



Spatially-Aware Information Retrieval on the Internet

SPIRIT is funded by EU IST Programme
Contract Number: IST-2001-35047



Practical Similarity Ranking

Deliverable number:	D18 5302
Deliverable type:	R
Contributing WP:	WP 5
Contractual date of delivery:	1 July 2004
Actual date of delivery:	2 August 2004
Authors:	Avi Arampatzis, Marc van Kreveld
Keywords:	Relevance Ranking

Abstract: In this report, we describe a practical relevance ranking procedure, as it is implemented and integrated in the interim prototype of the SPIRIT search engine. We review the theoretical models and ideas presented in the previous three deliverables of WP5, and state the practical decisions and refinements made during implementation. Possible improvements are identified which will lead to an advanced and final version.

Contents

- 1. INTRODUCTION 3
- 2. BASIC CONCEPTS & PROCESSES 3
- 3. FOOTPRINT RELEVANCE 5
- 4. DOCUMENT SPATIAL RELEVANCE 6
- 5. DOCUMENT RELEVANCE --- COMBINING MULTIPLE SCORES 6
 - 5.1. Score Normalization 6
 - 5.2. Score Combination 6
 - Non-distributed method 7
 - Distributed method 7
- 6. IMPLEMENTATION 7
 - 6.1. Input 8
 - 6.2. Output 8
 - 6.3. Log 8
- 7. LIST OF FUTURE IMPROVEMENTS 9
- REFERENCES 9

D18:5302

Practical Similarity Ranking

1. Introduction

In this report, we describe the practical relevance ranking component (Re1Rank V1.4, 20/6/2004), as it is implemented and integrated in the interim prototype (Version 2 beta) of the SPIRIT search engine [1].

We review the theoretical models and ideas presented in the previous three deliverables of WP5 [2,3,4], and state the practical decisions and refinements made during implementation and integration into the interim SPIRIT prototype.

We identify possible improvements and extensions that will lead to an advanced and final version.

2. Basic Concepts & Processes

From a geo-spatial perspective, a query is represented as a single **footprint**, and each document of the web-collection is represented by a bag of footprints. A footprint contains the coordinates of a real-world location. In the context of the SPIRIT project, three levels of detail for footprints are supported:

1. A **single point**, or the **centroid** (also a single point) of a more complex structure, e.g., of a polygon.
2. An axis-aligned minimal **bounding box** of the area of interest, completely defined by two extreme points (the south-west and north-east).
3. A **polygon**, defined by a list of its successive points.

In the interim prototype, representations up to level 2 are supported. Footprints may have more than one representation; in such a case, the highest-level one will be preferred in future versions. For example, if a bounding box is present, it will be preferred over the centroid.

A **spatial relationship** or **connector** relates a non-spatial to a spatial term and defines the geographical area of interest with respect to the query footprint. The connectors supported by the relevance ranking component of the interim prototype are:

1. **inside**,
2. **near**,
3. **north-of, south-of, east-of, and west-of**.

The formulas we use to calculate, for these connectors, **footprint similarity scores** between query and document footprints are given in Section 3. When all footprints in a document are assigned a similarity score with respect to the query footprint and connector, a **document spatial similarity score** for the document can be calculated. In Section 4 we elaborate on how this is done.

Documents also have a **textual similarity score** that is determined by the search-engine using the **BM25** term weighting scheme [5]. BM25 scores are based on the probability¹ of a document being relevant to a query. These scores are calculated using three pieces of information such as term frequency (how many times a query word occurs in the document), inverse document frequency (how many documents contain a query word, in the whole collection), and document length. Thus, the relevance ranking component also deals with combining the spatial and textual document scores into generating a single ranking. There are 5 different options (denoted with bold letters below) for score combination:

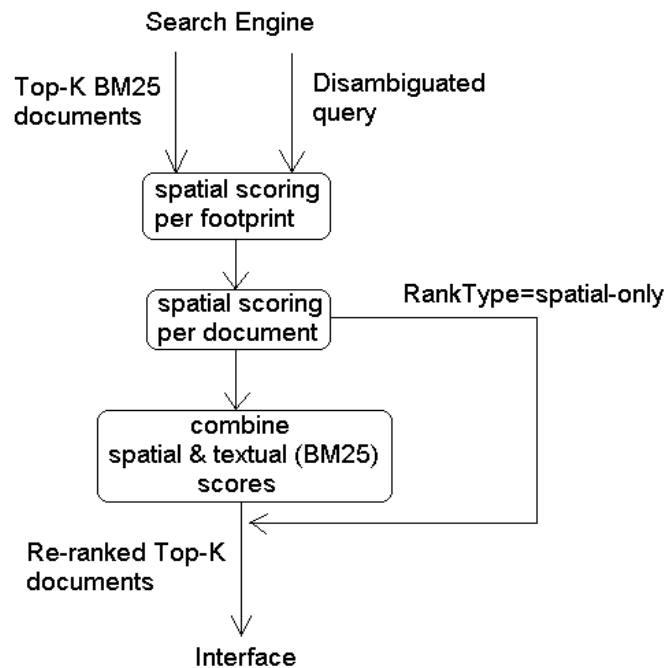
1. **BM25**: all spatial awareness is turned off. Documents are ranked purely textually.
2. **spatial**: the document's spatial similarity score is used as the document score; the BM25 score is not used at all.
3. methods for combining BM25 with spatial scores:
 - a. **non-distributed**
 - b. distributed methods:
 - i. **distance-distributed**
 - ii. **angle-distributed**

