Automatically Combining Ranking Heuristics for HTML Documents

Joaquin Rapela [*]
IBM Almaden Research Center
San Jose, CA, 95120-6001, US
rapela@usc.edu

ABSTRACT

Current search engines use several criteria or heuristics to rank *HTML* documents. *HTML* ranking heuristics need to be combined into a ranking function that given a text query returns a ranked list of *HTML* documents. The standard approach is to build a weighted average by manually estimating the importance of every heuristic and assigning a weight proportional to the estimated importance. In the current paper we apply an automatic method for combining *HTML* ranking heuristics. Using recall/precision evaluations we study the performance of the automatic method and using collections of *HTML* documents with different characteristics we show that the automatic method finds weights tailored to specific characteristics of each document collection.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search process*; I.7.2 [**Document and Text Processing**]: Document Preparation—*Markup languages*

General Terms

Performance, Design, Experimentation

Keywords

HTML, *WWW*, *HTML* ranking, *HTML* ranking heuristics, automatic combination *HTML* ranking heuristics, Information Retrieval

1. INTRODUCTION

An *HTML* ranking heuristic is a function that given a query returns a list of *HTML* documents ranked according to a specific characteristic of these documents. Several heuristics can be used to improve the ranking of *HTML* documents. The link structure can be used to select relevant pages [9, 8], words in `<META>` fields could be treated as content descriptors [3], words in the first lines of *HTML* documents could be considered more important than words in the rest of the document [12].

HTML ranking heuristics need to be combined into a ranking function that given a text query returns a ranked list of *HTML* documents. The standard approach is to build a weighted average, to manually estimate the importance of every heuristic and to assign to each heuristic a weight proportional to the estimated importance.

Manually assigning weights to heuristics is a difficult and time consuming task. Moreover, an intuitively correct choice of weights could lead to a ranking function that performs worse than a non-weighted ranking function.

In the current work we applied the automatic method for combining multiple retrieval systems described in [1] to the problem of automatically combining *HTML* ranking heuristics. The heuristics are combined in a weighted average and an automatic method is used to infer the optimal weights from training data.

Recall/precision evaluations are used to show that manually weighting a multi-heuristic ranking function with intuitively correct weights can lead to a performance degradation respect to a non-weighted (all weights equally set to 1.0) multi-heuristic ranking function. We also use recall/precision evaluations to show that an automatically weighted outperforms a non-weighted multi-heuristic ranking function.

Collections of *HTML* documents with different characteristics are used to show that the algorithm finds weights tailored to specific characteristics of each collection. The algorithm is learning the weights from the document collection.

Section 2 presents related work. Section 3 describes the algorithm used to find the weights for combining the heuristics. Section 4 depicts the results of the experiments and Section 5 shows a problem in the algorithm used to find the weights.

The main contributions of the current paper are the application of the automatic method for combining multiple retrieval systems described in [1] to the problem of automatically combining *HTML* ranking heuristics, the evaluation of the method with web data and the detection of a flaw in the

[*]Currently on leave at University of Southern California

WIDM 2001, Atlanta, GA, USA
© ACM 2001 1-58113-444-4/01/11 ...$12.00

algorithm which motivated its modification, currently being developed.

2. RELATED WORK

In [2] a strategy to automatically combine *HTML* ranking heuristics is described. The *HTML* ranking heuristics are combined in a weighted sum and a genetic algorithm is used to find the weights. The weights are constrained to be integers in the range [1..15].

The current work differs from that in [2] in that we use a numerical optimization strategy to find the real weights. By using real weights instead of integers in the range [1..15] we can explore a wider range of possibilities for combining the heuristics.

3. AUTOMATICALLY COMBINING HTML RANKING HEURISTICS

This section briefly describes the automatic method used to find the weights of multi-heuristic ranking functions. For a detailed description refer to [1].

R_θ is the multi-heuristic ranking function which is to be optimized. $R_\theta(q, d)$ provides the single relevance estimate of document d for query q using weights $\theta = (\theta_1, \theta_2, \dots, \theta_n)$. Large values of $R_\theta(q, d)$ imply that the multi-heuristic ranking function estimates that d is likely relevant to the query q; small values imply that d is less relevant.

