Iterative Procedural Internet Search

A Trusov\(^1,2\) and A Gashkov\(^2\)

\(^1\) Perm Centre of Scientific and Technical Information, 9 Popova st. 614099 Perm, Russia
\(^2\) Microprocessor Automation Means Department, Perm National Research Polytechnic University, 7 Professora Pozdeeva st., 614013 Perm, Russia
gashkov@dom.raid.ru

Abstract. The present study investigates the problem of sorting search result by relevance. The new algorithm based on iterative procedural Internet search is developed. The aim of the study was to improve the existing algorithm. Results show that applying the improved algorithm approximately doubles Normalized Discounted Cumulative Gain at k and therefore greatly decrease average time to find relevant documents in search results list.

1. Introduction
Recent advances in information and communication technologies have provided the possibility of finding any knowledge accumulated by humanity. The global computer network hosts and stores giant unstructured information resources that can be described by a variety of information elements - records, to which free and effective access to a wide range of users should be provided remotely. However, the potential of users to access a large part of human knowledge accumulated on the Internet conflicts with the need to navigate them when solving fairly narrow information retrieval tasks. The rich experience of overcoming such contradictions has been accumulated in the library system. However, the authors of many documents [1, 2] on the Internet do not use this experience, often because of the lack of special knowledge. Attempts to introduce web2, or semantic Internet, are also difficult, since they largely rely on the goodwill of the authors and sometimes contradict commercial interests (accurate description of the content of the page can lead to a decrease in traffic to the site and, consequently, to a fall in revenues).

This research aims to develop the existing algorithm [3] to improve user experience by transferring number of tasks to computing system. As a result, this should create an algorithm which is able to increase the relevance of search results on first page(s).

2. Method
2.1. Existing method
Complex systems study is complicated by curse of dimensionality. Usual ways to break the curse are dimension reduction and brute force – as power of processing units is increasing, more problems can be solved with it. Both ways require clear understanding of complex system. One of system in question is search in Internet.
If the behaviour of the system is a consequence of the step-by-step execution of some elementary functions, then the key to formalizing the description of system behaviour is formalization of the elementary functions description and composition rules. Being defined on the entire state space, the functions become capable of composition, since each new state after the execution of the next function can serve as a starting point for the subsequent one. In this case, the algorithm of implementing functional correspondences (transformations) is of fundamental importance.

Based on the conducted research of approaches to searching for information in the Internet, a procedural model [3] of the information support system for the innovation development processes of the region with unstructured information resources of distributed information systems of the Internet has been developed (Figure 1).

Figure 1. Iterative Internet search algorithm.

A procedural model based on the functional representation of information retrieval in the Internet ensures the formation of an informational pattern of the desired (exact) solution of the stated search task, allows you to explore the subject area in accordance with the search image of the document, on the one hand to ensure the formation of an extended thesaurus of the subject area due to synonymy,
and on the other hand, determine the boundaries of the search (narrowing the subject area) using the homonymy.

The developed model of information search in the Internet allows formalizing the search by the user and significantly improve the quality of search results and/or reduce the time spent.

Information search in the Internet is iterative task, just as search itself so learning it too [4]. Therefore, the quality of information search $u(n)$ can be described by monotonically increasing function of the number of iterations $n$. To search for the necessary information in a given period of time $t$, the average number of iterations is determined by the formula:

$$
\hat{n} = \frac{\bar{t}}{t_f} = \frac{t_s}{|l| \bar{t}_{sn}},
$$

(1)

where $\bar{t}_i$ is average time to solve one search problem, $t_s$ is total Internet search time, $|l|$ is average number of search steps and $\bar{t}_{sn}$ is average step time.

Time of one step is counted as:

$$
t_{sn} = t_u + t_m,
$$

(2)

where $t_u$ is user time and $t_m$ is machine time. It is obvious that the reduction of any of the two components leads to an increase in the average number of iterations for the same time, and, consequently, to an improvement in the quality of the search. Both components are equivalent in terms of their contribution to the solution of search problems in distributed information systems. However, as a rule, $t_u \gg t_m$, which makes the task of decreasing $t_u$ more important for reducing the time for one search step. Assuming that the algorithm described above allows us to form search skills that are close to optimal, one of the ways to reduce time we see the transfer of some work from the analytical part to the machine part, tending to approximate equality of $t_u$ and $t_m$. Ideally, the function should be optimized:

$$
C_u t_u + C_m t_m \rightarrow min,
$$

(3)

where $C_u$ is the cost of operator time and $C_m$ is the cost of machine time.

2.2. New algorithm

In the algorithm presented on Figure 1, we found the following functions that can be executed by processing unit instead of a user:

1. Creating synonymous series.
2. Extracting descriptors (in query and document).
3. Forming information pattern of document.
4. Building search prescription system.
5. Analysing documents returned by search engines:
   - syntactic analysis
   - semantic analysis.

