Abstract. Query being the essential in an Information Retrieval (IR) system and user looking to fulfil information need has to formulate a query usually consisting of a small set of keywords summarizing the information need. It reiterates the significance of the query processing in an information retrieval system. The main objective of query processing is to translate a user level query into a low level machine understandable query. The main purpose of this paper is to analyse query clustering along with the query processing and semantic web. Query clustering being one of the emerging techniques and query processing being one of widely used in database systems makes an interesting combination to be analysed. In this paper an attempt is made to evaluate the performance of query clustering along with query processing algorithms in each of the category. The evaluation was based on dataset as specified by Forum for Information Retrieval [FIRE11].

Keywords: Query expansion, Query optimization, Query parsing, Query clustering and Web semantics.

1. Introduction
The main purpose of an IR system is to retrieve all the documents which are relevant to a user query while retrieving as few non-relevant documents as possible. The typical Information Retrieval (IR) model of the search process consists of three essentials: query, documents and search results. A user looking to fulfil information need has to formulate a query usually consisting of a small set of keywords summarizing the information need. Queries are posed by interactive users and can be ambiguous in nature. An interactive query goes through the entire path of query parser, expansion, optimization, and processing [1]. Although the existing methods may suggest good queries in some scenarios, none of them models the immediately preceding queries as context systematically, and uses context information effectively in query suggestions. Context-aware query suggestion is challenging in both modelling context and scaling up query suggestion using context [13]. To overcome this issue a query clustering using users search history is suggested. Along with query clustering technique the query processing technique is used. In the query processor, a user’s queries and data modification commands, turns into a query plan – a sequence of operations (or algorithm) on the database from high level queries to low level commands. Complex queries are becoming commonplace, with the growing use of decision support systems. These complex queries often have a lot of common sub-expressions, either within a single query, or across multiple such queries run as a batch.

A query parser, basically, translates users search string into specific instructions for the search engine. It stands between user and the documents users are seeking, and so its role in text retrieval is vital. This helps in increasing the precision of the search engine. Semantic Web technologies enable people to create data stores on the Web, build vocabularies, and write rules for handling data. Linked data are empowered by technologies such as RDF, SPARQL, OWL, and SKOS [15]. The main purpose this being used is empower the users and hence making search a personalised one.
The main objective to use query optimization helps to exploit the common sub-expressions and hence to reduce the evaluation cost. In this paper an attempt is made to analyse the performance based on the importance of query clustering along with each step in query processing i.e. the query optimization, query expansion, query parsing and semantic web by implementing some of the algorithm from each of the mentioned categories. Experimental results shows that the significance of each steps in query processing along with query clustering. To evaluate the quality and effectiveness of query processing, the two basic measures for information retrieval are used.

**Precision:** The fraction of retrieved documents that are relevant to the user query intent.

**Recall:** The fractions of relevant documents that are retrieved are in context with user query.

### 2. Related Work

Amit Goyal *et al.* analyses exponential growth in number of possible strategies with the increase in number of relations in a query has been identified as a major problem in the field of query optimization of relational databases. But as the size of a query grows, exhaustive search method itself becomes quite expensive. By modifying the $A^*$ algorithm to produce a randomized form of the algorithm and compared it with the original $A^*$ algorithm and exhaustive search [3].

Yannis E. Ioannidis primarily discuss the core problems in query optimization and their solutions, and only touch upon the wealth of results that exist beyond that. More specially, author concentrates on optimizing a single at SQL query with ‘and’ as the only Boolean connective in its qualification (also known as conjunctive query, select-project-join query, or non-recursive Horn clause) in a centralized relational DBMS, assuming that full knowledge of the runtime environment exists at compile time [4].

Prasan Roy *et al.* demonstrates that multi-query optimization using heuristics is practical, and provides significant benefits. They propose three cost-based heuristic algorithms: Volcano-SH and Volcano-RU, which are based on simple modifications to the Volcano search strategy, and a greedy heuristic [5].

Joseph M. Hellerstein defines a query cost framework that incorporates both selectivity and cost estimates for selections. The algorithm is called Predicate Migration, and proves that it produces optimal plans for queries with expensive methods [9].

