The ConQuest System

Paul E. Nelson
VP of Research & Development

9705 Patuxent Woods Drive, Columbia, Maryland 21046
(410)-290-6290

Introduction

ConQuest software has a commercially available text search and retrieval system* called "ConQuest" (for Concept Quest). ConQuest is primarily an advanced statistical based search system, with processing enhancements drawn from the field of Natural Language Processing (NLP).

ConQuest participated in Category A of TREC, and so produced results for 50 test queries over the entire 2.3 Gigabyte database. In this category, we constructed queries and submitted results for two different ranking functions. These two functions tested the difference between local and global document relevancy, and are fully described later.

In TREC-2, ConQuest had a very strong showing. Our recall scores in particular improved by about 18 percentage points over the adjusted TREC-1 scores. Our precision scores were also very competitive.

The purpose of this paper is to discuss how we prepared for TREC-2: how queries were performed, what initial judgments were made and why, and interpretation of the results. Then, I will cover the tests which were performed after TREC-2, and how these tests clearly identify the areas where ConQuest could most effectively be improved.

System Architecture

For a complete discussion of the system architecture of ConQuest, see the TREC-1 conference proceedings, or call the author. The following overview is meant as a brief refresher.

ConQuest uses pre-built indexes to perform text database searches at fast speeds. In such a system, all text to be searched must first be indexed. These indexes are then used for all searching; the original document data is not required.

ConQuest uses a dictionary augmented with a semantic network for both indexing and queries. The dictionary is a list of words where each word contains multiple meanings. Each meaning contains syntactic information (part-of-speech, feature values), and a dictionary definition.

The semantic network contains nodes which correspond to meanings of words. These nodes are linked to other related nodes. Relationships between nodes are extracted from machine readable dictionaries. Some example relationship types include synonym, antonym, child-of, parent-of, related-to, part-of, substance-of, contrasting, and similar-to.

The ConQuest dictionary was generated automatically from several Machine Readable Dictionary (MRDs) sources, commercially available. This gives ConQuest the most robust and thorough coverage of English available. It is the completeness of coverage that drives performance gains in recall and precision.

Since ConQuest is a commercially available product, many additional components, not required for TREC-2, are also available, such as true client/server, graphical user interfaces, routing and dissemination, and sophisticated application program interfaces.

Query

Generally speaking, ConQuest attempts to refine and enhance the user's query. The result is then matched against the indexes to look for documents which contain similar concepts.

Queries are not "understood" in the traditional sense of natural language processing. ConQuest makes no attempt to deeply understand the objects in the query, their interaction, or the user's intent. Rather, ConQuest attempts to understand the meaning of each individual word and the importance of the word. It then uses the set of meanings and their related terms (retrieved from the semantic networks) as a statistical set which is matched against document information stored in the indexes.

* For additional information on ConQuest, please contact the author.

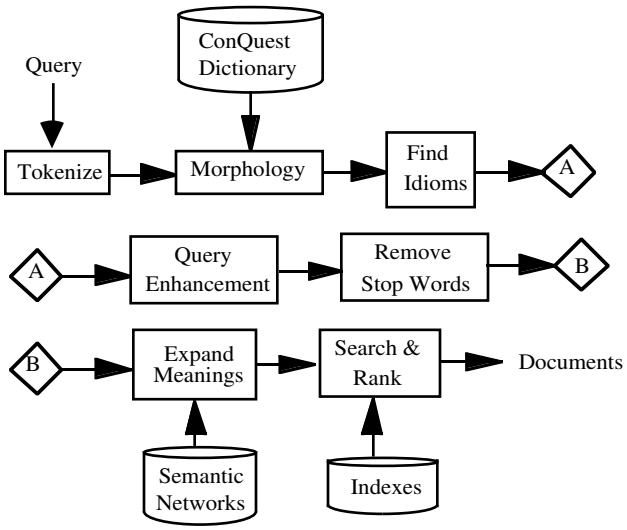

Figure 1 The Query Process

The following is a description of the modules used for query:

