WEBIST 2006

SECOND INTERNATIONAL CONFERENCE ON WEB INFORMATION SYSTEMS AND TECHNOLOGIES

Proceedings

Internet Technology
Web Interfaces and Applications

SETÚBAL, PORTUGAL · APRIL 11-13, 2006

ORGANIZED BY

INSTICC

SPONSORED BY

INSTITUTO POLITÉCNICO DE SETÚBAL

IN COOPERATION WITH

OMG
OBJECT MANAGEMENT GROUP

INSTICC PRESS

AN ALGORITHM TO USE FEEDBACK ON VIEWED DOCUMENTS TO IMPROVE WEB QUERY

Enabling Naïve Searchers to Search the Web Smartly

Sunanda Patro, Vishv Malhotra, David Johnson
School of Computing, Pvt Box 100, University of Tasmania, Hobart TAS 7001 Australia
spatro@postoffice.utas.edu.au, vishv.malhotra@utas.edu.au, dgjohnso@utas.edu.au

Keywords: Web searching, Information need, Boolean query, Relevance feedback.

Abstract: This paper presents an algorithm to improve a web search query based on the feedback on the viewed documents. A user who is searching for information on the Web marks the retrieved (viewed) documents as relevant or irrelevant to further expose the information needs expressed in the original query. A new web search query matching this improved understanding of the user's information needs is synthesized from these text documents. The methodology provides a way for creating web search query that matches the user's information need even when the user may have difficulty in doing so directly due to lack of experience in the query design or lack of familiarity of the search domain. A user survey has shown that the algorithmically formed query has recall coverage and precision characteristics better than those achieved by the experienced human web searchers.

1 INTRODUCTION

Quality of the links returned in a web search depends on how well the query embodies the user's information needs. An erudite user is able to state a web search query using appropriate terms and jargon to obtain links to the valuable resources. A user searching for information in a new or unfamiliar domain faces difficulties. The difficulties are caused by the searcher's inability to provide suitable terms and synonyms, or not being able to combine them suitably to express the information needs accurately. An unsatisfactory query may return an overwhelming majority of links to resources of no interest to the searcher or may fail to identify useful resources. The former problem is called a *precision* problem and the latter a *recall* problem.

Information foraging (Aula, Jhaveri and Kaki, 2005) is an unusual but apt description of the common web search behavior. It suggests that a typical searcher aims to *maximize the amount of valuable information they gain in a unit time*. This attitude manifests in many well-known observations; for example, few queries have more than three terms; use of operators in the web queries is rare; and only a small number of links appearing at the top of links returned by a search engine are viewed by the searchers (Jansen, Spink, Bateman and Saracevic, 1998), (Hölscher and Strube, 2000). Indeed, the search strategy used by the web searchers simply mimics well-known Artificial Intelligence heuristic called *hill-climbing* (Kopec and Marsland, 1997). In turn, like the heuristic, a search may end in a sub-optimal local maximum. The searcher misses to retrieve the best documents matching the information needs. Some of these searches fail to even retrieve a satisfactory document, leaving the searchers frustrated and believing that appropriate resources do not exist on the Web for their needs.

As a user views documents on the Web, useful examples of documents that user considers somewhat relevant as well as those that are irrelevant are generated. Few browsers make use of this information to improve the quality of the search. Previously, Cohen et al. (1996) and Malhotra et al. (2005) have described algorithms to generate Web search queries from example documents. These algorithms have relied on established techniques in text-categorization to construct queries to select relevant examples and reject irrelevant examples. The generated Boolean expressions have good recall and precision characteristics but the query can be too large to be effectively processed by a search engine. Suggested approach of breaking a single query into a series of queries is not effective; few searchers bother to access more than a few documents let

alone try a sequence of queries. Thus, the onus is on the query synthesis algorithm to provide a single query that meets the user's information needs well when a searcher asks for help. This paper gives such an algorithm to generate a query. Only an integrated query provides access to resources that meet all aspects of the user's information needs as opposed to some aspects of the needs.

Oyama et al. (2004) have suggested a different approach to improve the quality of the web search experience. They identify a domain specific *keyword spice* to augment query terms so that only the resources from the relevant domain are targeted by the search. They illustrate their technique by searching for beef cooking recipes. While a single word query *beef* returns few links to the useful resources, the keyword spiced query *beef & ((ingredients & !season & !description) | tablespoon)* has good success in meeting the user's information needs. The main limitation of the approach is that one needs to develop a keyword spice for each information domain a user may be interested in. Even if such a collection of keyword spices could be developed, the problem remains unabated as the users need to select correct keyword spices for their information needs.

This paper presents an algorithm to construct web queries that fit the query interface of Goolge search engine. The algorithm uses Incremental Learning (IL) algorithm (Sanchez, Triantaphyllou, Chen and Liao, 2002) as modified in Malhotra et al. (2005) to define an initial set of minterms. These minterms collectively select all relevant examples and each minterm rejects examples marked as irrelevant by the viewing searcher. The algorithm chooses and organizes these minterms to devise query that accesses the best resources as measured by the precision and recall characteristics and yet are small to meet the size limit of the search engine. Cohen et al. (1996) also generate a set of minterms using a rule-learning technique RIPPER (Cohen, 1995). However, the RIPPER expressions are already optimized to minimize misclassification errors. This makes the expression less amenable to further transformations to reduce their sizes to meet limitations of search engine query interfaces.