The first three tasks are well studied and performed by all search engines in one way or another. Building search prescription is mostly a user's task, as it is restricted by autocomplete in common search engines. Although some progress in this area was shown with using of synonymous.

The last task is least studied and cannot be used nowadays by commercial search engines just because of computing power lack. Also frequent users queries can be processed without such analysis. But in science-intensive industries queries usually include high specific terms and that analysis is required.
We study the use of syntactic analysis for search results ranking. Here we assume that the user is high experienced and can effective build search prescription using information retrieval query language.

We assume that search query is term of at least two words described with dependency grammar. The main steps of algorithm is:

1. Determining query term.
2. Writing down term in parenthesis representation.
3. Searching Internet for documents containing all query words.
4. Calculating weight for each document.
5. Sorting documents by weight (the lesser the better).

Steps from 1 to 3 are carried by user, step 4 – by any common search engine and steps 5 and 6 are topic of our research.

The algorithms for document weight calculating is presented on Figure 2.

\begin{algorithm}
\textbf{algorithm} page-weight is:
\begin{algorithmic}
\State \textbf{input}: page \( P \), query \( Q \) (\( n \) words with list of links)
\State \textbf{output}: page weight \( w \)
\State \text{set} weight \( w = +\text{infinity} \)
\State \text{using} NLTK Snowball stemmer split \( P \) into list of sentences \( L \)
\For {each sentence \( S \) in \( P \)}
\State \( L := \) pairs list of linked words of \( Q \) in \( S \)
\State \( k := \) cardinality of \( L \)
\If {\( k < 2 \)} \text{then skip} \( S \)
\State \( l_{\text{min}} := 0 \)
\For {each pair \( I \) in \( L \)}
\State \( l_{\text{min}} := l_{\text{min}} + \) minimal total distance for \( I \) in \( S \)
\EndFor
\State \( r := 1-(1/l_{\text{min}})+(n-k) \)
\If {\( r < w \)} \text{then} \( w := r \)
\EndIf
\EndFor
\end{algorithmic}
\end{algorithm}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{algorithm.png}
\caption{Sentence weight calculation algorithm pseudocode.}
\end{figure}

3. Evaluating and results

Evaluation of search results ordering was made with Normalized Discounted Cumulative Gain for first \( k \) documents – NDCG@k [5]. The NDDG@k metric in contrast with recall/precision takes in consideration positions of relevant documents. The \( k \) parameter cut off documents in rest of search results list after \( k \), as results on pages after first few are rarely to never watched.

\[ NDCG@k = \frac{DCG@k}{IDCG@k}, \]

where \( DCG@k \) is Discounted Cumulative Gain, IDCG@k is Ideal Discounted Cumulative Gain, when first \( k \) documents are sorted by relevancy.

\[ DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)}, \]

where \( rel_i \) is the graded relevance of the result at position \( i \).

We evaluated DCG@20 for some requests. Relevancy of each returned document was set by an experts. Search results and reference sort order were obtained through Google custom search API. NDCG@20 for both Google and our sort function is shown in the Table 1.
Table 1. NDCG@20 for two sorting type.

| Sort         | NDCG@20 |
|--------------|---------|
| Google       | 0.428   |
| Our approach | 0.824   |

Table 1 demonstrates that using sentence boundaries detection and dependency grammar for results sorting can improve the metric about two times and can lead to greatly reduced time while finding relevant documents.

4. Conclusion

The study indicates that it is possible to create a new method of sorting search results with use of syntactic information. The next problem to solve is low quality of sentence boundaries detection. NLTK Snowball stemmer produces lots of hypotheses sentences glued up from real ones (up to tens of sentences). This sometimes leads to false high value of NDCG@k and, as result, irrelevant documents pushed to top of list. Further research questions are related to the possibility of automated build of dependency tree of query and thus transferring even more functions from user to computer.

References

[1] Vojskunskij V 1992 About formation of search characteristics Scientific and technical information 2 no 9 pp 6–9
[2] Kharitonov V, Trusov A, Greceev A and Belyakov A 1999 The principle of aggregation in the procedure for developing simulators of man-machine systems TProc. PVI RV 2 pp 7–10
[3] Trusov A and Kharitonov V 2005 Theoretical Foundations Of Information Retrieval Automation In Distributed Information Systems Of The Global Computer Network (Perm: Perm CNTI) p 277
[4] Kharitonov V and Trusov A 2000 The set-theoretic model of the simulator of distributed information systems Cross-university collection of papers: Information control systems (Perm: PSTU Press.) pp 41–49
[5] Järvelin K and Kekäläinen J 2002 Cumulated gain-based evaluation of IR techniques ACM Transactions on Information Systems 20(4) pp 422–46