The goal is to find values of θ such that R_θ best ranks the documents for a set of queries. The notion of "best ranking" is with respect to a user specified ordering over the documents for each query. R_θ has successfully ranked the documents for query q when the ordering implied by the values of R_θ correspond to the ordering specified by the user.

The user specified ordering is formalized by the binary relation $>_q$ defined by Wong and Yao [13]. For documents d and d' from the collection of documents D: $d >_q d' \leftrightarrow$ the user prefers d to d'.

R_θ has successfully ranked the documents for query q when the ordering implied by the values of R_θ corresponds to the partial ordering $>_q$. The optimization method heuristically searches the space of parameters seeking θ such that for each pair of documents d and d' belonging to the document collection D: $R_\theta(q, d) > R_\theta(q, d')$ whenever $d >_q d'$.

We optimize a criterion which measures how well the ordering implied by R_θ corresponds to the relation $>_q$. The criterion is derived from Guttman's Point Alienation measure [5], a statistical estimate of the rank correlation between two variables. The criterion is defined in Eq. 1 where d, d' are documents from the collection D, q is a query from the collection Q and θ is the set of weights.

$$J(R_\theta) = \frac{-1}{|Q|} \sum_{q \in Q} \frac{\sum_{d >_q d'} R_\theta(q, d) - R_\theta(q, d')}{\sum_{d >_q d'} |R_\theta(q, d) - R_\theta(q, d')|} \qquad (1)$$

The criterion J is an average, over the queries in Q, of the rank match between $>_q$ and R_θ for the documents in D. Note that when R_θ perfectly orders the documents with respect to $>_q$, the numerator and denominator of Eq. 1 are equivalent, since the sum of the differences $R_\theta(q, d) - R_\theta(q, d')$ is equivalent to the sum of the absolute value of differences. In this case, the ratio is 1.0 and the criterion takes on its minimum value -1.0. When R_θ is completely disordered, the numerator is the negative of the denominator, the ratio is -1.0, and the criterion is maximized at 1.0. The goal is to minimize the criterion. Figure 1 provides a small example of the evaluation of criterion.

3.1 Numerical Optimization Method

Since $J(R_\theta)$ is mostly differentiable with respect to θ, any number of standard numerical optimization methods which make use of the gradient can be used [11]. In the current work we used the Polak-Ribiere implementation of the conjugate gradient method.

Though it is mostly differentiable, $J(R_\theta)$ has critical points with zero derivative. One such critical point occurs when R_θ assigns the same ranking value for all $d \in D$. This is an extreme condition and is very infrequent. A more prevalent critical point occurs when $R_\theta(q, d) = R_\theta(q, d')$, as the derivative of $|x|$ at $x = 0$ is required. In this case, we define $\delta|x|/\delta x = -1.0$ for $x = 0$; this has the effect of forcing $R_\theta(q, d)$ to be strictly greater than $R_\theta(q, d')$ when $d >_q d'$.

4. EXPERIMENTS

The servlets[1] based interface to the multi-heuristic ranking system used in the current experiments can be reached at [7].

4.1 Implemented HTML Ranking Heuristics

The implemented heuristics use the indexed format of *HTML* documents. The indexing process creates inverted indexes containing word occurrences in the different fields of the *HTML* documents and creates a connectivity graph representing the hypertext link connectivity between the *HTML* documents (Fig. 2).

A stop list is used to remove common words that have not discriminant power. The stop list contains common words in standard documents (e.g. A, HE or THE) as well as common words in *HTML* documents (HOMEPAGE, WEB or WWW).

A stemming algorithm is used to reduce words to stems [10]. Finally the stems are stored in inverted indexes [4].

4.1.1 Implemented Field Heuristics

Field heuristics rank *HTML* documents according to the similarity of the query and the text in a specific field of *HTML* documents. Different similarity measures have been used to evaluate the similarity between queries and text. In the current experiments we use a normalized inner product with IDF studied in [6].