In section 2, we briefly introduce relevance feedback approaches used in text-categorization and explain why we have chosen to use Boolean expression based query synthesis approach. The section also summarizes IL algorithm to construct a set of minterms. This set of minterms constitutes the primary input for our query synthesis algorithm described in Section 3. The main goal of the algorithm would be to synthesize web query that fits the query interface of the search engine without unduly compromising its access to the best web resources. Section 4 presents results from a user survey to establish the effectiveness of the synthesized queries. In section 5 we conclude with a description of planned work to integrate the algorithm with a browser and also other applications.

2 RELATED BACKGROUND

Relevance feedback has been studied extensively in the context of text categorization (Baeza-Yates, 1991, Sebastiani, 2002). Given a corpus of documents, certain terms are chosen as discriminators. A query is a vector assigning weights to the terms. Relevance feedback and query expansion are used to adjust the terms and their weights so that query is more aligned to documents considered relevant and avoids documents considered irrelevant (Ruthven and Lalmas, 2003).

Notwithstanding their success and usefulness in text-categorization the vector based queries are little used for web searching. Vector queries do not express information needs in a way humans can easily interpret. Thus, search engines use Boolean expression based user interface for web searching. Vector based approaches also use a large number of terms in a query. On the other hand, it is important for the search engines to limit the terms in queries to deliver results efficiently and within an acceptable time frame.

A web query also differs from the text categorization in regards to its aims. An ideal text categorization query for an information need is required to locate all relevant documents without retrieving any irrelevant document. The practical algorithms – for example, RIPPER – aim for misclassification minimization using a cost model for errors. A web searcher typically views only a few (usually, one) documents – clearly, one would like these documents to meet the information need perfectly.

A *web search query* is a list of terms (words) punctuated by Boolean operators AND (&), OR (|) and NOT (!). Following on from Google conventions, AND is an implied operator and not explicitly written. Operator NOT applies to a single term and has highest precedence. The operator AND (&) has lowest precedence. For a set of textual documents, D, and a search query, Q, expression $D\ \sigma\ Q$ denotes the results of search by query Q over the document set D with the following interpretation:

Case *Q* is *term*	{doc | doc ∈ *D* and *term* occurs in document doc}
Case *Q* is *!term*	{doc | doc ∈ *D* and *term* does not occur in document doc}
Case *Q* is *(R & S)*	$(D \sigma R) \sigma S$
Case *Q* is *(R | S)*	$(D \sigma R) \cup (D \sigma S)$

We shall assume the readers familiarity with the standard terminology related to Boolean expressions (Aho and Ullman, 1992) especially the terms minterm, maxterm, Conjunctive Normal Form (CNF), and Disjunctive Normal Form (DNF).

Quality of a query is a subjective notion. Information retrieval systems measure the quality through two objective measures precision (P) and recall (R). Suppose a given collection of N documents containing I irrelevant documents is searched by a query that returns r relevant and n non-relevant documents. Precision (P) and recall (R) of the query are defined as follows: P = r/(r+n) and R = r/(N – I).

These definitions require knowledge of the various parameter values: N, I, n and r. The Web is large and ever expanding collection of documents. Therefore, it is impossible to know these values for a web search. We therefore use other measures, P@20 and C@20 to model precision and recall respectively (Patro and Malhotra, 2005). P@20 is defined as fraction of top 20 links returned by query that the searcher finds relevant. C@20 is defined based on the estimated number of documents (say, E) in the search engine database matching the query. C@20 is defined as $min(log_2(E \cdot P@20)/20,1)$. Briefly, $E \cdot P@20$ is the number of relevant documents the query is able to find in the search engines database. Logarithmic scale caters for reduced marginal utility of the larger sets. It can be easily noticed that for specific information need, a query with better recall returns higher C@20 value. We shall combine the two measures through harmonic mean to define a metric for quality of a query Q@20: Q@20 = 2/(1/P@20 + 1/C@20).

2.1 Building Blocks of a Query

For the sake of completeness, in this section, we briefly describe the algorithm to generate minterms needed to synthesize the queries. Further details of the algorithm can be found in (Patro, 2006).

We assume that a searcher seeking help to improve query has used an initial query to retrieve and view a few documents. As the documents are viewed the user divides them into two sets: *Relevant* containing documents that have some information of interest to the user; and, *Irrelevant* containing documents that have little information to interest the user.

The query synthesis algorithm makes use of the initial query and a set of minterms derived from documents in sets *Relevant* and *Irrelevant* to synthesize a new query. To construct this set of minterms, the algorithm first constructs a series of maxterms. Each maxterm is derived by selecting a series of terms from the documents in set *Relevant*. This selection of the terms to augment maxterms is facilitated by computing selectivity of the terms. The selectivity for term `t` in construction of $(i+1)^{st}$ maxterm is defined as follows (In the following $maxterm^p$ denotes a partially constructed $(i+1)^{st}$ maxterm):

TR = *Relevant* – (*Relevant* σ $maxterm^p$);

TR_t = TR σ t;

TIR = *Irrelevant* σ ($maxterm_1$ &…& $maxterm_i$));

TIR_t = TIR σ t;

selectivity (t) =

$((|TR_t|)(|TIR|-|TIR_t|))/(((|TR|-|TR_t|+1)(|TIR_t|+1))$.

A maxterm selects all documents in set *Relevant* and reject one or more documents in set *Irrelevant*. When enough maxterms have been found such that each irrelevant document is rejected by at least one maxterm, a Boolean expression is formed by taking conjunction of initial user query and constructed maxterms. The expression is simplified to its DNF form. Minterms that do not select any relevant document are dropped from this expression. The set of remaining minterms constitute the required set of minterms for query synthesis algorithm described in the next section.

