# Multi-Objective Resource Selection in Distributed Information Retrieval

**Shengli Wu and Fabio Crestani**[*]
Department of Computer and Information Sciences
University of Strathclyde
Glasgow G1 1XH, Scotland, UK
Email: <shengli, fabioc>@cs.strath.ac.uk

## Abstract

In a Distributed Information Retrieval system, a user submits a query to a broker, which determines how to yield a given number of documents from all possible resource servers. In this paper, we propose a multi-objective model for this resource selection task. In this model, four aspects are considered simultaneously in the choice of the resource: document's relevance to the given query, time, monetary cost, and similarity between resources. An optimized solution is achieved by comparing the performances of all possible candidates. Some variations of the basic model are also given, which improve the basic model's efficiency.

**Keywords:** information retrieval, resource selection, distributed digital libraries.

## 1 Resource Selection in Distributed Information Retrieval

With the rapid growth of information available via Internet, a huge number of resources are available to users. However, the user often finds it difficult to select the most appropriate resource, given an information need. Distributed Information Retrieval (DIR) can help in this task by automating the *resource selection process* and making it transparent to the user [8].

In the Information Retrieval and Database research communities, much of the attention has been devoted to select those resources which could provide the largest number of relevant documents. Some heuristic methods have been proposed, for example, in [2, 5, 11, 9, 1, 6].

More recently, Fuhr introduced cost as a factor to be considered for resource selection [3], where the cost includes such factors as access time, monetary cost of the service, expected retrieval quality, expected number of relevant documents in the resource, query processing cost and document delivery cost. However, the resource selection model is too abstract and does not help much in the practical decision of which resource to select, given these costs. In addition, Fuhr's model mixes many type of costs into one objective–cost, so lacking the flexibility of meeting various kinds of user requirements. For example, a user A could be concerned mostly about relevance, and not be concerned with monetary cost of accessing the documents or the time needed to retrieve them. While user B might not care much about the number of relevant documents he will retrieve and be mostly concerned with getting a few relevant documents quickly and cheaply. In such situations, a distinctive policy needs to be applied to each user. In this paper we will consider independently relevance, time, and monetary cost as the main resource selection factors.

[*]Author to whom correspondence should be sent.

Another important factor in resource selection in DIR is the chance of getting *document duplicates*. The issue of dealing with documents duplication in different resources has been approached by several researchers, for example in [7, 4]. It is obvious, that if many duplicates are present in the fused retrieval set, the user will waste money and time and even if the service was free, it would still be annoying for the user to be presented with several identical documents. To prevent this from happening, two things could be done at two different stages by DIR systems. Firstly , at resource selection stage, a distinctive policy can be used to select those resources which are more likely to provide document duplicates for a given query. Secondly, at data fusion stage, duplication (totally or partially) can be checked out and redundant documents can be eliminated. While for duplicate detection, processing documents at data fusion stage is an indispensable complex process, since documents could not be identical but similar, sophisticated resource selection policy could alleviate the problem and help improving the effectiveness and efficiency of the whole system. Therefore, a two-stage process solution is a sensible way. In this paper we will not deal with document processing for duplicate detection, but we will introduce in the resource selection model a factor that will take into account the chance of getting document duplicates.

We argue that multi-objective optimization could provide an appropriate solution to deal with the above problems. In the following of this paper, we will present a multi-objective resource selection model for DIR. The model considers four aspects of resources: relevance of the documents to the given query, time, monetary cost, and chance of getting document duplicates. The rest of the paper is organized as follows. Section 2 presents the framework and the basic resource selection model. Section 3 provides a general solution to the above model. Section 4 briefly discusses extensions to the basic model. Section 5 concludes the paper.

## 2 A Basic Resource Selection Model

Assumption 1. In every local resource, the documents retrieved for any query are ranked in a descending order of estimated relevance.

We think this is an acceptable assumption since it is quite commonly met by most of today's Web search engines, Digital Library systems, and Information Retrieval systems.

Assumption 2. For any query, the user always specify a number $n$, instructing to the DIR system to retrieve the top $n$ estimated relevant documents.