- *Tokenize:* Divides a string of characters into words.
- *Morphology:* An advanced form of stemming; attempts to remove suffixes and perform spelling changes to reduce words to simpler forms which are found in the dictionary. For example, one morphology rule will take "babies," strip the "ies," add "y," and produce "baby," which is found in the dictionary.
- *Find Idioms:* This module finds idioms in the text and indexes the idiom as a single unit. This prevents idioms such as "Dow Jones Industrial Average" from getting confused with queries on "industrial history." Words inside of idioms can still be located individually, if desired.
- *Query Enhancement:* The user is given the opportunity to enhance the query for additional improvement in precision and recall. There are many options available here, but the two most important are to choose meanings and weight query terms.

  Choosing a meaning of a word will restrict the expansion of words to only related terms which are relevant to the chosen meanings. This reduces noise in the query. When running in automatic mode, ConQuest expands all meanings of all words.

  Weighting query terms identifies the importance of the various words in the query. These weights are used by the search engine when ranking documents and computing document relevance factors.
- *Remove Stop Words:* Small function words—such as determiners, conjunctions, auxiliary verbs, and small adverbs—are removed from the query.
- *Expand Meanings:* Words in the query are expanded to include related terms.
- *Search and Rank:* ConQuest uses an integrated search and rank algorithm (described in the next section) which considers the relevance rankings of documents throughout the search process. Since ranking and search are integrated, the search engine automatically produces the most relevant documents right away.

Queries can be expanded to a very large number of terms, if desired. If the user wishes for the greatest amount of recall, a 5 word query can be expanded to 200 or 300 related terms.

Many other query features are also available in ConQuest, including wildcards, fuzzy spelling expansion, numeric and date range searching, boolean, mixed boolean and statistical, fielded searching (a variety of types), and searching over document categories.

Ranking Factors

Ranking and retrieval with ConQuest uses a variety of statistics and criteria, which are flexible and can be modified to handle varying requirements. The following are some of the factors used in ranking:

Completeness: A good document should contain at least one term or related term for each word in the original query.

Contextual Evidence: Words are supported by their related terms. If a document contains a word and its related terms, then the word is given a higher weight because it is surrounded by supporting evidence.

Semantic Distance: The semantic network contains information on how closely two terms are related.

Proximity: A document is considered to be more relevant if it contains matching terms which occur close together, preferably in the same paragraph or sentence.

Quantity: The absolute quantity of hits in the document is also included, but is not as strong a discriminator of relevance as the other factors.

ConQuest is the first truly "concept-based" search system to operate over unrestricted domains. If a document contains the word and some of its related terms, the word is more likely to be used in the correct context, using the "contextual evidence" factor above. In this way, ConQuest can determine word meanings at query time.

Coarse and Fine Grain Ranking

To further improve retrieval speed, ConQuest performs the search in two phases. The first is "coarse-grain." This phase is integrated with the document search process. Documents are output from the ConQuest search engine in descending coarse-grain rank order.

To compute the coarse-grain rank for a document, the statistics for the words contained in the document are combined using the coarse-grain ranking function. The inputs to this function include the semantic network

strength of each word, frequency in query, expansion terms, inverse document frequency, and query structure.

Once a document is found using coarse-grain rank, a second phase of relevancy ranking is applied, called "fine-grain" rank. This second phase uses a different ranking function which has access to more local information within the document. The inputs to this function include all of the inputs used in coarse-grain ranking, plus word location, proximity, frequency in document, and document structure.

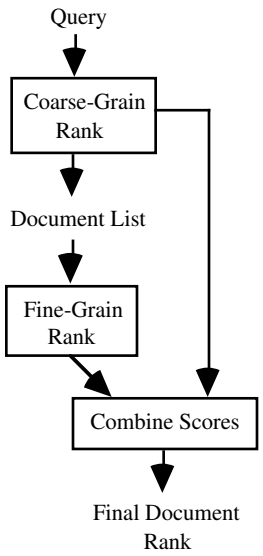

Figure 2 Fine and Coarse Grain Ranking

In general, the coarse-grain rank of a document represents global information on the document. It is a score that applies more to the document as a whole. The coarse-grain rank will be high for a document if it contains a large number of query words and related terms, ignoring the position of those terms in the document.