We implemented a *Title* heuristic that uses the text inside the `<TITLE>` field of *HTML* documents, a *Top Text* heuristic that uses the first 200 words of *HTML* documents, a *Highlighted Text* heuristic that uses the text inside `H1` and `H2`

[1]http://java.sun.com/products/servlet

$$\{d_1, d_2\} >_{q_1} \{d_3\}$$

Query	*Document*	H_1	H_2
q_1	d_1	0.7	0.5
q_1	d_2	0.8	0.6
q_1	d_3	0.9	1.0

$$J(R_\theta) = -\frac{\frac{0.7\theta_1+0.5\theta_2}{\theta_1+\theta_2} - \frac{0.9\theta_1+1.0\theta_2}{\theta_1+\theta_2} + \frac{0.8\theta_1+0.6\theta_2}{\theta_1+\theta_2} - \frac{0.9\theta_1+1.0\theta_2}{\theta_1+\theta_2}}{\left|\frac{0.7\theta_1+0.5\theta_2}{\theta_1+\theta_2} - \frac{0.9\theta_1+1.0\theta_2}{\theta_1+\theta_2}\right| + \left|\frac{0.8\theta_1+0.6\theta_2}{\theta_1+\theta_2} - \frac{0.9\theta_1+1.0\theta_2}{\theta_1+\theta_2}\right|}$$

Figure 1: Example of the evaluation of the J criterion for one query, Q=$\{q_1\}$ and 3 documents, D=$\{d_1, d_2, d_3\}$. The user prefers document d_1 and d_2 to document d_3 for query q_1. The ranking function is configured with two heuristics, H_1 and H_2, whose rankings are shown in the third and fourth columns of the table.

fields and inside hypertext links[2] and an *All Text* heuristic that uses all the text in *HTML* documents.

4.1.2 Implemented Links Heuristics

The links heuristics use a graph, created during the indexing phase, representing the link connectivity between indexed documents (Fig. 2)

To compute the similarity of a document d to a query q the *Out Links* (*In Links*) heuristic gets from the links connectivity graph all the documents pointed by (that point to) document d and returns the average value of the *All Text* heuristic for query q and each of these documents (Fig. 2).

4.1.3 Multi-Heuristic Ranking Function

In the following experiments we evaluated the performance of the multi-heuristic ranking function illustrated in Fig. 3. The evaluated ranking function returns a single relevance estimate of document d for query q using weights $\theta = (\theta_1, \ldots, \theta_7)$. It combines in a weighted average the heuristics described in Section 4.1.1 and Section 4.1.2.

4.2 Experiment Design

The following experiments are designed to evaluate the benefits of using the automatic mechanism for finding weights of multi-heuristic ranking functions.

Experiment 1 (Sec. 4.4) compares the recall/precision performance of a ranking function using intuitively correct, manually selected weights with a ranking function using no weighting (all weights equally set to 1.0). The experiment shows that care should be taken when selecting weights for multi-heuristic ranking functions.

Experiment 2 (Sec. 4.5) compares the recall/precision performance of a multi-heuristic ranking function using automatically selected weights with a multi-heuristic ranking function using no weighting. It presents evidence showing that automatically selected weights can improve the performance of multi-heuristic ranking functions.

[2]In *HTML* hypertext links are encoded in anchor tags of the form $< a ... >$ *text* $< /a >$. The Highlighted Text Heuristic uses the *text* between the $< a ... >$ and $< /a >$ tags

We wanted to find whether the automatic algorithm selects weights tailored to particular characteristics of the training corpus. For example, if documents in a given corpus are densely linked to related documents, then we would expect that the automatic mechanism will assign high weights to link heuristics.

Experiment 3 (Sec. 4.6) compares the weights found for documents from Encyclopaedia Britannica (Sec. 4.3.1) with the weights found for documents in the training set of the corpus Single Root Category from Yahoo (Sec. 4.3.2). Experiment 4 (Sec. 4.7) compares the weights found for documents in the training set of the corpus Single Root Category from Yahoo (Sec. 4.3.2) with the weights found for documents in the corpus Multiple Root Categories from Yahoo (Sec. 4.3.3).

Due to limitations in our software architecture we could not manage large document collection and we had to limit our evaluations to small data sets. These limitations narrow the validity of our results that should only be interpreted as initial evidence.

4.3 Document Collections

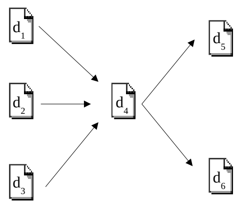
To evaluate the method for automatically finding weights we used three collections of *HTML* documents. One collection contained documents from Encyclopaedia Britannica[3] and two collections contained documents from Yahoo[4].

Documents in these collections have been manually grouped into categories. For each category we constructed text queries and for each query we set the documents of the category as relevant to the query.