Suppose we have $m$ resources for consideration, $D$ denotes the set of resources we have, and $D(i)$ $(1 \leq i \leq m)$ denotes the $i$-th resource among them. In the following, all the parameters are related to the result of a given query $Q$. $Doc(i)$ denotes the result set we get from $D(i)$ with respect to $Q$. $N(i)$ denotes the number of documents in $D(i)$. $Doc(i,j)$ denotes the $j$-th document in $Doc(i)$, and $R(i,j)$ denotes the relevance estimate for $Doc(i,j)$. According to Assumption 1, all retrieved documents appear in descending order of estimated relevance. Therefore, for any $1 \leq j \leq N(i)-1$, $R(i,j) \geq R(i,j+1)$. $T_{all}(i,j)$ denotes the time spent to retrieve the first $j$ documents in $Doc(i)$, while $C(i,j)$ denotes the charge or monetary cost to be payed for $Doc(i,j)$.

Assumption 3. For every local resource $D(i)$, and every document $D(i,j)$ returned from it for a given query, the estimated relevance $R(i,j)$ has been normalized so that $0 \leq R(i,j) \leq 1$ always holds.

It is also useful to consider the following measures:

$$Avg\_T(i,j) = \frac{T_{all}(i,j)}{j}$$

$$Avg\_C(i,j) = \frac{1}{j}\sum_{k=1}^{j} C(i,k)$$

$$Avg\_R(i,j) = \frac{1}{j}\sum_{k=1}^{j} R(i,k)$$

which denote the average time, the average charge, and the average relevance of the first $j$ $(1 \leq j \leq N)$ documents in $Doc(i)$, respectively.

Among them, $Avg\_T(i,j)$ and $Avg\_C(i,j)$ need further normalization. Since both are usually measured in the same or comparable units, a multi-resource-wide (global) normalization is possible. For $Avg\_T(i,j)$, we define for any $1 \leq i \leq m$ and $1 \leq j \leq N(i)$:

$$T(i,j) = T_{all}(i,j) - T_{all}(i,j-1)$$

$T(i,j)$ is the time difference between the first $j-1$ and $j$ documents obtained from $D(i)$, so, for example, $T(i,1)$ is the time needed for retrieving document $D(i,1)$ from $D(i)$. We can also define:

$$T_{max} = \max_{(1 \leq i \leq m, 1 \leq j \leq N)} T(i,j)$$

and

$$N\_Avg\_T(i,j) = \frac{T_{all}(i,j)}{j * T_{max}}$$

For C(i,j), we first define

$$C_{max} = \max_{(1 \leq i \leq m, 1 \leq j \leq N(i))} C(i,j)$$

which yield:

$$N\_Avg\_C(i,j) = Avg\_C(i,j)/C_{max}$$

In a similar way, $Avg\_T(i,j)$ and $Avg\_(i,j)$ are normalized into $N\_Avg\_T(i,j)$ and $N\_Avg\_C(i,j)$ respectively, yielding values in the range $[0,1]$.

Assumption 4. We have a measure of similarity between each pair of resources $D(i)$ and $D(j)$. The similarity measure is normalized in the range $[0,1]$ and can be represented through a similarity matrix $S$, such as $S(i,j)$ denotes the similarity between resource $D(i)$ and $D(j)$.

The following two properties always hold true for $S$: a) for any $1 \leq i \leq m$, $S(i,i) = 0$, and b) for any $1 \leq i \leq m$, $1 \leq j \leq m$, $S(i,j) = S(j,i)$.

The aim of the similarity measure is to provide an estimate of the possibility of having duplicate documents retrieved from two different resources. We will not address the derivation of this measure of similarity in this paper. Suffices to say that this could be done via the analysis of the results of sample queries and can be kept up to date by monitoring the results of users queries.

Suppose $(X = \{x_1, x_2, ..., x_m\})$ is an array, here each $x_i$ $(1 \leq i \leq m)$ is an integer representing the number of documents obtained from $D(i)$. We define a multi-resource similarity measure in the following way. Firstly, We define a function

$$f(i) = \begin{cases} 0 & if \;\; x_i = 0 \\ 1 & if \;\; x_i \geq 1 \end{cases}$$

for $1 \leq i \leq m$, which indicates which resource has documents occurring in the result. Then we define:

$$S_{res-all} = \sum_{i=1, i<j}^{m-1} \sum_{j=2}^{m} f(i)f(j)S(i,j) \qquad (1)$$

which sums up the similarity measure of every pairs of different resources whose documents occur in the result.

