibration of ranking spaces will provide less accurate results than the calibration of result spaces.
Homogenous result sets with heterogeneous topics (RDF, semantic networks, Topic Maps) will play a major role in the merging of heterogeneous data through cross-calibration.
When none of these techniques can be applied, no coherent ranked final result space can be obtained.

References

[Brin] Sergey Brin and Lawrence Page. *The Anatomy of a Large-Scale Hypertextual Web Search Engine*
http://212.25.163.79/pagerank/documents/googlerank01.pdf

[Cormack] G. V. Cormack, C.L.A. Clarke, C.R. Palmer and D.I.E. Kisman. *Fast Automatic Passage Ranking (MultiText Experiments for TREC-8)*

http://www.ai.mit.edu/people/jimmylin/papers/Cormack99.pdf

[Gionis] Aristides Gionis, Surajit Chaudhuri, Sanjay Agrawal and Gautam Das. *Automated Ranking of Database Query Results* http://www-diglib.stanford.edu/~gionis/papers/cidr03.pdf

[Lagoze] C. Lagoze, J. Hunter. *The ABC Ontology and Model*, Journal of Digital Information, Special Issue - selected papers from Dublin Core 2001 Conference http://metadata.net/harmony/JODI_Final.pdf

[Mendelzon] Alberto O. Mendelzon and Davood Rafiei. *An Autonomous Page Ranking Method for Metasearch Engines* http://www2002.org/CDROM/poster/48.pdf

[Muscat] Muscat Limited. *The Muscat Difference Explained* http://www.infoagent.nl/nl/products/FTSExplained.pdf

[Poell2001] R.A. Poell, *The Semantic Network of IKM-I3*. Proceedings Knowledge Technologies Conference, Austin Texas, 2001, slides: http://www2.gca.org/knowledgetechnologies/2001/ proceedings/Poell%20Slides.ppt

[Ridings] Chris Ridings. *PageRank Explained* http://212.25.163.79/pagerank/documents/googlerank02.pdf

[Sugiyama] Kazunari Sugiyama. *A Method of Re-ranking Web Search Results Using their Hidden Hyperlink Structure* http://www.cs.ust.hk/vldb2002/VLDB2002-proceedings/papers/S34P14.pdf (Slides)

[VanWieringen] C.J. van Wieringen. *Searching for Semantically Equivalent Relations in a Semantic Network* (unpublished), TNO-FEL, 2002