SEARCH ENGINE OPTIMIZATION USING QUERY PARSING

Kailash Kumar[1], Vinod Bhadu[2]
[1]Research Scholar
[1]Suresh Gyan Vihar University, Jaipur
[1]Email: maheshwari_kailash2002@rediffmail.com
[2]Technology Lead, Infosys Limited, Chandigarh

Abstract: The Internet has become a vast information source in recent years. Searching the desired information or document via internet is one of the most important issues. Every search engine has its own database that defines its own set of documents which are searched by the search engines. No single search engine is capable of searching all kinds of data via Internet. In the modern practices, an interface is created to call multiple search engines in order to satisfy the users query. Calling all search engines for each users request is not feasible as it increases the cost because the network traffic is increased by sending the query to different search engines, some of which may be useless. The problem can be solved by parsing the user's query. The research paper suggests an accurate and fast search mechanism using query parsing techniques.

Keywords: Meta search engine, parsing, information retrieval, search engines, network traffic.

I. INTRODUCTION

Nowadays, internet has become an important source of information. To find the desired data on internet, many search engines have been created. Every search engine has its own database that defines the set of documents that can be searched by the search engine. Generally, the index is created in the database for each documents and it is stored in the search engine which is used to identify the document in the database. Since, the index is already there in the database hence, it becomes very difficult for the search engine to answer the user's query efficiently.

There are two types of search engines which exist in the market, namely, General purpose search engines and Special purpose search engines. General purpose search engines provide searching capabilities for all kinds of documents on the internet while Special purpose search engines focus on documents which are confined to specific domain. Google, Alta vista and HotBot are example of General purpose search engines whereas there are millions of Special purpose search engines which currently available on the internet.

The number of web pages is increasing at very high rate on the internet. Therefore, it is very monotonous to find all kind of data in a single search engine due to several reasons. First, the processing power and storage capabilities may not scale to the rapidly increasing and virtually infinite amount of data. Second, collecting all kind of data on the internet and maintaining it rationally up to date is not easy task if not possible. It becomes a time consuming process for the crawlers which are used by the search engines to collect the data automatically.

An alternative approach is to use the multilevel search engines over the internet. At the lower level, local search engines are used which are grouped at higher level based on the relatedness of their database which in turn, grouped together to form next higher level and so on. At the top level, we have only one search engine called Meta search engine. Whenever, the Meta search engine receives the request from the user in the form of query, it passes the request to appropriate (Meta) search engine in depth first search order. This approach has its own advantages. First, the response time of the query processing is substantially reduced because user queries are evaluated against smaller database in parallel. Second, the index of the local search engine is modified only when the documents in its database are updated, i.e. updating of index is localized. Third, the local information can be collected more conveniently, easily and timely. Last, but not the least, the memory space and processing power of each local search engine can be managed easily. The schematic diagram for the above mentioned scheme is shown below in fig 1.
When a single search engine calls many Meta search engines, the there may be serious problem of inefficiency. For example, for a given query, only small fraction of all search engines may contain useful documents. As a result, if every search engine is called for each query, then there may be substantial loss of network bandwidth and network traffic may be created. Moreover, local resources of each search engine will be wasted when useless databases are searched. The most appropriate solution to this

problem is to first identify those search engines that are most likely to provide useful results to a given user query and then pass it to the appropriate search engine for desired documents. But the question is how to identify potentially useful search engines. The solution to the above problem is to rank all underlying databases in decreasing order of their usefulness for each query using some metadata that describes the contents of each database. Generally, the ranking is based on some parameters which ordinary users may not be able to utilize to fit their needs. The current approach can describe the user, to some degree of accuracy, which search engine is likely to be the most useful, the second most useful, etc for a given user query. Although, such ranking scheme will be useful but it can not say anything about the usefulness of any particular search engine to the user.