The construction process ensures that each minterm rejects every irrelevant document viewed by the searcher. Collectively the minterms in the set select every relevant example identified by the searcher. This set of minterms is called *MSet*.

We illustrate this with an example. Suppose the following prose describes information need of a student.

> *We are to search information about plant Eucalyptus.*
>
> *A page containing any one of the following criteria in a brief description is treated as relevant:*
>
> *• A relevant page should contain the herb information or description of the plant.*
>
> *• The page should provide information relating to its use or growing of the plant.*
>
> *The page containing the word Eucalyptus but not related to above criteria may be treated as irrelevant.*

The student uses keyword *eucalyptus* to retrieve a number of documents and mark them as relevant and irrelevant based on the information needs

described above. The student viewed 71 text documents and marked 23 relevant and 48 irrelevant documents. We will continue to pretend, for the sake of example, that the student used them to devise a better search query. During a test run of the algorithm, five maxterms were constructed before all retrieved irrelevant documents could be rejected by at least one maxterm. The resulting CNF Boolean was as follows: *(eucalyptus) (fruit | tall | cream | drought | asthma) (tree | evergreen | alcohol) (gum | south | blue | book) (white | found | green) (plant | long | ground | index)*. In turn, DNF expression had 720 minterms of which 634 were found to be non-redundant. These 634 minterms define the MSet for synthesizing the query.

3 QUERY SYNTHESIS

The minterms in a non-redundant DNF query collectively select all documents in set *Relevant*. However, a document may be selected by several minterms. For example, the query concerning eucalyptus has 23 relevant documents, but has 634 non-redundant minterms in the Boolean expression. No more than 23 minterms are needed to select 23 documents.

A minimal set of minterms that selects each document in set *Relevant* is all that is needed to derive a complete search query. As the minimization problem is NP-hard, we use a heuristic to construct a compact cover for set *Relevant*. The query is derived from this DNF expression.

The main idea used in the implemented algorithm is to add one minterm into an incomplete cover set at a time. At each augmentation step, for each candidate minterm function *gain* is computed based on the number of new relevant documents that the minterm selects and its effect on query size. The minterm providing maximum gain is added to the cover set.

To construct a Google query from a given set of minterms the algorithm determines a term that occurs most frequently in the minterms. The term is factored out of the minterms using the following equivalence rule: (A B) | (A C) ≡ A (B | C).

The application of this rule partitions the original set of minterms in two sets of minterms. The first set of minterms is comprised of the minterms that do not include the term. The second set of minterms is created as the result of factoring of the term from minterms which included the factored term. The algorithm is applied recursively to two sets of minterms to achieve further reduction in the size of the query.

3.1 Trading Precision for Query Size

Many queries generated by the algorithm are small to fit the constraints imposed by the search engines. The oversize query for our example was (assuming old Google limit of 10 terms in a query): *eucalyptus ((tall white) | (gum (green | alcohol)) | (evergreen blue) | (fruit south) | (cream found))*. The oversized queries need to be trimmed in size. The trimming, however, would adversely affect the recall or precision of the query.

The preservation of recall rather than the precision is preferred as it generates a query that integrates the aspects of the documents in set *Relevant* into the synthesized query. The property is useful as the synthesized query benefits from all examples, even those that only partially satisfy the user's information needs. A precision-centric approach would require each document in set *Relevant* to be fully satisfying user's information needs causing an obvious paradoxical demand on the user.

Each minterm used to construct the web query rejects every document in set *Irrelevant*. Thus, each minterm has terms to reject every document in set *Irrelevant*. A minterm can be reduced in size, by dropping one or more of these terms from it. Such a transformation, affects the minterm in two ways: (i) A reduced-size minterm either selects the same set of relevant documents as the original minterm or selects some additional documents from set *Relevant*. (ii) The reduced-size minterm selects some documents from set *Irrelevant*. The original minterm selected no document from this set.

In turn, a search query constructed from the reduced-size minterms is affected as follows:

- A search query may need a smaller number of reduced-size minterms to cover all documents in set *Relevant*.
- The search query is composed of minterms with fewer terms in them.
- The search query fails to reject all documents in set *Irrelevant*.

Thus, the size of the constructed query will be smaller but will be of lower precision.

The query synthesis approach uses the reduced-size minterms that have best benefit-to-cost ratios. The cost is measured by the number of irrelevant documents selected by the reduced-size minterm and the benefit is measured by reduction in the number of terms in the constructed query.

A *quality* value – roughly representing a reciprocal of the cost – is assigned to each minterm to measure its ability to avoid irrelevant documents. The *quality* of minterm *mt* is defined by fraction: $|Relevant \sigma mt| / |Irrelevant \sigma mt|$. The *quality* of a

set of minterms is defined by the minimum *quality* of a minterm in the set.

The *benefit* is represented by the (incremental) change in the query size for each new relevant document selected. Since small query sizes are desired, synthesis algorithm used a reciprocal function *gain*. For minterm *mt*, gain is defined as *(number of new relevant documents selected by mt for the query)/ (change in the query size resulting from the introduction of mt in the query)*.

The set of all reduced-size minterms (called, *RSet*) is derived from set *MSet* by taking a minterm from *MSet* at a time. The reduced-size minterms are added to *RSet* by deleting one or more (but not all) terms from the selected minterm. Thus, each minterm will add 2^n-1 reduced-size minterms in set *RSet*, where n is the number of terms in the minterm.