The fine-grain rank, on the other hand, represents local information, because the proximity (physical closeness) of the terms is the strongest contributor. The fine-grain rank of a document will be high if there is a single strong reference contained in the document.

As shown in Figure 2 above, the final document score is computed as a combination of the coarse- and fine-grain scores.

Pre-TREC Experiments

In preparation for TREC-2, ConQuest performed numerous experiments to improve the coarse-grain ranking algorithms and data. These experiments included the following:

1. Statistical word studies (statistical regressions to predict the probability that a document containing a word is relevant)
2. Statistical word-pair studies
3. Various weighting formulae
4. Various query structuring techniques

These studies were all performed under the assumption that the coarse-grain ranking formula used for TREC was weaker than the fine-grain ranking formula. The concern was that coarse-grain ranking did not retrieve a large enough percentage of relevant documents in the initial retrieval set. It was thought that once these documents were retrieved, the fine-grain algorithms would effectively use proximity and term frequency information to sort the documents and put all of the truly relevant ones at the top of the list.

Unlike other systems, ConQuest did not have funding for these TREC studies. This put the TREC studies in direct conflict with other more pressing concerns, such as supporting customers, or providing new functionality such as client/server.

As a result, the testing from these early studies proved ambiguous and unreliable. We believe that this was due to the following:

- Since time and resources were limited, tests were performed on only a small number of queries (5-10). This did not provide a large enough sample set of queries to produce reliable test results.
- ConQuest never tested the original assumption that coarse-grain was the limiting step in improving accuracy.
- The queries for this testing were taken from the TREC-1 final test queries. However, many of these queries were hastily constructed and thus added noise to the test results.

Just before the TREC-2 results were due, ConQuest decided to concentrate most of its effort on improving the tools used to generate queries. The tools and processes created are described in the next section.

Generating Queries for TREC-2

Generating queries was primarily an automatic process, based on the initial TREC-2 topic descriptions. Manual input was used primarily to remove things: Words, word meanings, and expansions. This produced queries with only the terms that are relevant. If needed, a user can also set weights for query terms.

Note that all manual steps were performed for all queries before any documents were retrieved. In other words, no feedback information was used in generating the queries. This makes ConQuest fully compliant with the rules for ad-hoc queries in TREC-2.

Automatic Query Generation Steps

A special program was created to convert TREC-2 topic descriptions into ConQuest query log files. The architecture of this program is show in Figure 3.

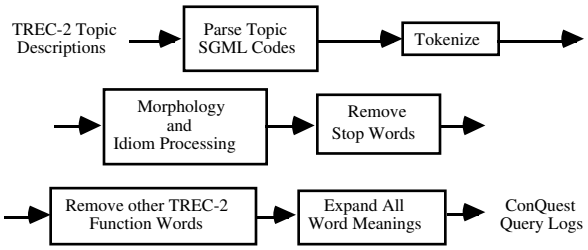

Figure 3 Program to Automatically Generate Query Log Files

The modules in the program are as follows:

Parse Topic - Reads through the topic looking for the SGML codes (such as <description>). The location within the topic for all words in the query are preserved in the final query log files.

Tokenize - Divides up strings into tokens.

Morphology - Locates all words in the dictionary and reduces them to root words if possible.

Idiom Processing - Collects idioms together as single terms, such as "United States."

Remove Stop Words - Removes conjunctions, determiners, auxiliary verbs, prepositions, etc.

Remove Function Words - Removes words such as "document," "relevant," and "retrieve" which are used often in TREC-2 narratives but do not help retrieval.

Expand Word Meanings - All word meanings are expanded using the ConQuest semantic network and all expansions are added to the query.

Note that all of these steps occur automatically with no manual input. The program also generates other statistics, such as the count of each term in the query, a count for each term for each section of the query (sections being the topic, description, narrative, concepts, and factors), and the total number of words in the query.