For a given query $Q$ and a given number $n$, we aim at finding a resource selection policy which could consider all the four above criteria (i.e. document's relevance to the query, time, charge, and chance of the document being a duplicate) at the same time. The problem can be qualified as a *multi-objective optimization problem.*

Suppose $(X = \{x_1, x_2, ..., x_m\})$ is a potential solution to this problem. An optimized solution to our problem should maximize:

$$R_{res} = \frac{1}{n} \sum_{i=1}^{m} Avg\_R(i, x_i) * x_i$$

and minimize at the same time:

$$T_{res} = \frac{1}{n} \sum_{i=1}^{m} N\_Avg\_T(i, x_i) * x_i$$

$$C_{res} = \frac{1}{n}\sum_{i=1}^{m} N\_Avg\_C(i, x_i) * x_i$$

$$S_{res} = 2/m'(m'-1)S_{res-all} \qquad (2)$$

Here $m'$ denotes the number of resources which contributes some documents to the result, so the total number of different resource pairs is $m'(m'-1)/2$.

We use the *Utility Function Method* to deal with this multi-objective optimization problem [10]. Firstly, a utility function is defined for each of the above objectives depending on their importance; then a total utility function can be defined. In our case, we just adopt a linear function by defining a coefficient for each of the objectives, but some more complex functions could be used. Therefore, we could define a total utility function as follow:

$$U = k_1 R_{res} - k_2 T_{res} - k_3 C_{res} - k_4 S_{res} \qquad (3)$$

where $k_1$, $k_2$, $k_3$, and $k_4$ are coefficients whose values range in $[0, 1]$ and $k_1 + k_2 + k_3 + k_4 = 1$.

We now have to to maximize U.

## 3 Solution to the Basic Resource Selection Model

To simplify the discussion, we suppose that for each resource $D(i)$, the size of $Doc(i)$ is always $n$. This could be achieved in the following way: if the size of the retrieved document set for $D(i)$ is greater than $n$, we can keep the first $n$ documents and discard the rest; if the size of document set is less than $n$, we reach $n$ with some documents with a very low $R$ (i.e. relevance estimate value close to 0), and with very high $C$ and $T$ values (close to 1). In short, we should guarantee that such dummies will not be selected later as real documents. However, it should be noted that it is often the case that number of estimated relevant documents from each resource is often greater than $n$.

One way to obtain the overall optimum solution is by enumerating all possible candidates and decide which one is the best. Notice that $n$ documents in $m$ resources can be mapped into a m-digit number with carry $n + 1$. The problem can be mapped into finding out all such numbers whose digits in all places sum to $n$. The following rules are always true for the numbers which satisfy our requirement:

- $n$ is the smallest;
- $n\underbrace{0.........0}_{m-1}$ is the largest;
- if $T$ is a satisfied number whose digit in the units is not 0, then beyond $T$, $T + n$ is the smallest one which satisfies our requirement;
- if $T = n' \underbrace{0.........0}_{l\&1\leq l<m-2}$ is a satisfied number, then T + $(n - n' + 1)\underbrace{0..........0}_{l-1}(n' - 1)$ is the next one beyond T.

Based on above, we can use the algorithm reported in Figure 1 to compute the optimum solution. For each solution, such algorithm calls Procedure Cal_utility to calculate its utility, compares them, and keeps the best as the final solution.