Finally, we systematically construct a series of search queries till we obtain one that meets the size constraint of the search engine. At each stage, we choose a cut-off value for attribute *quality*. The first cut-off value chosen is infinity, representing the case where the original *MSet* is used as the set of minterms to construct the query. In the subsequent runs, minterms exceeding the chosen *quality* cut-off are chosen from *RSet* to define minterm set for input to query synthesis algorithm. The set is cleaned first to remove dominated minterms. A minterm is dominated if another minterm with a smaller number of terms selects all relevant documents selected by the dominated minterm. If the synthesized query is not suitable for the search engine, the next lower value for attribute *quality* is chosen to construct another candidate query.

To illustrate the procedure with *eucalyptus* example:

- The first query synthesized at quality level infinity was: *eucalyptus ((tall white) | (gum (green | alcohol)) | (evergreen blue) | (fruit south) | (cream found))*. Clearly this query with 12 words is an oversized query for Google search engine.
- Selecting the next highest *quality* value which is 12.0, the query obtained is: *eucalyptus ((fruit | (tall white) | (evergreen (gum | blue)) | (alcohol gum) | (cream found)))*. This query has 11 words, which is again an oversized query for Google search engine.
- The next lower *quality* value is 11.0. At this *quality* value, the synthesized query is: *eucalyptus (fruit | tall | (gum (white | alcohol) | (evergreen blue))*. This query with 8 words is acceptable to the Google search engine.

4 RESULTS AND DISCUSSION

A web search query synthesized from only a few relevant documents is not expected to be precise. The deficiency results as small number of available relevant documents may not train the CNF Boolean expression adequately. As the number of relevant documents used to train the Boolean expression increases the precision of the synthesized query improves. The recall also benefits from the increase in the number of relevant example documents available for query synthesis. Different relevant documents tend to use different vocabulary. The diversity in vocabulary introduces new terms, including synonyms and related words, to the synthesis process.

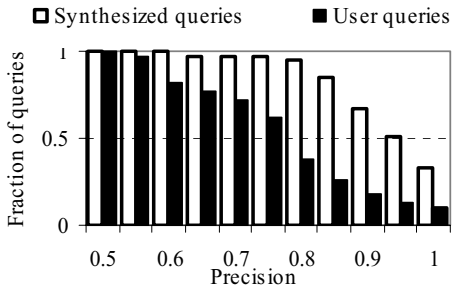
Our initial experience suggests that query synthesized from six relevant documents gives a precision of about 50%. This level is satisfactory as several surveys have pointed that a successful web search is followed by an access to only one document (Jansen, Spink, Bateman and Saracevic, 1998). Queries synthesized from sets of 8 or more relevant documents out-perform experienced human searchers.

4.1 User Survey

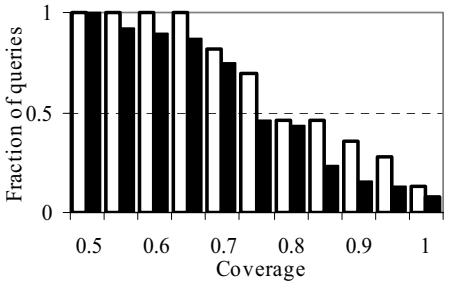
To compare the effectiveness of the synthesized queries against human queries we conducted 39 search sessions involving 25 topics. For each topic, the intended information need was described in a form similar to one illustrated for eucalyptus example in Section 2. Documents for the first 70 links returned by Google when searched using (single-word) topic title as keyword were downloaded. Our goal was to get some 10 to 20 relevant text documents for each topic to synthesize queries. Since many of these 70 links where defunct or non-text documents, for each topic a different number of usable text documents remained for query synthesis. For each topic, available text documents were classified manually by the authors. The number of relevant documents for topics ranged from 2 to 44 with average of 17 documents a topic. Average number of irrelevant documents for a topic was 38. These collections were used to synthesize queries.

Table 1: Statistical summary of the performances of user defined queries and synthesized queries on precision (P), coverage (C) and quality (Q) metrics.

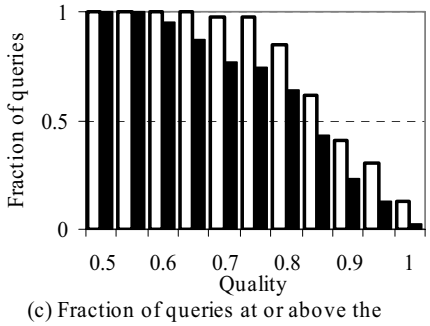
Metrics	User query			Synthesized query		
	P	C	Q	P	C	Q
Min	0.50	0.42	0.50	0.60	0.63	0.65
Average	0.75	0.72	0.72	0.91	0.79	0.84
Max	1.00	1.00	0.91	1.00	1.00	0.98
Median	0.75	0.70	0.72	0.95	0.74	0.84
Std. Dev.	0.14	0.14	0.11	0.09	0.13	0.09



(a) Fraction of queries at or above the precision values



(b) Fraction of queries at or above the coverage values



(c) Fraction of queries at or above the quality values

Figure 1: Distribution of performance metrics for precision, coverage and quality metrics for user devised and synthesized queries from survey sessions.