The automatic method for finding weights (Sec. 3) uses a document preference relation $>_q$ where $d >_q d'$ whenever document d is preferred to document d' for query q. For the current document collections we established $d >_q d'$ whenever d belongs to the category for which q was constructed and d' belongs to a different category.

4.3.1 Encyclopaedia Britannica

[3]http://www.britannica.com
[4]http://www.yahoo.com



$$
\begin{aligned}
\mathit{InLinks}(q,d_4) &= \frac{\mathit{AllText}(q,d_1)+\mathit{AllText}(q,d_2)+\mathit{AllText}(q,d_3)}{3}\\
\mathit{OutLinks}(q,d_4) &= \frac{\mathit{AllText}(q,d_5)+\mathit{AllText}(q,d_6)}{2}
\end{aligned}
$$

Figure 2: Connectivity graph, *In Links* heuristic and *Out Links* heuristic.

$$
\begin{aligned}
R_\theta(q,d) = {} & \frac{\theta_1\ \mathit{Title}(q,d)+\theta_2\ \mathit{Meta}(q,d)+\theta_3\ \mathit{TopText}(q,d)+\cdots}{\theta_1+\theta_2+\theta_3+\cdots}\\
& \frac{\cdots+\theta_4\ \mathit{HighText}(q,d)+\theta_5\ \mathit{AllText}(q,d)+\theta_6\ \mathit{InLinks}(q,d)+\cdots}{\cdots+\theta_4+\theta_5+\theta_6+\cdots}\\
& \frac{\cdots+\theta_7\ \mathit{OutLinks}(q,d)}{\cdots+\theta_7}
\end{aligned}
$$

Figure 3: Multi-heuristic ranking function used in the current experiments.

The corpus from Encyclopaedia Britannica contains 183 documents grouped into 8 categories.

Every document contains a table of contents with hypertext links to related documents in the same category. Thus, in the corpus from Encyclopaedia Britannica, the link heuristic should receive a large weight in the ranking function.

Documents in Encyclopaedia Britannica contain a large amount of text. They usually start with an introduction to a topic, it follows a development and they conclude with links to related articles. This represents a significant difference respect to standard web documents that usually contain a small amount of text and summarize their content in the initial paragraphs.

4.3.2 Single Root Category from Yahoo

To build the corpus Single Root Category from Yahoo we automatically downloaded 318 documents from 29 sub-categories under `Science` :: `Computer_Science`.

For each sub-category we created a text query by extracting keywords from the sub-category title; the documents of the sub-category and the sub-categories contained within it were set as relevant to the query (e.g. for the category `Science` :: `Computer_Science` :: `Algorithms` we created a query `Algorithms` and we set the documents from the category `Science` :: `Computer_Science` :: `Algorithms` and from `Science` :: `Computer_Science` :: `Algorithms` :: `Genetic_Algorithms` as relevant to the query).

The corpus was further partitioned into a training and an evaluation subset. The training subset contained 141 documents and 12 queries and the evaluation subset contained 177 documents and 15 queries.

4.3.3 Multiple Root Categories from Yahoo

The corpus Multiple Root Categories from Yahoo contains 387 documents grouped into 8 categories. For each category we created a text query by extracting keywords from the category title and setting the documents of the category as relevant to the query.

Documents in the previous corpus belong to the computer science domain, cover similar topics and share a common jargon whereas documents in the current corpus belong to different topic areas, cover different topics and use different vocabulary.

4.4 Experiment 1: Manual Weights

Using the documents in the evaluation subset of the corpus Single Root Category from Yahoo (Sec. 4.3.2) we compared the recall/precision performance of the multi-heuristic ranking function illustrated in Fig. 3 using manually selected weights versus using no weighting (all weights equally set to 1.0).

For the manually selected weights we set the highest weight (5.0) to the Title heuristic because words in the title of *HTML* documents tend to be good content descriptors. Authors of web pages frequently use relevant keywords in the meta field so we assigned the next higher weight (4.0) to the Meta heuristic. We selected a higher weight for the Title heuristic because in *HTML* documents the title field is used more frequently than the meta field and because the meta field sometimes contains general keywords not related to the topic of the document to improve the retrieval of the document by search engines (web spamming).