Let us explain how the Procedure CalUtility works. Suppose we know the values of $D$, $m$, $n$, $k_1$, $k_2$, $k_3$, $k_4$, $R(i, j)$, $C(i, j)$, $T_{all}(i, j)$, and $S(i, j)$ for $(1 \leq i \leq m,\ 1 \leq j \leq n)$. One simple way, when we get a solution $(X = \{x_1, x_2, ..., x_m\})$, is to calculate its utility directly by Equation 3. Firstly, we can calculate $N\_Avg\_T(i, j)$, $N\_Avg\_C(i, j)$, and $Avg\_R(i, j)$ for any $i$ and $j$, that takes $O(mn)$ time. Then we get $R_{res}$, $T_{res}$, and $C_{res}$, that takes $O(m)$ time. For similarity, we first calculate out $S_{res-all}$ (Equation 1), and finally $S_{res}$ (Equation 2), which should take $O(mn)$ time in all. Finally, we sum them together.

The above process has room for improvement. The utility in Equation 3 for a solution can be divided into four parts. The first three parts of the utility have common properties, that

```
01. Algorithm 1: Calculating the Optimum solution
02. Input: m, n; //m databases and n documents needed
03. Output: best_utility, x(1..m); // For the optimum solution
04. //Variable used: p points to the lowest non-zero place
05. //Variable used: A(0..m) for keeping a n-digit number
06. best_utility = -∞; //Lines 6-8 for initialization
07. for i:= 0 to m-1 do A(i) := 0;
08. A(m) := n;
09. while (A(0) ≠ 1) do
10. { utility := CalUtility(A);
11.     if (utility > best_utility)
12.     { best_utility := utility;
13.         for j := 1 to m do X(j) := A(j);
14.     }
15.     //Lines 16-23 for setting up next solution
16.     if (A(m) ≠ 0) //the digit in the units is not 0
17.     { A(m) := A(m)-1; A(m-1)++;
18.         if A(m)=0 p := m-1;
19.     }
20.     else // the digit in the units is 0
21.     { A(m) := A(p)-1; A(p) := 0; A(p-1)++;
22.         if A(m)=0 then p–;
23.     }
24. }
25. Procedure CalUtility(A)
26. //Calculate the utility of a given solution
```

Figure 1: Algorithm for the Basic Model

is, for any given resource, the utility of a solution from one part is only determined by the number $j$ of documents which is involved in the result. For example, for relevance it is $j * Avg\_R(i,j)$, if the first $j$ documents in resource $D(i)$ are involved in the solution. But for resource similarity, the situation is different, since that part of utility function is decided by all the resources which have some documents appearing in the result set.

If we pre-calculate the value for every element of $U'(i,j)$:

$$U'(i,j) = j*(k_1 Avg\_R(i,j) - k_2 N\_Avg\_T(i,j) - \\ -k_3 N\_Avg\_C(i,j)) \qquad (4)$$

then, for every solution, we can get its partial utility value by just one scan of each $x_i$ value of every resource $D(i)$, i.e. $\sum_{i=1}^{m} U'(i,x_i)/n$. However, there is no simple way for getting $S_{cre}$, which has to be handled as described above. In such a way, CalUtility needs $O(m^2)$ time for every execution.

Theorem 1. For m resources and n documents, the number of all possible solutions are: $S(m,n) = \frac{m(m+1)....(m+n-1)}{n!}$. The proof of this theorem cannot be reported in this paper for space limitations.

According to Stirling's formula, $n! \approx (n/e)^n \sqrt{2\pi n}$. Therefore, $S(m,n) = \frac{(m+n-1)!}{n!(m-1)!} \approx \frac{((m+n-1)/e)^{m+n-1}\sqrt{2\pi(m+n-1)}}{(n/e)^n\sqrt{2\pi n}((m-1)/e)^{m-1}\sqrt{2\pi(m-1)}} \approx \frac{(m+n)^{m+n}}{n^n m^m}$. When $m = n$, the above function gets its maximum, so the worst time complexity of $S(m,n)$ is $O(2^{m+n})$.

## 4 Some Variations and Related Solutions

The basic model in Section 2 could precisely reflect the multi-objective resource selection problem, but the optimum solution given in Section 3 is not very efficient. The reason for that is due to resource similarity. We need to have complete information about the similarity between resources. In this section we will remove some of these requirements to improve efficiency. These variation to the basic model are still sound and reflect more appropriately the real resource selection problem.