The survey volunteers were all university students and staff with extensive experience with the web searching. For a chosen topic, a volunteer devised a web query based on the description given to them. The volunteer formed the query interactively and used web search to test queries as they formed them before identifying their best query. We then gave the volunteer the synthesized query. Using the volunteer's query and the synthesized query separate set of 20 top documents were retrieved. These documents were then classified by the volunteer based on their understanding of the information need presented in the description statement. The arrangement overcomes bias in classification due to interpretation of the information need or influence from the classifications made by the authors in forming the synthesized queries.

The statistical summary of data collected during the survey is presented in Table 1. Figure 1 shows cumulative distributions of precision (P@20), coverage (C@20) and quality (Q@20) of user devised (volunteer) queries and synthesized queries. It is clear from the figure that for any specified precision, coverage and quality threshold fraction of synthesized queries performing better than human queries is better.

Figure 2 shows the same data but compares data for user and synthesized query from the same survey session. The bars shown in the graphs represent differences in performance of two queries (user and synthesized) on the chosen metrics. Again synthesized queries have performed better more often than the volunteers' queries.

Hypotheses tests on paired data confirm that the synthesized queries perform better than the user queries on precision and quality metrics at 99.99% confidence. The confidence level judging the better performance for synthesized query on coverage metrics is lower at 99%. It is worth stressing that coverage without a supporting precision does not make a query better – an inane query *TRUE* has 100% coverage but 0% precision for every search.

4.2 Discussion

Only in two survey sessions human queries performed better than the synthesized queries on precision metrics. One of these sessions searched for information on Australian marsupial *Kangaroo*. The synthesized query reads: *kangaroo ((tail pouch) | (feet found) | (user grass))*. The human query that gave better precision was *kangaroo (macropodiade | animal)*. The precision of the synthesized query was

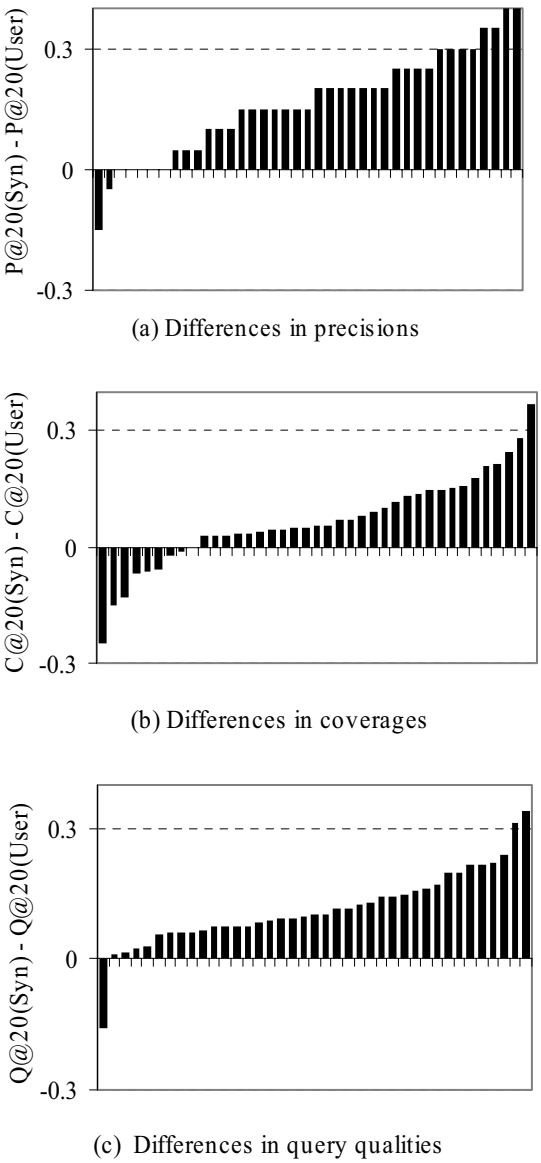


Figure 2: Performance differences between user and synthesized queries noticed in survey sessions. Each metrics is sorted independently.

0.85 and of the user query 0.9. The coverage for the two queries was 0.928 for the human query, and 0.678 for the synthesized query. The search was repeated in another session. In the latter search, the human devised query was *(kangaroo animal Australia)* with precision 0.7 and coverage 0.827. The synthesized query had a precision of 0.9 and coverage of 0.677 in this session.

In our post survey analysis, it was noted that there are a large number of business and commercial web-sites using term *kangaroo*. The synthesized query has correctly abstracted the nature of the desired information from the given relevant documents. The first volunteer has avoided these business sites using an uncommon but a Kangaroo specific term. The second searcher has been less successful. The combined harmonic mean score, Q@20, for the first volunteer query (0.914) correctly rates it better than the other two queries with scores of around 0.76.

The other case giving better precision by a human query *(ostrich bird struthio neck)* over a synthesized query *ostrich ((((bird largest) | neck) world) | ((neck | (bird brown)) ground)* searched for information about Ostrich. Precision for the human query was 1.0 and coverage 0.545. The synthesized query had precision 0.85 and coverage 0.699. A different volunteer used query *(ostrich Africa flightless largest)*. This gave the precision of 0.85 and coverage of 0.601. However, both human queries are narrowly focused and are less general than the synthesized query. One of them is focused exclusively on *neck* and the other on size (*largest*) of the bird.

It is interesting to note that keywords *macropodiade* and *struthio* are not common vocabulary of a typical web searcher. We believe that these volunteers were very motivated, knowledgeable persons in the topic of their search. Anecdotally, both these searches occurred in the earlier stages of the survey when volunteers had many topics available to choose from. We believe these volunteers searched topics that were close to their interests and knowledge.