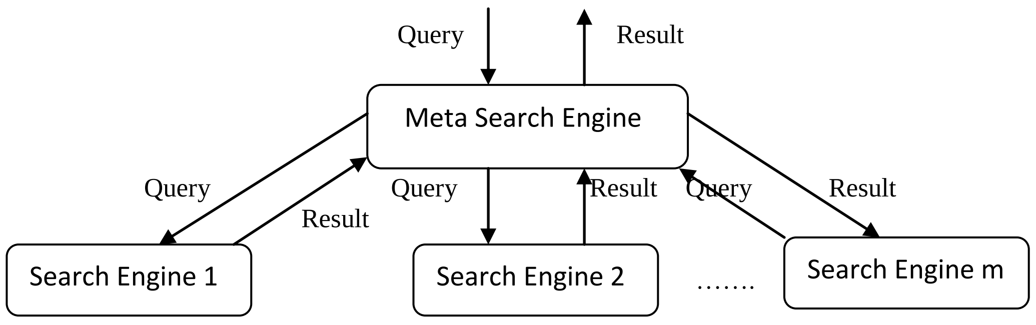

Fig 1: A Typical Meta Search Engine

The usefulness of any search engine for a given user query is measured in terms of two parameters, first, the number of documents (NoDoc) in the database of the search engine that is more likely to be useful to the query, that is, the similarities between the query and the documents as measured by a certain global similarity function are higher than a specified threshold, and second, the average similarity (AvgSim) of these potentially useful documents. It is important to keep in mind that the global similarity function may vary with the local similarity function used by a local search engine. These two parameters together describe the usefulness of any search engine for a given user query. Mathematically, these terms can be defined as follows:

$$NoDoc(T,q,D) = cardinality(\{d | d \in D \text{ and } sim(q,d) > T\})$$

And

$$AvgSim(T,q,D) = \frac{\Sigma d \in D \wedge sim(q,d) > T^{sim(q,d)})}{NoDoc(T,q,D)}$$

Where, *T*: a threshold value
D: database of the search engine
$Sim(q,D)$: Similarity between user's query q and document d in the database D.

It is very important for the users to determine which search engine to use and how many numbers of documents to be retrieved from the each search engine. For instance, if the user can forecast that a highly ranked search engine with a large database has very few useful documents and searching such a large database is not cost effective, then the user may not use that search engine. The cost of using such a search engine can be reduced by limiting the number of documents to be returned to the number of useful documents in the search engine.

In this paper, a new measure which is easy to understand and informative is proposed to characterise the usefulness of a search engine with respect to the users query. Next, a subrange based estimation method is proposed to recognize search engines to use for a given query and to estimate the usefulness of a search engine for the query.

II. RELATED WORK

In order to discover useful search engines to a query, some attributes about the database of each search engine must be stored in the Meta search engine. Such information is known as the representative of a search engine. Based on the representatives used, different methods can be developed for identifying useful search engines.

Several Meta search engines are in working using a range of methods to recognize potentially useful search engines [3], [5], [7], [8], [9] and [10]. However, the database representatives used in most Meta search engines cannot be used to estimate the number of globally most similar documents in each search engine [1], [7], [8] and [10]. In addition, the precautions that are used by these Meta search engines to rank the search engines are not easy to understand. As a result, separate methods have to be used to convert these measures to the number of documents to retrieve from each search engine.

Another shortcoming of these measures is that they are independent of the similarity threshold. As a result, a search engine will always be ranked the same regardless of how many documents are desired, if the databases of these search engines are fixed. This is in conflict with the following situation. For a given query, a search engine may contain many moderately similar documents but very few or zero highly similar documents. In this case, a good measure should rank the search engine high if a large number of moderately similar documents are desired and rank the search engine low if only highly similar documents are desired.

A probabilistic model for distributed information retrieval is proposed in [2]. The method is more suitable in an environment where documents previously retrieved have been identified to be either relevant or irrelevant.

A database of m distinct terms is represented in gGlOSS [5] by *m* pairs (*fi:Wi*), where *fi* is the number of documents in the database that contain the ith term and Wi is the sum of weights of the ith term over all documents in the database, $i = 1, 2, 3 \ldots\ldots m$. The usefulness of a search engine with respect to a given query is defined to be the sum of all documents similarities with the query that are greater than a threshold

A method is proposed in [8] to estimate the number of useful documents in a database for the binary and independent case. In this, each document d is represented as a binary vector such that a 0 or 1 at the ith position indicates the absence or presence of the ith term in document d, and the occurrences of terms in different documents are assumed to be independent. A significant amount of information will be lost when documents are represented by binary vectors. As a consequence of which, these methods are rarely used in practice. The estimation method in [6] assumes term weights to be non-binary.

III. USEFULNESS ESTIMATION

In this section, the basic method for estimating the usefulness of a search engine is described which allows the values of the term weight to be any non-negative real numbers. The basic assumptions used in this method are: the distributions of the occurrences of the terms in the documents are independent and all documents having a term have the same weight for the term for a given database in the search engine. This basic method can very accurately estimate the usefulness of the search engine. Next, subrange-based statistical method is described which can eliminate the second assumption.