`H1` or `H2` fields or the text in hypertext links contain highlighted words that are usually representative of the content of the document. We assigned the next weight (3.0) to

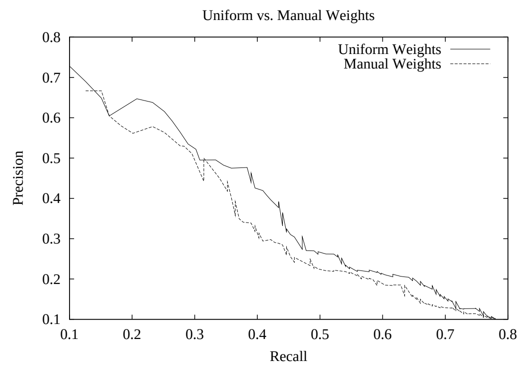


Figure 4: Recall/Precision comparison between a manually weighted and a non-weighted multi-heuristic ranking function.

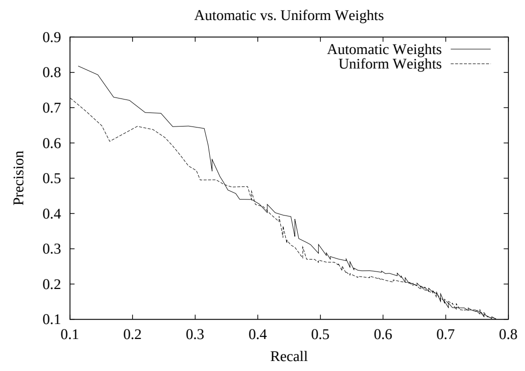


Figure 5: Recall/Precision comparison between an automatically weighted and a non-weighted multi-heuristic ranking function.

the Highlighted Text heuristic. We ranked the Highlighted Text heuristic lower than the previous two heuristics because *HTML* documents frequently contain links not directly related to the topic of the document, advertisement links for example, and keywords extracted from these links will be bad content descriptors.

We set a weight of 1.0 for the other 4 heuristics (All Text, Top Text, In Links, Out Links).

The results of the recall/precision comparison are illustrated in Fig. 4.

Discussion: For almost every recall value the precision of the manually weighted ranking function is worse than that of the non-weighted counterpart. This experiment shows that care must be taken when weighting ranking heuristics because a weighting based on intuition could lead to a performance degradation respect to the use of no weighting.

4.5 Experiment 2: Automatic Weights

We used documents in the training subset of the corpus Single Root Category from Yahoo (Sec. 4.3.2) to automatically find the weights of the multi-heuristic ranking function illustrated in Fig. 3. The selected weights are shown in the third column of Table 1.

Using the documents in the evaluation subset of the corpus Single Root Category from Yahoo (Sec. 4.3.2) we compared the recall/precision performance of the ranking function using automatically found weights with a ranking function using no weighting. Fig. 5 shows the results of the evaluation.

Discussion: For low recall values the automatic weights perform better than the uniform weights. Search engine users usually look only at the top 20 ranked documents. These documents correspond to the low recall section of the recall/precision curve. Therefore the automatic weights improve the precision for documents commonly examined by search engine users.

The documents and queries used to find the weights are different from the documents and queries used in the recall/precision evaluation. Therefore, the results from Fig. 5 indicate that the automatic method is able to select weights that result in performance improvements for queries not considered during the training session. This characteristic of the automatic method to generalize to new queries and documents is what makes the automatically weighted ranking function of greater value than the non-weighted counterpart.

Heuristic	*EB*	*Yahoo*	
		Single	Mult
Title	0.202	0.242	0.376
Meta	0	0.096	0.334
Top Text	0.133	1	0.985
Highlighted Text	0	0	0
All Text	0.255	0.435	1
In Links	0.554	0.024	0
Out Links	1	0.043	0

Table 1: Automatically found weights for different document collections. Column 1 shows the name of the heuristic and columns 2, 3 and 4 show the weights for Encyclopaedia Britannica (Sec. 4.3.1), Yahoo Single Root Category (Sec. 4.3.2) and Yahoo Multiple Root Categories (Sec. 4.3.3) respectively.

4.6 Experiment 3: Encyclopaedia Britannica versus Yahoo

We used the documents from Encyclopaedia Britannica (Sec. 4.3.1) and from the training set of the corpus Single Root Category from Yahoo (Sec. 4.3.2) to find the weights of the multi-heuristic ranking function illustrated in Fig. 3. The weights found for documents from Encyclopaedia Britannica and for documents from Yahoo are shown in the second and third columns of Table 1 respectively.