### 4.1 Static Variation Model

One possible way is to define an average similarity measure for every resource available:

$$Avg\_S(i) = \frac{1}{m-1}\sum_{j=1}^{m} S(i,j)$$

for any $D(i)$. Then based on that, we define $S_{res}$ as follow:

$$S_{res} = \frac{1}{m'}\sum_{i=1}^{m} f(i) Avg\_S(i) \qquad (5)$$

Here both $m'$ and $f(i)$ have the same meaning as in Equation 1 and 2, however, comparing the above equation with Equation 1 and 2, we see that in Equation 5 each addend is just decided by one resource rather than by a pair of them. In this variation of the basic model, all other three parts in the utility function are the same. Since the utility of each resource can be calculated independently from other resources we call this model *Static Variation Model*.

Notice that since we make our decision only considering $Avg\_S(i)$ for any given resource, and not the exact values, such a model is less accurate than the basic model. However, we can benefit from this simplified model as Algorithm 1 could be implemented more efficiently. We can evaluate the matrix $U(i,j)$ using the following equation:

$$U(i,j) = (k_1 Avg\_R(i,j) - k_2 N\_Avg\_T(i,j) -$$
$$- k_3 N\_Avg\_C(i,j) - k_4 Avg\_S(i)) * j \qquad (6)$$

In such a way, CalUtility(A) just needs $O(m)$, and not $O(m^2)$ as before.

Actually, we can benefit even more. In such a situation, the "divide-and-conquer" algorithm in Figure 2 could be used. The dividing step iterates over the number of resources, while the merging step is performed in Procedure BestUtility. Since the utility for documents in each resource is not related to any other resource which contributes to the result, we can use the maximum computed for $i$ resources in computing the maximum for $i+1$ resources. The time complexity of the algorithm is $O(mn^2)$.

```
01.   Algorithm 2: Computing the Optimum Solution
              for Static Variation Model
02.   Input: m, n, U(1..m, 0..n); //U(i,0) is always 0
03.   Output: best_utility, X(0..m); // For the optimum solution
04.   for i:= 1 to n do X(i) := U(1,i);
05.   for i:= 1 to m do
06.   { BestUtility(X, U(i), M);
07.     for j:= 0 to m do x(i):= B(i);
08.   }
09.
10.   Procedure BestUtility(X',U',M')
11.   for k:=0 to n do
12.   { M'(k) := X'(k);
13.     for j:=0 to k-1 do
14.         if $(M'(k) < (X(j) + U'(k-j)))$ {M'(k):=X'(j)+U'(k-j); }
15.   }
```

Figure 2: Algorithm for the Static Variation Model

### 4.2 Utility Function Analysis

Before further discussion, let us analyze the Utility function under Static Variation Model. For any given $i$, $U(i,j)$ only varies with $j$. We can rewrite Equation 6 as follow:

$$U_{db}(j) = k_1 Avg\_R_{db}(j) - k_2 N\_Avg\_T_{db}(j) -$$
$$- k_3 N\_Avg\_C_{db}(j) - k_4 Avg\_S_{db} \qquad (7)$$

Let us now discuss each of the four parts in Equation 7 one by one.

- $Avg\_S_{db}$ is invariant wrt $j$.
- The value of $Avg\_R_{db}(j)$ decreases when $j$ increases, according to Assumption 1 in Section 2. Further, we could assume that the relevance measure for documents in

$Doc(i)$ forms an arithmetic progression, that is, each document gets a certain amount less than its previous one when ordered by relevance. Then $Avg\_R_{db}(j)$ ($N\_Avg\_R_{db}(j)$) is a straight line with negative slope.

- Connection time is one important part of all the time needed. It becomes a bigger part if only few documents are retrieved. If we assume that every document has the same size and the data transfer rate per unit time keeps constant, then $Avg\_T_{db}(i, j)$ ($N\_Avg\_T_{db}(i, j)$) is monotonically decreasing with the number of documents $j$.