Manual Query Generation Steps

There were two manual steps used to generate queries:

1. Remove words, word meanings, and/or expansions
2. Set term weights (if necessary)

Fortunately, ConQuest has graphical user interfaces (GUIs) for removing words, word meanings, and expansions from the queries automatically generated. A user merely brings up the query and uses the mouse to select items to be deleted.

In TREC-2, terms were not weighted in the traditional sense, but rather were categorized into three sets:

1. Terms that embody the entire query, which would make good search terms if used by themselves
2. Terms which embody a necessary portion of the query, but not the entire concept
3. All other related terms

These categories provide simple guidelines for setting term weights, which make it much easier to generate queries. Evaluations using the TREC-2 test topics determined the functions for the actual term weights.

To emphasize once more, no document feedback was used for these manual steps. All query adjustments were performed without executing any query. Only after all queries were generated were the final results generated.

The TREC-2 Results

ConQuest scored very well in TREC-2. In particular, our recall percentages were quite high. Our average precision scores were not as good, but still competitive.

ConQuest submitted two sets of results for TREC-2, CnQst1 and CnQst2. Both sets used the same coarse-grain algorithm which retrieved the best 5000 documents from the database. The difference between the two results was how these 5000 documents were sorted to derive the top 1000 documents which were used for the official results.

The first set (CnQst1) used fine-grain as the only sorting algorithm. This algorithm primarily depends on local proximity information, although word statistics and query structure are also incorporated.

The second set of results (CnQst2) was a weighted average of the fine-grain and coarse-grain statistics for each document. As it turned out, this combination of local (fine-grain) and global (coarse-grain) statistics provided significantly better statistics.

The relatively modest addition of global information improved the results more than expected. Previous experience had always indicated that fine-grain information, especially the proximity test, was the strongest contributor to document relevancy.

Some additional insights can be extracted from topic analyses presented at the TREC-2 conference. Specifically, the topics where ConQuest excelled over other systems were also those which tended to have fewer relevant documents in the database. This indicates that local proximity statistics (used by ConQuest) are more important for these queries, since most other systems in TREC-2 are heavily weighted towards global document statistics. In other words, ConQuest appears to perform better for queries where one needs to find the "needle in the haystack."

Post TREC Analysis

After TREC-2, we had the chance to clean up our initial tests, gather new statistics, and perform some additional analysis.

The first step in this process was to prove the accuracy of the coarse-grain algorithm. Remember that initial tests attempted to improve the coarse-grain algorithm. But did the coarse-grain algorithm really need improvement? One indication that coarse-grain was accurate was provided by the CnQst2 run, which performed better than expected.

To check out the coarse-grain rank, we constructed graphs which more clearly shows its performance. Since fine-grain can only work on the results of the coarse-grain algorithm, what is the loss in recall for coarse-grain?

The following graph shows the cumulative recall percentage as documents are retrieved from coarse-grain rank. Every time a relevant document is retrieved, the recall percentage gradually inches up towards 100%. Note: these tests were run on just the Category B data.

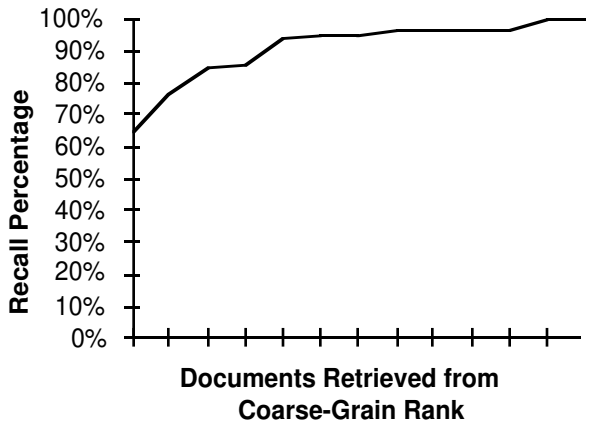

Figure 4 Cumulative Recall Percentage for Query #110

Figure 4 shows two exciting discoveries. The first is that the coarse-grain performance achieves over 95% recall. This strongly contradicts our initial fears that coarse-grain was not retrieving enough relevant documents.

