Chapter 7
Queries: Languages & Properties

Query Languages
Query Properties
This chapter covers the main aspects of queries, including

- the different languages used to express them
- their distribution and approaches for analyzing them with focus on the Web
Query Languages
We cover now the different kind of queries normally posed to text retrieval systems.

This is in part dependent on the retrieval model the system adopts.

That is, a full-text system will not answer the same kind of queries as those answered by a system based on keyword ranking.
Query Languages

There is a difference between information retrieval and data retrieval.

Languages for information retrieval allow the answer to be ranked.

For query languages not aimed at information retrieval, the concept of ranking cannot be easily defined:
- We consider these languages as languages for data retrieval.
- Some query languages are not intended for final users.
Query Languages

- There are a number of techniques to enhance the usefulness of the queries.

- Some examples are the expansion of a word to the set of its synonyms or the use of a thesaurus.

- Some words which are very frequent and do not carry meaning (called *stopwords*) may be removed.

- We refer to words that can be used to match query terms as *keywords*.
Query Languages

- Another issue is the subject of the *retrieval unit* the information retrieval system adopts.
- The retrieval unit is the basic element which can be retrieved as an answer to a query.
- We call the retrieval units simply *documents*, even if this reference can be used with different meanings.
Query Languages

Keyword Based Querying
Keyword Based Querying

- A query is the formulation of a user information need.
- Keyword based queries are popular, since they are intuitive, easy to express, and allow for fast ranking.
- However, a query can also be a more complex combination of operations involving several words.
Word Queries

The most elementary query that can be formulated in a text retrieval system is the word.

Some models are also able to see the internal division of words into letters.

- In this case, the alphabet is split into letters and separators.
- A word is a sequence of letters surrounded by separators.

The division of the text into words is not arbitrary, since words carry a lot of meaning in natural language.
Word Queries

The result of word queries is the set of documents containing at least one of the words of the query.

Further, the resulting documents are ranked according to the degree of similarity with respect to the query.

To support ranking, two common statistics on word occurrences inside texts are commonly used.

The first is called **term frequency** and counts the number of times a word appears inside a document.

The second is called **inverse document frequency** and counts the number of documents in which a word appears.
The other possibility of interpreting queries, popularized by Web search engines, is the **conjunctive form**

In this case, a document matches a query only if it contains all the words in the query.

This is useful when the number of results for one single word is too large.

Additionally, it may be required that the exact positions in which a word occurs in the text should be provided.

This might be useful for highlighting word occurrences in **snippets**, for instance, during the display of results.
Context Queries

Many systems complement queries with the ability to search words in a given context.

Words which appear near each other may signal higher likelihood of relevance than if they appear apart.

We may want to form phrases of words or find words which are proximal in the text.

**Phrase**

- Is a sequence of single-word queries
- An occurrence of the phrase is a sequence of words
- Can be ranked in a fashion somewhat analogous to single words

**Proximity**

- Is a more relaxed version of the phrase query
- A maximum allowed distance between single words or phrases is given
- The ranking technique can be depend on physical proximity
Boolean Queries

The oldest way to combine keyword queries is to use boolean operators.

A boolean query has a syntax composed of:

- atoms: basic queries that retrieve documents
- boolean operators: work on their operands (which are sets of documents) and deliver sets of documents

This scheme is in general compositional: operators can be composed over the results of other operators.
A **query syntax** tree is naturally defined

Consider the example of a query syntax tree below

```
AND
  
  translation

OR

  syntax

  syntactic
```

It will retrieve all the documents which contain the word **translation** as well as either the word **syntax** or the word **syntactic**.
The operators most commonly used, given two basic queries or boolean sub-expressions $e_1$ and $e_2$, are:

- $e_1 \text{ OR } e_2$: the query selects all documents which satisfy $e_1$ or $e_2$
- $e_1 \text{ AND } e_2$: selects all documents which satisfy both $e_1$ and $e_2$
- $e_1 \text{ BUT } e_2$: selects all documents which satisfy $e_1$ but not $e_2$
- $\text{NOT } e_2$: the query selects all documents which not contain $e_2$
Boolean Queries

- With classic boolean systems, no ranking of the retrieved documents is provided
  - A document either satisfies the boolean query or it does not
- This is quite a limitation because it does not allow for partial matching between a document and a user query
- To overcome this limitation, the condition for retrieval must be relaxed
  - For instance, a document which partially satisfies an AND condition might be retrieved
- The **NOT** operator is usually not used alone as the complement of a set of documents is the rest of the document collection
Boolean Queries

- A **fuzzy-boolean** set of operators has been proposed
- The idea is that the meaning of $AND$ and $OR$ can be relaxed, so that they retrieve more documents
- The documents are ranked higher when they have a larger number of elements in common with the query
Query Languages
Beyond Keywords
Pattern Matching

- A **pattern** is a set of syntactic features that must be found in a text segment.
- Those segments satisfying the pattern specifications are said to **match** the pattern.
- We can search for documents containing segments which match a given search pattern.
- Each system allows specifying some types of patterns.
- The more powerful the set of patterns allowed, the more involved queries can the user formulate, in general.
Pattern Matching