- As for charge, we simply consider the following possible alternatives: a) a charge per unit of time; b) a charge per unit of data; c) a charge per document. Again, if we suppose every document has the same size and the data transfer rate per unit of time is constant, then the first two become identical. In such a situation, $Avg\_C_{db}(j)$ is just like $Avg\_T_{db}(j)$ in shape. In the third situation, $Avg\_C_{db}(j)$, just like $Avg\_S_{db}$, keeps constant wrt variations of $j$. We can prove that in such situations the shape of $P_{db}(j)$ can only be decreasing or increasing monotonically, or firstly increasing to a certain point then decreasing. What is more, the corresponding $U_{db}$ values could be different from one to another for different resources, still, their general shapes are always very similar to each other.

In practical situations, things could be more complicated especially for time and charge. We will not try to present any more detailed considerations fitting practical use.

### 4.3 A Greedy Algorithm and the Dynamic Variation Model

For reasons of space we will not be able to present here other two algorithms: the *Greedy algorithm*, which assumes that only one maximum exists in the utility function for each resource, and the *Dynamic Variation Model*, which remove the assumption related to the availability of similarities reported in the Static Variation Model. Both the Greedy algorithm and the Dynamic Variation Model bring the worst complexity of resource selection to $O(mn)$.

The details of these algorithm will be presented in an extended version of this paper.

## 5 Conclusion

In this paper, we propose a multi-objective model for resource selection in distributed information retrieval, in which four aspects are considered: document's relevance to a given query, query time, query expense, and similarity between resources. An optimized solution is achieved by comparing the performances of all possible solutions. In addition, some variations to the basic multi-objective model have been proposed as well, for more efficient implementations.

The following are some related issues demanding further consideration:

- In this paper we assume that relevance of documents to a query is normalized for all resources. However, that is not the usual case in practice. Further effort is needed, especially when each resource use a different indexing and retrieval model.

- A similarity measure between resources has been assumed throughout this paper, but how to define and implement such a measurement is not trivial.

- The algorithms presented in this paper (and some that could not be presented for space limitations) need further study with regards to their efficiency.

We are currently working in these directions.

**Acknowledgements**

This work is supported by the EU Commission under IST Project MIND(IST-2000-26061). More information about MIND can be found at `http://www.mind-project.net/`.

## References

[1] J. Callen, M. Connell, and A. Du. Automatic discovery of language models for text databases. In *Proceedings of ACM SIGMOD International Conference*, Philadelphia, USA, May 1999.

[2] J.K. Callen, Z. Lu, and W. Croft. Searching distributed collections with inference networks. In *Proceedings of the 18th annual International ACM SIGIR Conference*, Seattle, June 1995.

[3] N. Fuhr. A decision-theoretic approach to database selection in networked ir. *ACM Transaction on Information Systems*, 17(3):229–249, 1999.

[4] H. Garccía-Molina and N. Shivakumar. A copy detection mechanism for digital documents. In *Proceedings of 2nd International Conference in Theory and Practice of Digital Libraries*, Austin, USA, June 1995.

[5] L. Gravano and H. García-Molina. Generalizing gloss to vector-space database and broker hierarchies. In *Proceedings of 21st VLDB Conference*, Zűrich, Switzerland, 1995.

[6] D. Hawking and P. Thistlewaite. Methods fro information server selection. *ACM Transaction on Information Systems*, 17(1):40–76, January 1999.

[7] K. Monostori, A. Zaslavsky, and H. Schmit. Document overlap detection system for distributed digital libraries. In *Proceedings of ACM International Conference on Digital Libraries*, pages 226–227, San Antonio, June 2000.

[8] H. Nottelmann and N. Fuhr. MIND: an architecture for multimedia information retrieval in federated digital libraries. In *Proceedings of the DELOS Workshop on Interoperability in Digital Libraries*, Darmstadt, Germany, 2001.

[9] A. Powell, J. French, J. Callen, M. Connell, and C. Viles. The impact of database selection on distributed searching. In *Proceedings of ACM SIGIR Conference*, pages 232–239, Athens, Greece, July 2000.

[10] S.S. Rao. *Optimization Theory and Applications*. Wiley Eastern Limited, 1984.

[11] B. Yuwono and D. Lee. Server ranking for distributed test retrieval systems on the internet. In *Proceedings of the Fifth International Conference on Database Systems for Advanced Application*, Melbourne, Australia, April 1997.