The algorithm was developed and tested through survey using the old Google limit of 10 terms in a query. The limit has since been raised to 32. The concession, however, does not make the part of the algorithm that reduces query size unnecessary. Compact query is necessary for good precision and recall. Well targeted terms in the query help the ranking algorithms in the search engines to order the links well. Shorter queries are also easily understood by the human searchers.

5 CONCLUDING REMARKS

The synthesized queries have performed better than queries that humans could devise. The links returned by the synthesized queries are different and much expanded collection from those used in the synthesis. The links to the documents used in synthesis are not easily located in those returned by the new query. The correctness of the learned abstraction is evident by the high precisions achieved by the synthesized queries.

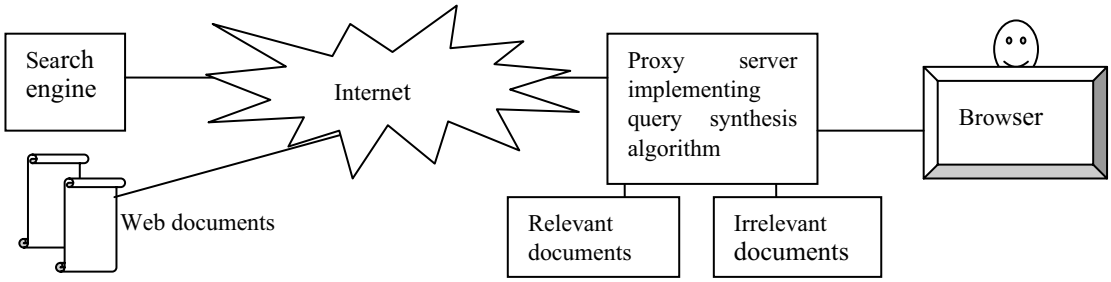


Figure 3: Architecture for integrating query synthesizer with web browser.

Figure 3 gives a brief outline for integrating the algorithm with a web browser. All traffic between the browser and the Internet flows through a proxy that records the documents and the user's feedback. The proxy augments the displayed document with additional buttons for interaction with the searcher.

A criticism of our approach could be made based on the perception that the searchers may not be willing to spend time and effort the documents. We acknowledge that the majority of searches will be satisfactorily and efficiently serviced through the standard search engine interfaces. However, the hard and frustrating searches, albeit uncommon, do need help. We believe that our approach provides a mechanism for such searches without reducing the experiences of the searches that do not require this help.

The algorithm described in this paper has other applications too. It can provide a convenient way for locating related emails. Other application area includes search for all related files on a file system using tools like Google Desktop.

REFERENCES

Aho, A. V. and Ullman, J. D., 1992. Foundations of Computer Science. Computer Science Press. NY.

Aula, A., Jhaveri, N. and Kaki, M., 2005. Information Seach and Re-Access Strategies of Experienced Web Users. In the intl World Wide Web (WWW2005) conf. ACM, NY.

Baeza-Yates, R., and Riberio-Neto, B., 1991. Modern Information Retrieval, Addison-Wesley. Reading, Ma,

Cohen, W. W., 1995. Fast Effective Rule Induction. In 12th Intl. Conf. on Machine Learning.

Cohen, W. W. and Singer, Y., 1996. Learning to Query the Web. In AAAI-96 Workshop on Internet-Based Information Systems. AAAI Press, Menlo Park, CA.

Hölscher, C. and Strube, G., 2000. Web Search Behavior of Internet Experts and Newbies. In Proc. of the 9th intl. World Wide Web conf. on Computer networks, : the intl. journal of computer and telecommunications networking. North-Holland Publ.

Jansen, B. J., Spink, A., Bateman, J. and Saracevic, T., 1998. Real Life Information Retrieval: A Study of User Queries on the Web. SIGIR Forum, 32 (1), 5-17.

Kopec, D. and Marsland, T. A., 1997. Search. The CRC Press, Inc.

Malhotra, V., Patro, S. and Johnson, D., 2005. Synthesise Web Queries: Search the Web by Examples. In 7th Intl Conf. on Enterprise Information Systems (ICEIS2005), Volume 2. INSTICC, Portugal.

Oyama, S., Kokubo, T. and Ishida, T., 2004. Domain-Specific Web Search with Keyword Spices. IEEE Transaction on Knowledge and Data Engineering, 16 (1), 17-27.

Patro, S. and Malhotra, V., 2005. Characteristics of the Boolean Web Search Query: Estimating Success from Characteristics. In 1st intl conf. on web info. systems and technologies (WEBIST2005). INSTICC, Portugal.

Patro, S., 2006. Synthesising Web Search Queries from Example Text Documents. Master of Science Thesis, School of Computing, University of Tasmania, Launceston. Website: eprints.comp.utas.edu.au.

Ruthven, I. and Lalmas, M., 2003. A Survey on the Use of Relevance Feedback for Information Systems. Knowledge engineering Review, 18 (2), 95-145.

Sanchez, S. N., Triantaphyllou, E., Chen, J. and Liao, T. W., 2002. An Incremental Learning Algorithm for Constructing Boolean Functions from Positive and Negative Examples. Computers and Operations Research, 29 (12), 1677-700.

Sebastiani, F., 2002. Machine Learning in Automated Text Categorization. ACM Comp. Surveys, 34 (1), 1-47.