The most used types of patterns are:

- **Words**: a string which must be a word in the text
- **Prefixes**: a string which must form the beginning of a text word
- **Suffixes**: a string which must form the termination of a text word
- **Substrings**: a string which can appear within a text word
- **Ranges**: a pair of strings which matches any word which lexicographically lies between them
- **Allowing errors**: a word together with an error threshold
- **Regular expressions**: a rather general pattern built up by simple strings
- **Extended patterns**: a more user-friendly query language to represent some common cases of regular expressions
Pushing the fuzzy model even further, the distinction between \( AND \) and \( OR \) could be completely blurred.

In this case, a query becomes simply an enumeration of words and context queries.

The negation can be handled by letting the user express that some words are not desired.

Then the documents containing them are penalized in the ranking computation.

Under this scheme we have completely eliminated any reference to boolean operations and entered into the field of **natural language** queries.
Query Languages

Structural Queries
The text collections tend to have some structure built into them

- The standardization of languages to represent structured texts has pushed forward in this direction

Mixing contents and structure in queries allows posing very powerful queries

Queries can be expressed using containment, proximity or other restrictions on the structural elements

More details in Chapter 13 on Structured Text Retrieval
The three main types of structures:

a) form-like fixed structure
b) hypertext structure
c) hierarchical structure
Fixed Structure

- The structure allowed in texts was traditionally quite restrictive
- The documents had a fixed set of fields, and each field had some text inside
  - Some fields were not present in all documents
  - Some documents could have text not classified under any field
  - They were not allowed to nest or overlap
- Retrieval activity allowed: specifying that a given basic pattern was to be found only in a given field
Fixed Structure

- When the structure is very rigid, the content of some fields can be interpreted as numbers, dates, etc.
- This idea leads naturally to the relational model, each field corresponding to a column in the database table.
- There are several proposals that extend SQL to allow full-text retrieval.
  - Among them we can mention proposals by the leading relational database vendors such as Oracle and Sybase, as well as SFQL.
Fixed Structure

- Hypertexts probably represent the opposite trend with respect to structuring power
- Retrieval from hypertext began as a merely navigational activity
- That is, the user had to manually traverse the hypertext nodes following links to search what he/she wanted
- Some query tools allow querying hypertext based on their content and their structure
An intermediate model which lies between fixed structure and hypertext is the **hierarchical structure**.

An example of a hierarchical structure: the page of a book and its schematic view.

Chapter 6

We cover in this chapter the different kind of...

6.1 Keyword Based...

6.3 Structured Queries

We cover... Keyword Based... Structural...
An example of a query to the hierarchical structure presented

```
IN
  figure
WITH
  section
WITH
  title
  "structural"
```

This parsed query returns the image below
Query Languages
Query Protocols
Query Protocols

 Sometimes, query languages are used by applications to query text databases.

 Because they are not intended for human use, we refer to them as protocols rather than languages.

 The most important are/were:

  - Z39.50
  - Wide Area Information Service (WAIS)
Query Protocols

In the CD-ROM publishing arena there are several proposals for query protocols.

The main goal of these protocols is to provide disc interchangeability.

We can cite three of them:

- **Common Command Language (CCL)**
- **Compact Disk Read only Data exchange (CD-RDx)**
- **Structured Full-text Query Language (SFQL)**

SFQL is based on SQL and also has a client-server architecture.

The language does not define any specific formatting or markup.
Query Protocols

For example, a query in SFQL is:

```
Select abstract from journal.papers
where title contains "text search"
```

The language supports boolean and logical operators, thesaurus, proximity operations and some special characters as wild-cards and repetition.

For example:

```
... where paper contains "retrieval"
or like "info %" and date > 1/1/98
```
Query Properties
Characterizing Web Queries

- The notion of **information seeking** encompasses a broad range of **information needs**
- People have different search needs at different times and in different contexts
- A number of researchers have attempted to classify and tally the types of information needs
Most web search engines record information about the queries

This information includes the query itself, the time, and the IP address.

Some systems also record which search results were clicked on for a given query.

These logs are a valuable resource for understanding the kinds of information needs that users have.

The first studies concentrated on basic statistics:

- query occurrences
- term occurrences
- query length
Characterizing Web Queries

- Jansen & Spink claim a trend of a decreasing percentage of one-word queries.

- Jansen et al. found that the percentage of three word queries increased from 28% in 1998 to 49% in 2002.

- In May 2005, Jansen et al. conducted a study using 1.5M queries gathered from the Dogpile search engine. This study found that the mean length of the queries was 2.8 terms, with the longest query having 25 terms.