The second discovery is that the high recall figures are achieved quickly. This implies that ConQuest can retrieve fewer documents (greatly improving speed) and still achieve high accuracy.

To further establish these claims, we repeated the analysis on all queries in the TREC-2 topic set, then averaged the results together, as shown in the next graph:

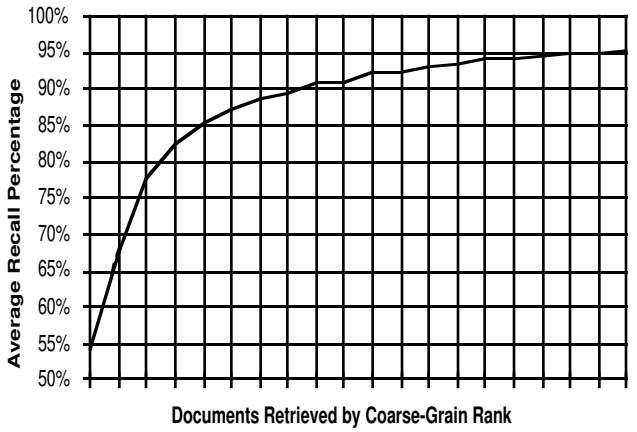

Figure 5 Cumulative Recall as Documents are Retrieved using Coarse-Grain Rank

This figure is an average over all queries. The average strongly correlates with the results from query #110. This verifies the two discoveries identified above.

Some initial studies also more clearly show the difference between fine-grain and coarse-grain sorting of documents. The following figure shows both graphs superimposed:

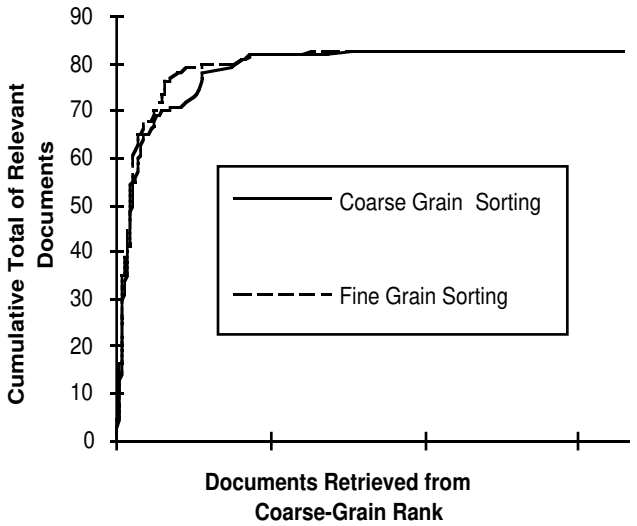

Figure 6 Coarse-Grain Sorting vs. Fine-Grain Sorting for TREC-2 Topic #135

In this diagram, we see that fine-grain sorting is in fact better than coarse-grain. In other queries, the results are more mixed. Clearly, the difference is not as great as was initially assumed.

This suggests that the area where ConQuest can most improve is not in the coarse-grain ranking algorithm, but rather in improving the fine-grain algorithm, or providing a better combination of the two.

Upon further study, we believe we now know why. When the fine-grain algorithm was developed, the programmers assumed an average query length of about 5 words. Studies of typical users indicate that their preferred query type is a

simple two or three word phrase. Therefore, a fine-grain rank tuned for short queries has provided the best and most accurate system for our commercial users.

In TREC-2, however, the queries are often 40-50 words long. These analyses have shown us that our initial fine-grain algorithms are not as accurate for such long queries.

Future Analysis

Our tests indicate that more study of other fine-grain algorithms is where ConQuest can most likely improve its scores for TREC-3. New ways of looking at proximity and positional information in the document will be explored and compared against the existing coarse-grain ranking results.

We still feel that our existing fine-grain algorithm is best for the typical commercial user, and we are looking for ways to fully test this hypothesis.

Finally, we are now much more sensitive to the effects of query size on fine-grain algorithms and are looking more closely at ways to desensitize our fine-grain algorithms, or to adapt them easily to different query lengths.