Discussion: For documents from Encyclopaedia Britannica the links heuristics, Out Links and In Links, received the highest weights, whereas for documents from Yahoo they received low weights. As noted in Sec. 4.3.1, documents from Encyclopaedia Britannica contain a table of contents with hypertext links to related documents in the same category. This means that links heuristics, by considering the content of linked documents, will return valuable information for the ranking process and, thus, they should receive large weights. On the other side, documents from Yahoo are poorly linked to other documents inside Yahoo so the link heuristics, that only considers links to other documents inside the same corpus, will be of less help for the ranking process.

As we mentioned in Section 4.3.1, documents from Encyclopaedia Britannica are structured in the same way as their equivalent in the paper version of the encyclopedia; they usually start with an introduction to a topic, it follows a development and they conclude with links to related articles. This represents an important difference compared to standard web documents that usually summarize their content in the initial paragraphs. This difference is reflected in the weights found by the automatic method. For documents from Yahoo, where text in the initial paragraphs summarize the content of the document, the Top Text heuristic received the highest weighting, whereas for documents from Encyclopaedia Britannica, where the initial paragraphs usually contain introductory material, the Top Text received a low weight.

Documents from Encyclopaedia Britannica do not contain meta fields; thus, the automatic method assigned a zero weight to the Meta heuristic.

The zero weight of the Highlighted Text heuristic in both corpora represents a problem of the automatic method that will be discussed in Section 5.

4.7 Experiment 4: Single versus Multiple Root Categories from Yahoo

We used the documents from the training set of the corpus Single Root Category from Yahoo (Sec. 4.3.2) and from the corpus of Multiple Root Categories from Yahoo (Sec. 4.3.3) to find the weights of the multi-heuristic ranking function illustrated in Fig. 3. The weights found for the corpora Single Root Category and Multiple Root Categories are shown in columns 3 and 4 of table Table 1.

Discussion: When documents cover different topics the meta and title fields of an *HTML* document are better content discriminators than when documents cover related topics. Accordingly, the automatic method assigned a higher weight to the Meta and to the Title heuristics for the corpus Multiple Root Categories from Yahoo than for the corpus Single Root Category from Yahoo .

As noted in Sec. 4.3.3 documents from the corpus Single Root Category from Yahoo share a common jargon whereas documents in the corpus Multiple Root Categories from Yahoo use different vocabulary. Therefore the plain text should be a better content discriminator in the corpus Multiple Root Categories from Yahoo than in the corpus Single Root Category from Yahoo. This feature has been captured by the automatic method that assigned a higher weight to the *All Text* heuristic in the corpus Multiple Root Categories from Yahoo than in the corpus Single Root Category from Yahoo.

Yahoo does not include two documents from the same site in a single category; thus, categories from Yahoo will contain documents from different sites. Considering that the majority of the hypertext links are internal links, links between pages of the same site, it is reasonable to expect that documents in a single category of Yahoo will be poorly linked to other documents in the same category. Thus, the majority of the links from a document in Yahoo to another document in Yahoo will occur between documents in different categories.

Categories in the corpus Multiple Root Categories from Yahoo cover different topics. Thus, it is unlikely that a document in one category will link to a document in another category. On the other side, categories in the corpus Single Root Category from Yahoo cover related topics and it is probable that a document from one category will link to a document in another category. Therefore, documents in the corpus Single Root Category from Yahoo contain more links to other documents in the same corpus than in the corpus Multiple Root Categories from Yahoo. This feature is reflected in columns 3 and 4 of Table 1 where the weights assigned to the links heuristics are greater for the corpus Single Root Category from Yahoo than for the corpus Multiple Root Categories from Yahoo.

Again, the zero weight of the Highlighted Text heuristic in both corpora represents a problem of the automatic method that will be discussed in Section 5.

5. PROBLEMS WITH THE ALGORITHM FOR FINDING WEIGHTS

The algorithm for finding weights of multi-heuristic ranking functions (Sec. 3) optimizes the ability of the ranking function to match the document preference relation $>_q$. This is a precision concept. The algorithm is optimizing the precision of the ranking function respect to the document preference relation. Recall is another important characteristic of ranking functions that is not being considered by the optimization algorithm.