AN ALGORITHM TO USE FEEDBACK ON VIEWED DOCUMENTS TO IMPROVE WEB QUERY

Enabling naïve searchers to search the web smartly

Vishv Malhotra
University of Tasmania, Hobart
(Co-authors: Sunanda Patro, David Johnson)

The paper presents

- An algorithm to devise Boolean search query to better exploit search engine interfaces
  - From text examples
  - To make the search precise (fewer false hits)
  - Improve the search recall (miss less matches)
  - *Allows imperfect examples*

Web Search Interface

- Boolean expression based
- Constructing Boolean expressions to exactly match an information need is very difficult
  - Best expression seeks “every right resource” and “only right resources”
- Humans use keyword lists with rudimentary adoration to express needs
- Modern search engines mostly succeed

Why we need help?

- … Frequent success in searching does not mean
  - No case of frustrating or unsuccessful search, or
  - People find the Web searching effortless, or
  - Satisfactory resources are easily located

Measurements of Success

- Precision
  - Did we find only the right resources?
  - What fraction of the searched documents are relevant?
  - P@20

Measurements of Success

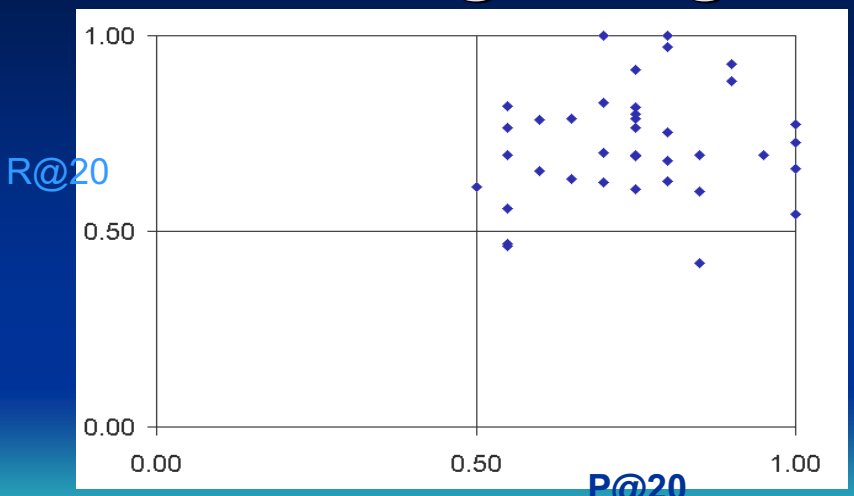
- Recall
  - How many (potential) relevant documents would the query locate?
  - R@20 (saturating at 1)

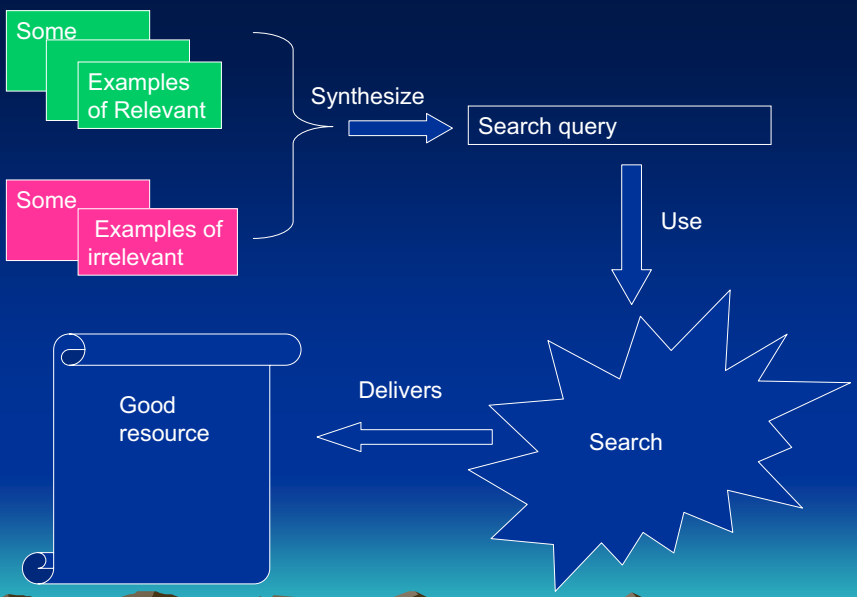
Utility gain decreases with size

Count of good resources accessed

$$\frac{1}{20}\log_2\left(P@20 \times Expected \quad matches\right)$$

Human Queries: P@20 vs R@20





Algorithm

- Identify attributes (keywords) that separate relevant examples from irrelevant examples
- Combine the keywords as Boolean expression to fully separate two groups of examples
- An improved version of this Boolean expression is a search query

Example Run of Algorithm

- Original query keyword: eucalyptus
  - Used to download example documents
  - Categorize examples as *Relevant* and *Irrelevant*

Example Run of Algorithm

- Construct maxterm (=sum of keywords) to select all *Relevant* and reject some *Irrelevant*
  - fruit + tall + cream + drought + asthma
- Repeat step with remaining Irrelevant
  - tree + evergreen + alcohol
  - gum + south + blue + book
  - white + found + green

Example Run of Algorithm

- Take product of maxterms to create minterms (=products of keywords)
- Discard minterms that are
  - Ineffective: selects no relevant
  - Dominated: not as good as some other minterm
  - Simplify minterms as needed – drop keywords from minterms, if useful

Example query

- Devise candidate search query by selecting *simplified* minterms to cover all Relevant examples
  - eucalyptus (tall + fruit + gum white + alcohol gum + evergreen blue)
- Or
  - eucalyptus (fruit + tall + (gum (white + alcohol)) + (evergreen blue))