IV. BASIC METHOD

Consider a database D of a search engine with m distinct terms. Each document d in the database can be represented as a vector $d = (d1, d2 \ldots\ldots\ldots dm)$, where di is the weight of the ith term ti in representing the document, $1 \le i \le m$. Let us consider the query $q = (u1, u2, \ldots\ldots., um)$, where ui is the weight of the ti in the query, $1 \le i \le m$. If the term does not appear in the query, then its corresponding weight will be zero. The similarity between query q and document d can be defined as the dot product of their respective vectors, i.e. $sim(q\,, d) = u1 * d1 + \ldots. + um * dm$.
In this method, the database D is represented as m pairs $\{(pi, wi)\}$, where $1 \le i \le m$ and pi is the probability that term ti appears in a document in D and wi is the average of the weights of ti in the set of documents containing ti. For a given query $q = (u1, u2, \ldots\ldots., um)$, the database representative is used to estimate the usefulness of database D.

Consider the following generating function:

$$(p1 * X^{w1*u1} + (1 - p1)) * (p2 * X^{w2*u2} + (1 - p2)) * \ldots..* (pr * X^{wr*ur} + (1 - pr))$$

Where, X is a dummy variable.

Let us consider the query q and database D. If the terms are independent and the weight of term ti whenever present in a document is wi, which is given the database representative $(1 \le i \le r)$, then the coefficient of X^s in the above function is the probability that a document in D has similarity s with q.

V. SUBRANGE BASED ESTIMATION METHOD

In the basic method, it was assumed that all documents having a term have the same weight for the term. While in realistic, it is not so. This sort of problem is overcome in the subrange based statistical method. The different documents having a term may have different weight for the term.
Let us consider a term t, ω the average and σ be the standard deviation of the weights of term t in the set

of documents containing the term t. Let p be the probability that the term t appears in a document in the database.

Let t be the term in a specific query then the following equation is included in the probability generating function:

$$p * X^{u*w} + (1 - p)$$

Where, u is the weight of the term in the user query. The above equation assumes that the term t has uniform weight of w for all documents containing the term while in reality the term weights may have non-uniform distribution among the documents having the term. Let the weights of the terms be $w1, w2, \ldots, wk$,
Where, $w1 > w2 > \ldots\ldots > wk$
k: the number of documents having the term and k = p * n and
n: the total number of documents in the database.

Now, suppose that we partition the weight range of t into four subranges, each containing 25 percent of the term weights, as follows. The first subrange contains the weights from $w1$ to ws, where $s = 25\% * k$; the second subrange contains the weights from w_{s1} to w_t, where $t = 50\% * k$; the third subrange contains the weights from w_{t+1} to w_v, where, $v = 75\% * k$ and the last subrange contains weights from w_{v+1} to w_k. In the first subrange, the median is the $\left(25\% * \frac{k}{2}\right) th$ weight of the term weights in the subrange and is w_{m1}, where, $m1 = 12{:}5\% * k$; similarly, the median weights in the second, the third, and the fourth subranges have median weights $wm2$, $wm3$ and $wm4$, respectively, where, $m2 = 37{:}5\% * k, m3 = 62{:}5\% * k$, and $m4 = 87{:}5\% * k$.

Then, the distribution of the term weights of t may be approximated by the following distribution: The term has a uniform weight of w_{m1} for the first 25 percent of the k documents having the term, another uniform weight of w_{m2} for the next 25 percent of the k documents, another uniform weight of w_{m3} for the next 25 percent of documents and another uniform weight of w_{m4} for the last 25 percent of documents.

With the above weight approximation, for a query containing term t, polynomial (4) in the generating function can be replaced by the following polynomial:

$$p1 * X^{u*w_{m1}} + p2 * X^{u*w_{m2}} + p3 * X^{u*w_{m3}} + p4 * X^{u*w_{m4}} + (1 - p)$$

Where, pj is the probability that term t occurs in a document and has a weight of w_{mj} , for $j = 1,2,3,4$. Since, 25% of those documents having term t are assumed to have a weight of w_{mj} for term t, for each j, $pj = p/4$. Essentially, polynomial (5) is obtained from polynomial (4)by decomposing the probability p that a document has the term into four probabilities, $p1, p2, p3, and\ p4$, corresponding to the four subranges.

It is important to note that the subrange-based method needs to know the standard deviation of the weights for each term. As a result, a database with m terms is now represented as m triplets$\{(pi, wi, \sigma i)\}$, for $i = 1, \ldots. m$, where pi is the probability that term ti appears in a document in the database, wi is the average weight of term ti in all documents containing the term and σi is the standard deviation of the weights of ti in all documents containing ti. Furthermore, if the maximum normalized weight of each term is used by the highest subrange, then the database representative will contain m quadruplets$\{(pi, wi, \sigma i, mwi)\}$, with m_{wi} being the maximum normalized weight for term ti. The experimental results indicate that the maximum normalized weight is a critical parameter that can drastically improve the estimation accuracy of search engine usefulness.