Francesco Colace *et al.* proposes a query expansion method to improve accuracy of a text retrieval system. The technique makes use of explicit relevance feedback to expand an initial query with a structured representation called Weighted Word Pairs. Such a structure can be automatically extracted from a set of documents and uses a method for term extraction based on the probabilistic Topic Model [2].

Joshua Goodman proposes two new algorithms: the *Labelled Recall Algorithm*, which maximizes the expected Labelled Recall Rate, and the *Bracketed Recall Algorithm*, which maximizes the Bracketed Recall Rate [6].

Adrian D. Thurston *et al.* presents two enhancements to a basic backtracking LR approach which enable the parsing of computer languages that are both context-dependent and ambiguous [8].

Kaarthik Sivashanmugam *et al.* develops a semantic Web services where by the Web services are annotated based on shared ontologies, and use these annotations for semantics-based discovery of relevant Web services. They also discuss one such approach that involves adding semantics to WSDL using DAML+OIL ontologies. And also uses the approach for UDDI to store these semantic annotations and search for Web services based on them [7].

Zhen Liao *et al.* proposes summarization of individual queries into *concepts*, where a concept consists of a small set of queries that are similar to each other. Using concepts to describe contexts, we can address the sparseness of queries and interpret users’ search intent more accurately [13].

Kenneth Wai-Ting Leung *et al.* proposes a new two phase personalized agglomerative clustering algorithm that is able to generate personalized query clusters [14].

### 3. Query Clustering

Query clustering is a process of grouping similar queries into a concept by measuring the similarity between two queries. In all of the below mentioned algorithms the queries are clustered based users search intent i.e, by using the users search log.

From the query clustering table 3 we infer that QSC-IS algorithm out performs all the algorithms shown in the table. It’s been observed by iterative scanning of clusters increases both precision by 20% and recall by 25%.
4. Query Optimization

Query optimization is a function of many relational database management systems in which multiple query plans for satisfying a query are examined and a good query plan is identified. The improved $A^*$ algorithm, when used for query optimization, gives output comparable to exhaustive search in minimal amount of search space. Improved $A^*$ algorithm uses two linked lists instead of one used in original $A^*$ algorithm. It also considers global costs. Algorithm for optimizing $n$ relations performs a large number of local optimizations. Each one starts at a random node and repeatedly accepts random downhill moves until it reaches a local minimum and it also returns the local minimum with the lowest cost found.

The three cost-based heuristic algorithms: Volcano-SH and Volcano-RU, which are based on simple modifications to the Volcano search strategy, and a greedy heuristic. The greedy heuristic incorporates novel optimizations that improve efficiency greatly.

To optimize a tree, all predicates are pushed down as far as possible, and then repeatedly apply the Series-Parallel Algorithm Using Parallel Chains to each stream in the tree, until no more progress can be made.

From the query optimization table we infer Algorithm for optimization $n$ relations performs in both the basic measures of Precision and the recall criteria of IR. The algorithm shows a precision of 21.3% and a recall of 41.73% which is comparatively higher when compared with other algorithms in the table 2.

5. Query Expansion

In WWP feature selection the aim is to extract from a set of documents a compact representation, named Weighted Word Pairs (WWP), which contains the most discriminative word pairs to be used in the text retrieval task. In the Relations Learning stage, where graph relation weights are learned by computing probabilities between word pairs and in the Structure Learning stage, where an initial WWP graph, which contains all possible relations between aggregate roots and aggregates, is optimized by performing an iterative procedure.

<table>
<thead>
<tr>
<th>Table 1. Query clustering.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query clustering</td>
</tr>
<tr>
<td>Query Stream Clustering (QSC)</td>
</tr>
<tr>
<td>Query Stream Clustering with Iterative Scanning (QSC-IS).</td>
</tr>
<tr>
<td>Agglomerative Clustering</td>
</tr>
<tr>
<td>Personalised Agglomerative Clustering</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2. Query optimization.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization Algorithms</td>
</tr>
<tr>
<td>A* algorithm[3].</td>
</tr>
<tr>
<td>Algorithm for optimizing $n$ relations[3][4].</td>
</tr>
<tr>
<td>Volcano-SH</td>
</tr>
<tr>
<td>Volcano-RU</td>
</tr>
<tr>
<td>a greedy heuristic</td>
</tr>
<tr>
<td>Predicate Migration[9]</td>
</tr>
</tbody>
</table>
Query Processing Along with Query Clustering Analysis in Information Retrieval

Table 3. Query expansion.