In Section 5.2, the methods for combining scores are explained in more detail.

To summarize, in order to produce a relevance ranking of documents with respect to a query, the following steps are taken:

1. For every document footprint, a footprint similarity score is produced with respect to the query footprint and connector.
2. For every document, a document spatial similarity score is produced based on the geographical similarity scores of the footprints contained in the document.
3. Document spatial similarity scores are usually combined with textual BM25 scores into a document similarity score.
4. Documents are ranked in a descending order of their document similarity scores.

The following Figure depicts this process.



¹ Actually, BM25 scores are neither probabilities nor lie into the [0,1] interval; they are merely an order-preserving transformation of the probability of relevance and can be quite unpredictable (in their range) across queries. Thus, BM25 scores are not normalized, they are positive numbers with theoretically no upper limit (practically they are finite), so it is not possible to compare scores between different queries.

3. Footprint Relevance

- **inside.** Binary operator defined between a query's bounding-box and a document's footprint (bounding-box or point). The co-ordinates are checked for containment, which is a trivial operation.
- **near.** $\text{near}(a,b) = \exp(-L * D(a,b))$, where a and b are the centroids of a query's and a document's footprint, $D(a,b)$ is their Euclidian distance. Thus, proximity scores decay exponentially from 1 to 0 (asymptotically) with increasing distance. L controls the rate of decay, or, in real-world terms, "how far is far". Note that this formula is different than the initial one reported in [2]. The differences are that we now have only one parameter (L) not two (thus it is simpler), but the new formula cannot behave linearly (which we do not think it is important).
- **north-of, south-of, east-of, west-of.** Assuming A and B are the centroids of document footprint and query footprint respectively, the general formula for direction-only is implemented in C++ as²:

```
inline double direction_of(const Point &A, const Point &B, const float phi)
{
    if (A==B) { return 1; }
    double psi=angle_to(A,B);
    if (phi != 0) {
        if ((psi>=phi-90)&&(psi<=phi+90))
            { return ( 1 - fabs(phi-psi)/90.0 ); }
        else { return 0; }
    }
    else {
        if ((psi>90)&&(psi<270)) { return 0; }
        else {
            if (psi<=90) { return ( 1 - fabs(phi-psi)/90.0 ); }
            else {
                psi = 360-psi;
                return ( 1 - fabs(phi-psi)/90.0 );
            }
        }
    }
}
```

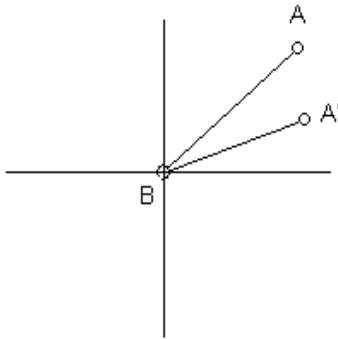
where ϕ is the angle of interest, for example, $\phi=90$ for north, 180 for west, etc. $\text{angle_to}(A, B)$ returns the angle of the vector BA from the positive x-axis, the origin is assumed on point B:

```
inline double angle_to(const Point &A, const Point &B)
{
    if (A==B) { return 0; }
    double phi = atan((A.y-B.y)/(A.x-B.x))*180.0/PI ;
    if (A.x<B.x) { phi = 180+phi; }
    else if (A.y<B.y) { phi = 360.0+phi; }
    return phi;
}
```

So, the direction-of routine calculates the deviation of this angle from the angle of interest. We also take into account proximity, so the final formulas for the connectors include nearness. For example: $\text{north-of}(a,b) = \text{direction-of}(a,b,90) * \text{near}(a,b)$.

Note that the formula for direction is reported with errors in [2] which we corrected here.

² We prefer here to give the C++ code directly, because the mathematical formula would not be much more readable due to its many legs (sub-cases for different values of ϕ).