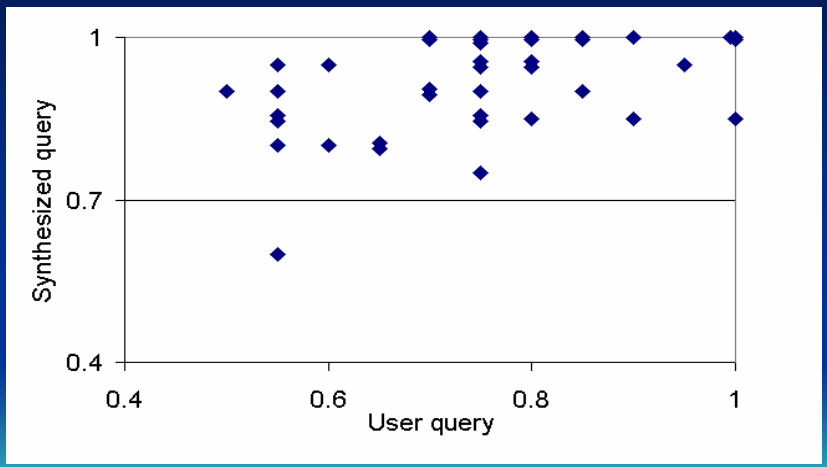
User Survey

- We pre-synthesized queries for some topics
- Let (experienced) volunteers search
  - A topic
  - Then use our query
  - User tells us P@20 for two cases

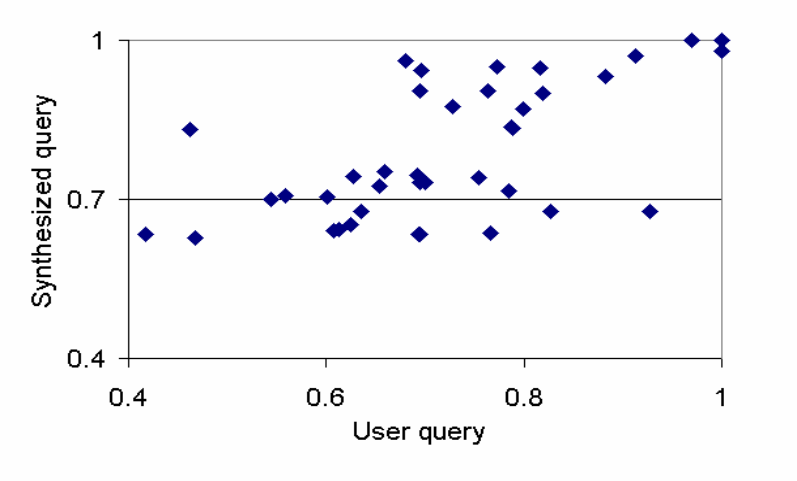
Topic List

- *Amoeba*
- Angle
- Cobra
- Eucalyptus
- Graphite
- Grasshopper
- Igloo
- Kangaroo
- Kohinoor
- Konark
- Lava
- Leopard
- Lotus
- Mango
- Nile
- Ostrich
- Ozone
- Pyramid
- Radium
- Rainbow
- Sun
- Titanic
- Veena
- Vitiligo

Scatter Chart P@20



Scatter Chart R@20

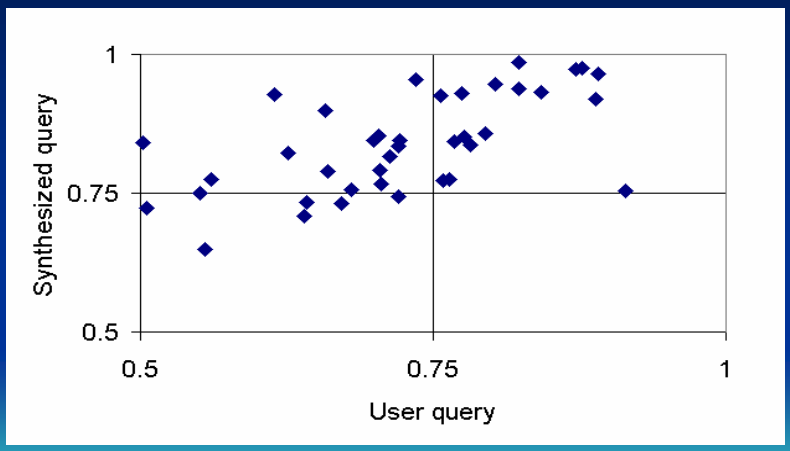


Measurements can be Fooled

- Precision can be increased arbitrarily by targeting very narrowly
- Recall can be increased by accepting every resource
- Combined measure

  Q@20 = 2/(1/P@20 + 1/R@20)

Scatter Chart Q@20



Conclusion

- Algorithm provides a way to construct effective queries
- We are working on combining algorithm as browser add-on
  - Using concurrent Java programs we could download and process text for about 100 links in 120-130 seconds over DSL line using PC (2.4G Intel+512MB) with 22Mbps wireless

Questions

... We believe we can support searchers when they need help in a difficult Web searching situation

For more information: eprints.comp.utas.edu.au

Synthesized vs Human Query

- In one case a human query performed better than synthesized
- Search for information on kangaroo had human query:
  - *kangaroo (macropodiade | animal)*
- Commercial sites using term *kangaroo* do not use term *macropodiade* that "biology" related sites use