<table>
<thead>
<tr>
<th>Query Expansion</th>
<th>Input</th>
<th>Output</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>WWP feature selection-relations learning stage</td>
<td>set of documents</td>
<td>vector of weighted word pairs g</td>
<td>26.29%</td>
<td>41.63%</td>
</tr>
<tr>
<td>the Structure Learning stage</td>
<td>a starting WWP structure</td>
<td>the set of parameters t which produces the best WWP graph</td>
<td>4.49%</td>
<td>18.90%</td>
</tr>
</tbody>
</table>

In query expansion, when the algorithm WWP feature selection-relation stage is implemented with the structure learning stage algorithm there is considerable improvements in the both the precision and recall parameter of IR. It can be observed that in table 2 that precision has almost increased by 22% and recall parameter doubled by 20%.

6. Query Parsing

The Labelled recall algorithm maximises expected recall rate and Bracketed recall which maximises the bracketed recall rate.

Table 4. Query parsing.

<table>
<thead>
<tr>
<th>Query Parsing</th>
<th>Input</th>
<th>Output</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labelled recall algorithm [6]</td>
<td>Tree $T_g$</td>
<td>MAXC(1,n) contains the score of best parse, where n is length of the tree</td>
<td>10.94%</td>
<td>18.45%</td>
</tr>
<tr>
<td>Bracketed Recall [8].</td>
<td>Tree $T_g$</td>
<td>Max-g. bracketed recall rate</td>
<td>9.80%</td>
<td>21.48%</td>
</tr>
</tbody>
</table>

In the query parsing when labelled recall algorithm is implemented with the Bracketed recall algorithm considerably improves in the recall parameter by approximately 3% even though the precision parameter remains almost the same and shown in table 3.

7. Web Semantics

The Semantic Web is a collaborative movement led by the international standards body, the World Wide Web Consortium (W3C). The standard promotes common data formats on the World Wide Web [12]. By encouraging the inclusion of semantic content in web pages, the Semantic Web aims at converting the current web dominated by unstructured and semi-structured documents into a “web of data”. The Semantic Web stack builds on the W3C’s Resource Description Framework (RDF).

Adding Semantics in WSDLs to develop semantic Web services where by the Web services are annotated based on shared ontologies, and use these annotations for semantics-based discovery of relevant Web services. One such approach that involves adding semantics to WSDL is using DAML+OIL ontologies. Adding Semantics in UDDI is used to store these semantic annotations and search for Web services based on them Semantic Web Service Discovery, Semantic annotations added in WSDL and in UDDI are aimed at improving discovery and composition of services involves ranking based on the semantic similarity between the precondition and effect concepts of the selected operations and preconditions and effect involves ranking based on the semantic similarity between the precondition and effect concepts of the template [7].

In the web semantics, when semantics is added to the WSDL, UDDI and Web Service Discovery there is considerable increase in the precision and recall parameters. Hence adding web semantics to the query shows a considerable improvement in performance in IR.

8. Experimental Results and Evaluation

The experiments were conducted on the FIRE system; all the algorithms in each category of query processing, web semantics and spelling errors in query were implemented in java. A Collection of the TREC WebTrack [13] and we measured precision and recall over a set Indian statistical institute, Kolkata. Precision was measured by precision at 10 (P@10) and mean average precision [10].

© Elsevier Publications 2013. 431
Table 5. Semantic web table.