For example, points A and A' will give non-zero scores only for north-of and east-of, since they lie above the x-axis (thus north of B) and right of the y-axis (thus east of B). Moreover, discarding their distance from B for now, A is more north than A' since it makes smaller angle with the y+ axis. Similarly, A' is more east than A since it makes smaller angle with the x+ axis.

Now, taking into account their distance from B through multiplying with the near(.) function, the final scores for the north-of and east-of operators can be calculated. However, they can only be given if the L of the near(.) function is known, and this depends on what the pairs (A,B) and (A',B), i.e. the combination of the search concept and location.

4. Document Spatial Relevance

For the interim prototype we followed the “best-match” approach, i.e. a document’s spatial similarity score is the highest footprint similarity score of the footprints it contains. Initial experiments with a few other approaches that use partial score contributions from all document’s footprints did not seem promising; they rather seem to reduce effectiveness returning dubious results for some queries. Also, trying to use the internal (in the document) and external frequencies (across the collection) of footprints for weighing them in a *tf*idf* fashion³ does not seem to work out.

5. Document Relevance — Combining Multiple Scores

5.1. Score Normalization

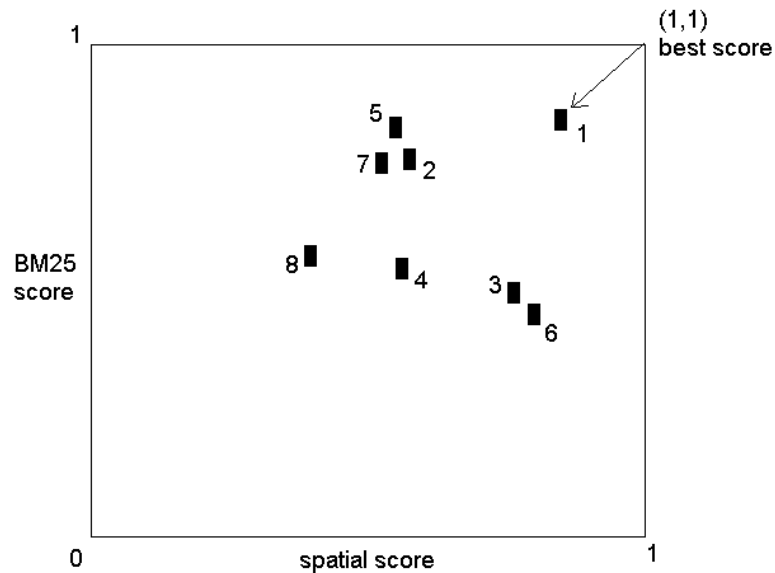
Spatial scores are normalized into [0,1] by definition (Section 3). BM25 scores can be quite unpredictable in their range and they are currently normalized linearly by dividing with the highest document score for the query.

We are aware that the way the individual scores are normalized may have an impact to the model for calculating the final score. This impact has to be determined experimentally. A better score normalization should take into account the underlying score distribution. It is shown in [7] that a scoring function consisting of a linear combination of (independent) term weights produces a Normal score density for the relevant documents in the limit of a high number of terms. Moreover, the non-relevant score density can be approximated by a decreasing exponential; thus the total score distribution can be approximated by an exponential-Gaussian mixture. Whether the same holds for spatial scores is currently unknown and it will be investigated when experimental data become available.

5.2. Score Combination

The following Figure depicts 8 documents having a spatial and a BM25 score. The problem is how to combine those two scores into one.

³ This refers to a family of term weighting schemes widely used in Text Retrieval and can be found in any Information Retrieval textbook.



Non-distributed method

The non-distributed method ranks the documents in an ascending order of their Euclidian distance from point (1,1) that is the best score.

Distributed method

The distributed method try to de-cluster documents with almost the same score components. The Figure above corresponds to a ranking from a distributed method. Note that although 4 is further from (1,1) it is ranked above 5.

We have two variations of the distributed method, one based on the angle of a yet un-ranked point to the already ranked, and one based on its distance from the already ranked. We will not expand on the details of the algorithms here; these are already reported in [3,4,6].

6. Implementation

The relevance ranking component is seen as a part of the search engine, thus it does not communicate directly with other components of the SPIRIT project. All input/output communication is performed with the search engine (GLASS) [5]. Currently (in the interim prototype), the relevance ranking routines are not integrated with the GLASS code.

The software is written in ANSI C++ and is at the moment still stand-alone. The Standard Template Library (STL) is used to store and manipulate the data. It has successfully been compiled with different versions of the g++ compiler under the Linux and Solaris operating systems.