VI. ISSUES ON APPLICABILITY

If the representative of a database used by an estimation method has a large size relative to that of the database, then estimation method will have a poor scalability, as this method is difficult to scale to thousands of text databases. Suppose each term occupies four bytes and each number (probability, average weight, standard deviation, and maximum normalized weight) also occupies 4 bytes. Consider a database with m distinct terms. For the subrange-based estimation method, m probabilities, m average weights, m standard derivations, and m maximum normalized weights are stored in the database representative, resulting in a total storage overhead of $20 * m$ bytes.
If the number of search engines is very large, the representatives can be clustered to form a hierarchy of representatives. Each query is first compared against the highest level representatives. Only representatives whose ancestor representatives have been estimated to have a large number of very similar documents will be examined further. As a result, most database representatives will not be compared against the query.

To obtain the accurate representative of a database, we need to know the following information: first, the number of documents in the database, second, the document frequency of each term in the database (i.e., the number of documents in the database that contain the term) and third, the weight of each term in each document in the database. First two things are needed to compute the probabilities and third one is needed to compute average weights, maximum normalized weights, and standard deviations. First two things can be easily obtained. For example, when a query containing a single term is submitted to a search engine, the number of hits returned is the document frequency of the term.

In an environment where internet is used, it may not be feasible to anticipate a search engine to provide the weight of each term in each document in the search engine. Following techniques can be used to obtain the average term weight, their standard deviations and maximum normalized term weights.

1. Sampling technique can be used to estimate the average weight and the standard deviation for each term. When a query is submitted to a search engine, a set S of documents will be returned as a result of the search. For each term t in S and each document d in S, the term frequency of t in d (i.e., the number of times t appears in d) can be computed. As a result, the weight of term t in document d can be computed. If the weights of t in a reasonably large number of documents can be computed then an approximate average weight and an approximate standard deviation for term t can be obtained. Since, the returned documents for each query may contain many different terms, the above estimation can be carried out for many terms at the same time.
2. Find out the maximum normalized weight for each term t directly as follows with respect to the global similarity function used in Meta search engine: submit term t as a single term query to the local search engine which retrieves documents according to a local similarity function.

VII. CONCLUSIONS

The paper introduces the usefulness measure of the search engine which is intuitive and easily understood by the users. A statistical method is presented to estimate the usefulness of a given search engine with respect to each query. Accurate estimation of the usefulness measure allows a Meta search engine to send queries to only the appropriate local search engines to be processes which in turn will save both the communication cost and the local processing cost substantially. Estimation method has the following properties:

1. The estimation makes use of the number of documents desired by the user (or the threshold of retrieval), unlike some other estimation methods which rank search engines without using the above information.

2. It guarantees that those search engines containing the most similar documents are correctly identified when the submitted queries are single-term queries. Internet users submit a high percentage of such short queries and they can all be sent to the correct search engines to be processed.

REFERNCES

[1] J. Callan, Z. Lu, and W.B. Croft, “Searching Distributed Collections with Inference Networks,” ACM SIGIR Conf.

[2] C. Baumgarten, “A Probabilistic Model for Distributed Information Retrieval,” Proc. ACM SIGIR Conf.

[3] L. Gravano and H. Garcia-Molina, “Generalizing GlOSS to Vector- Space Databases and Broker Hierarchies,” Proc. Int’l Conf. Very Large Data Bases.

[4] W. Meng, K. Liu, C. Yu, X. Wang, Y. Chang, and N. Rishe, “Determining Text Databases to Search on the Internet,” Proc. Int’l Conf. Very Large Data Bases.

[5] Howe and D. Dreilinger, “SavvySearch: A Meta-Search Engine that Learns Which Search Engines to Query,” AI Magazine, vol. 18, no. 2.

[6] Yu, W. Luk, and M. Siu, “On the Estimation of the Number of Desired Records with Respect to a Given Query,” Proc. ACM Trans. Database Systems.

[7] Kahle and A. Medlar, “An Information System for Corporate Users: Wide Area information Servers,” Technical Report TMC199, Thinking Machine Corp.

[8] M. Koster, “ALIWEB: Archie-Like Indexing in the Web,” Computer Networks and ISDN Systems, vol. 27, no. 2, pp. 175-182.

[9] U. Manber and P. Bigot, “The Search Broker,” Proc. USENIX Symp. Internet Technologies and Systems (NSITS ’97), pp. 231-239.

[10] B. Yuwono and D. Lee, “Server Ranking for Distributed Text Resource Systems on the Internet,” Proc. Fifth Int’l Conf. Database Systems for Advanced Applications (DASFAA ’97), pp. 391-400.