<table>
<thead>
<tr>
<th>Web Semantics</th>
<th>Input</th>
<th>Output</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adding Semantics in WSDL</td>
<td>Message parts using XML schema constructs</td>
<td>Mapping Message Parts to Ontological Concepts using XML schema constructs</td>
<td>39.46%</td>
<td>51.67%</td>
</tr>
<tr>
<td>Adding Semantics in UDDI</td>
<td>the second Model represent the ontologies of input</td>
<td>the third Model represent the ontologies of output</td>
<td>29.85%</td>
<td>48.07%</td>
</tr>
<tr>
<td>Semantic Web Service Discovery</td>
<td>Phase 1: Web services (operations in different WSDL files)</td>
<td>matches Web services functionality provided</td>
<td>24.01%</td>
<td>39.10%</td>
</tr>
<tr>
<td></td>
<td>Phase 2: result set from the first phase</td>
<td>Ranked first phase result set basis of semantic similarity between the input and output concepts</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Phase 3: result set from the first phase</td>
<td>Ranking based on the semantic similarity between the precondition and effect concepts of the selected operations and preconditions and effect concepts of the template.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Precision is the proportion of relevant documents among the retrieved documents. Precision is defined as follows

\[
\text{Precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}
\]

The method for estimating the recall of the algorithms is counting the number of full evaluations required to return a certain number of top results. This measure has the advantage of being independent of the software and the hardware environment, as well as the quality of the implementation.

Recall is the proportion of retrieved documents among the relevant documents. Recall is defined as follows

\[
\text{Recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}
\]

In query clustering, QSC-IS provides better results for both precision and recall parameters. It has been observed that even though the iterative scanning take some time and memory, it’s worth the implementation because of its accuracy in results. Another noticeable difference in results is that it provides the suggestion according to the users context.

In query optimization Algorithm for optimization n-relations provides better results in both IR criteria precision and recall when compared to other implemented algorithms.

In query expansion, when WWP feature Selection-Relations Learning Stage implemented along with the Structure Learning Stage increases both precision and recall parameter of IR.
In query parsing when Labelled Recall algorithm is implemented along with the Bracketed recall increases recall parameter considerably.

In the web semantics, adding semantics to WSD, UDDI and Web Service Discovery has seen a visible increase in precision and recall parameters.
For approximate query processing, a web semantics and query clustering strategies (when we can have false negatives) we measure two parameters: First we measure the performance gain, as before. Second, we measure the change in recall and precision by looking at the distance between the set of original results and the set of results produced by the implemented algorithms.

9. Conclusion and Future Work

In this paper, our main objective is to analyse the performance of query clustering with query processing. Query processing includes query optimization, query expansion, query parsing and web semantics.

In our experiments that empirically validate our approaches in various stages of query processing. Specifically the experiments substantiate two premises.

- In order to estimate recall and precision we need to identify the set of values in a sample that represent a single real world object. In our approach these sets correspond to the result of the query processing. The experiments show that the result of the query processing may be used to improve efficiency of the search process.
- When the sample is big enough and similarity metrics are adequate for the column domain the recall/precision results are very similar to the actual recall/precision values obtained when querying the database.

In future, we intend to integrate both query clustering and query processing into an intelligent search engine model. We would also like to extend it to various applications such as document clustering and ranking.

References

[4] Yannis E. Ioannidis, “Query Optimization”, University of Wisconsin. Partially supported by the National Science Foundation under Grants IRI-9113736 and IRI-9157368 (PYI Award) and by grants from DEC, IBM, HP, AT&T, Informix, and Oracle, ACM, 1996.
[5] Prasan Roy et al., “Efficient and Extensible Algorithms for Multi Query Optimization”, work was supported in part by a grant from Engage Technologies/Redbrick Systems. Part of the work of Prasan Roy was supported by an IBM Fellowship, SIGMOD ’00 Proceedings of the 2000 ACM SIGMOD international conference on Management of data.
[6] Joshua Goodman, “Parsing Algorithms and Metrics”, Harvard university, research support from National Science Foundation Grant IRI-9350192, National Science Foundation infrastructure grant CDA 94–01024, and a National Science Foundation Graduate Student Fellowship, ACL ’96 Proceedings of the 34th annual meeting on Association for Computational Linguistics.
Query Processing Along with Query Clustering Analysis in Information Retrieval


© Elsevier Publications 2013.