To appreciate the consequences of optimizing precision without considering recall consider a collection of 100 documents and 1 query where 50 documents are relevant to the query. Assume two ranking heuristics, H1 and H2. H1 returns 1 relevant document in response to the query and H2 returns 51 documents where documents ranked 1 through 49 are relevant to the query, document ranked 50 is irrelevant and document ranked 51 is relevant. At a maximum coordination level H1 achieves maximum precision 1.0 and low recall 1/50 and H2 achieves almost perfect precision 50/51 and maximum recall 1.0. Therefore, H2 that optimizes both precision and recall performs better than H1 that optimizes precision while achieving low recall.

H1 by ranking relevant documents higher than non-relevant ones perfectly matches the document preference relation $>_q$ while H2 that ranks a non relevant document before a relevant one does not perfectly match $>_q$. Thus, the ability of the ranking function to match the document preference relation will be optimized when H2 is not used in the ranking function. The optimization algorithm by only considering the precision of the ranking function is discarding the better heuristic.

This problem can be solved by combining in the optimization criterion a measure of the precision of the ranking function with a measure of its recall. We are currently developing an optimization criterion with these characteristics.

In the described experiments (Sec. 4) the Highlighted Text heuristic received a zero weight because it was the less precise of all the heuristics and, thus, the automatic method minimized its weight.

6. SUMMARY

An automatic method for combining retrieval systems [1] was applied to the problem of combining ranking heuristics into multi-heuristic ranking functions for *HTML* documents. The method was used to find the weights for multi-heuristic ranking functions structured as a weighted average of ranking heuristics. Experiments using Web data were developed to evaluate the performance of the automatic method.

We manually constructed an intuitively correct set of weights for a multi-heuristic ranking function. We compared the recall/precision performance of an intuitively weighted versus a non-weighted multi-heuristic ranking function. The intuitively weighted function performed worse than the non-weighted counterpart showing that care must be taken when combining *HTML* ranking heuristics.

Using recall/precision evaluations we showed that an automatically weighted multi-heuristic ranking function performs better than a non-weighted multi-heuristic ranking function.

Collections of *HTML* documents with different characteristics were used to find the weights for the multi-heuristic ranking function. We showed that for each document collection the selected weights were tailored to particular characteristics of each collection.

A problem in the algorithm used to find the weights was detected and a solution is currently being developed.

7. REFERENCES

[1] B.T. Bartell, G.W. Cottrell, and R.W. Belew. Automatic combination of multiple ranked retrieval systems. In *Proceedings of the Special Interest Group on Information Retrieval*, Dublin, Ireland, 1994.

[2] M. Cutler, H. Deng, S.S. Maniccam, and W. Meng. A new study on using html structures to improve retrieval. In *Proceedings of the Eleventh IEEE Conference on Tools with Artificial Intelligence*, pages 406–409, 1999.

[3] A discussion of search-engine web-page ranking schemes. Available at the Search Engine Watch web site http://searchenginewatch.com/rank.htm.

[4] W.B. Frakes and R. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms.* Prentice-Hall, 1992.

[5] L. Guttman. What is not what in statistics. In *The Statistician*, volume 26, pages 81–107, 1978.

[6] D. Harman. An experimental study of factors important in document ranking. In *Proceedings of the ACM SIGIR*, pages 186–193, Pisa, Italy, 1986.

[7] Multi heuristic system used in the described experiments. `http://www-scf.usc.edu/~rapela/projects/rankui`.

[8] J.M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677, San Francisco, California, January 1998.

[9] L. Page. The pagerank citation ranking: bringing order to the web. In *Proceedings of ASIS'98, Annual Meeting of the American Society for Information Science*, 1998.

[10] M.F. Porter. An algorithm for suffix stripping. In *Program*, volume 14 of *3*, pages 130–137, July 1980. Available as http://open.muscat.com/developer/docs/porterstem.html.

[11] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1991.

[12] E. Pringle, A. Lloyd, and L.D. Dowe. What is a tall poppy among web pages? In *Proceedings 7th International World Wide Web Conference*, pages 369–377, Brisbane, Australia, April 1998.

[13] S.K.M. Wong and Y.Y. Yao. Query formulation in linear retrieval models. In *Journal of the American Society for Information Retrieval*, volume 41 of *5*, pages 334–341, 1990.