Special attention was given to efficiency, so the component runs fast enough for demonstration purposes. The footprint similarity functions are simple and computed in linear time to the length of the input ranking. Concerning the score combination methods, the non-distributed one is also linear. The distributed ones are quadratic, but this doesn't seem to be a problem considering that the speed bottleneck is in other parts of the SPIRIT system. To give an indicative running time, using the computationally most expensive setup (e.g. angle- or distance-distributed) the relevance

ranking component takes only 70msecs (excluding input/output) to re-rank the top-100 BM25 documents on a AMD K6 500MHz processor with 128Mb RAM. The usage of RAM is linear to the length of the input ranking, and no intermediate files are needed.

All data are communicated using file I/O in the form of unix pipes. For example, it can be called as:

```
shell_prompt> ReIRank <inputfile >outputfile
```

ReIRank writes all logging information to the standard error stream. Logging information can be captured and appended to a log-file as:

```
shell_prompt> ReIRank <inputfile >outputfile 2>>logfile
```

The formats of the input-, output-, and log-files are shown below and they differ from the initial ones documented in [3]. We will not explain the formats since the details are too technical, and they are going to change in the next version of the component: we are going to use a SOAP interface.

6.1. Input

```
n 1
f UK3 1 -1147.47568 248.310909 2 -1221.865858 179.0732686 -1092.089695 327.117034 0 0 0 0
c inside
t non-distributed
d 100
r SPRT-003-020-111-0055173 1193 156 UNPRCSD 1 85
f 110851 1 -1206.809000 234.805900 0 0 19956 70 1
[another 84 footprint lines follow for this document]
[another 99 documents follow]
```

6.2. Output

```
d 100
r SPRT-004-002-136-0067773 1901 138 UNPRCSD 23 1 1 74
f UK353 1 -1146.5 188.639 2 -1151.78 179.073 -1141.15 192.863 0 19956 1944 1 1
f 157708 1 -1121.49 195.773 0 0 19956 230 1 0.783293
f UK2746 1 -1114.78 188.032 2 -1116.75 185.854 -1112.76 189.872 0 19956 187 1 0.742294
[another 71 footprint lines follow for this document]
r SPRT-011-045-052-0025556 7855 139 UNPRCSD 3 0.866093 2 6
[ 6 footprint lines follow for this document]
[another 98 documents follow]
```

6.3. Log

```
ReIRank: ReIRank 1.4, 20/06/2004, avgerino@cs.uu.nl
ReIRank: called on: Tue Jul 27 18:01:57 2004
ReIRank: Loading ranking from standard input...
ReIRank: Done in 20 msecs.
ReIRank: connector=inside, rankType=non-distributed, docc=100
ReIRank: Re-ranking...
ReIRank: Done in 10 msecs.
ReIRank: Sending ranking to standard output...
ReIRank: Done in 10 msecs.
```

Note that, if ReIRank is called using unix pipes, the running times reported for loading from standard input or sending to standard output include the running times of other programs that generate the input or read the output. For example, in a call like

```
shell_prompt> program_1 | ReIRank | program_2
```

the input time reported will include the total running time of `program_1`, since all programs start running at once. The output time will include the time that `program_2` spends in order to read the output.

7. List of Future Improvements

- Add a variable that controls the highest-level of detail for footprints to be used in calculations. If the requested level does not exist for a footprint, use the highest level present.
- Upgrade the component to allow for polygon footprints.
- Add more connectors...e.g., in-between.
- Abandon the file Input/Output interface. SOAP interface will be used.
- Determine rate of decay L for near-scores based on concepts (e.g. hotel, airport) and query footprints (e.g. London, Wales).
- Better individual score normalization before combining them into a single one (see Section 2). The individual textual and spatial scores have an impact on the distance of a document from the query, as well, as the angle. BM25 scores are currently normalized linearly (by dividing with the highest score for the query), and the spatial scores are normalized exponentially. Empirical data will show whether our current normalization is effective or needs fine-tuning.

References

[1] *Interim Prototype*, SPIRIT D16:1301, April 2004.

[2] *Simple and Useful Similarity Measures*, Avi Arampatzis and Marc van Kreveld, SPIRIT Report D9:5102, September 2003.

[3] *Simple Similarity Ranking Procedure*, Avi Arampatzis, Marc van Kreveld, and Iris Reinbacher, SPIRIT Report D14:5201, January 2004.

[4] *Abstract Multi-attribute Similarity Ranking*, Avi Arampatzis, Marc van Kreveld, and Iris Reinbacher, SPIRIT Report D17:5301, April 2004.

[5] *A Working Searching System*, H. Joho, and Mark Sanderson, SPIRIT Report D10:2101, January 2004.

[6] *Distributed Ranking Methods for Geographic Information Retrieval*, Avi Arampatzis, Marc van Kreveld, Iris Reinbacher, and Roelof van Zwol, Geoinformatica, 2004.

[7] *The Score-distributional Threshold Optimization for Adaptive Binary Classification Tasks*, A. Arampatzis and A. van Hameren, Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, September 2001.