- Results for a larger data set, 185M queries of Yahoo! UK in 2007, were presented by Skobeltsyn et al.
Table below shows the distribution of query lengths for these two last references

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>18</td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td>32</td>
<td>35</td>
</tr>
<tr>
<td>3</td>
<td>25</td>
<td>23</td>
</tr>
<tr>
<td>4</td>
<td>13</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>&gt;7</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
Characterizing Web Queries

- Queries, as words in a text, follow a biased distribution.
- In fact, the frequency of query words follow a **Zipf’s law** with parameter $\alpha$.
  - The value of $\alpha$ ranges from 0.6 to 1.4, perhaps due to language and cultural differences.
- However, this is less biased than Web text, where $\alpha$ is closer to 2.
- The standard correlation among the frequency of a word in the Web pages and in the queries also varies.
  - These values range from 0.15 to 0.42.
Characterizing Web Queries

This implies that what people search is different from what people publish in the Web

Relative **query frequency** vs. relative **document frequency** of each word in a vocabulary:

![Graph showing query frequency vs. document frequency]
Characterizing Web Queries

Some logs also register the **number of answer pages seen** and the **pages selected** after a search.

Many people refine the query adding and removing words, but most of them see very few answer pages.

The table below shows query statistics for four different search engines.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Words per query</td>
<td>2.4</td>
<td>2.6</td>
<td>2.3</td>
<td>1.1</td>
</tr>
<tr>
<td>Queries per user</td>
<td>2.0</td>
<td>2.3</td>
<td>2.9</td>
<td>–</td>
</tr>
<tr>
<td>Answer pages per query</td>
<td>1.3</td>
<td>1.7</td>
<td>2.2</td>
<td>1.2</td>
</tr>
<tr>
<td>Boolean queries</td>
<td>&lt;40%</td>
<td>10%</td>
<td>–</td>
<td>16%</td>
</tr>
</tbody>
</table>
In addition, the average number of pages clicked per answer can be low due to navigational queries (around 2 clicks per query).

Further studies have shown that the focus of the queries has shifted from leisure to e-commerce (more details later in the section about Query Topics).
Query Properties
User Search Behavior
User Search Behavior

State diagram of user behavior in a small search engine
Query Intent

Until the Web, the concept of a user query was associated with searching for information of interest.

The design of the search tool was always targeted to help the user write good queries.

This implies that the query language adopted was usually complex.

The Web changed this drastically, as users started to use search engines not only to find information, but also to achieve other goals.
Query Intent

Broder’s taxonomy of Web search goals:

- Navigational: The immediate intent is to reach a particular site (24.5% survey, 20% query log)

- Informational: The intent is to acquire some information (39% survey, 48% query log)

- Transactional: The intent is to perform some web-mediated activity (36% survey, 30% query log)

This taxonomy has been heavily influential in discussions of query types on the Web.
Query Intent

*Rose & Levinson developed a taxonomy that extended and changed somewhat from Broder’s*

*They noted that much of what happens on the Web is the acquisition and consumption of online resources*

*Thus they replace Broder’s transactions category with a broader category of resources*
Query Intent

Recent research have dealt with the automatic prediction of these classes.

These works use machine learning over different query attributes such as:

- Anchor-text distribution of the words in the queries
- Past click behavior

The main problem is that many queries are inherently ambiguous.

They can be classified in more than one class when the context of the search is not known.
Queries can also be classified according to the topic of the query, independently of the query intent.

For example, a search involving the topic of weather can consist of:

- the simple information need of looking at today’s forecast, or
- the rich and complex information need of studying meteorology.
Spink & Jansen et al have manually analyzed samples of query logs.

The authors found that queries relating to sex and pornography declined from 16.8% in 1997 to just 3.6% in 2005.

On the other hand, commerce-related queries increased from 13% to almost 25% in the same years.
## Query Topic

### Changes in Excite topics from 1997 to 2001 (%)

<table>
<thead>
<tr>
<th>Rank</th>
<th>Topic</th>
<th>1997</th>
<th>2001</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Commerce, travel, employment, or economy</td>
<td>13.3</td>
<td>24.7</td>
</tr>
<tr>
<td>2</td>
<td>People, places, or things</td>
<td>6.7</td>
<td>19.7</td>
</tr>
<tr>
<td>3</td>
<td>Non-English or unknown</td>
<td>4.1</td>
<td>11.3</td>
</tr>
<tr>
<td>4</td>
<td>Computers or Internet</td>
<td>12.5</td>
<td>9.6</td>
</tr>
<tr>
<td>5</td>
<td>Sex or pornography</td>
<td>16.8</td>
<td>8.5</td>
</tr>
<tr>
<td>6</td>
<td>Health or sciences</td>
<td>9.5</td>
<td>7.5</td>
</tr>
<tr>
<td>7</td>
<td>Entertainment or recreation</td>
<td>19.9</td>
<td>6.6</td>
</tr>
<tr>
<td>8</td>
<td>Education or humanities</td>
<td>5.6</td>
<td>4.5</td>
</tr>
<tr>
<td>9</td>
<td>Society, culture, ethnicity, or religion</td>
<td>5.4</td>
<td>3.9</td>
</tr>
<tr>
<td>10</td>
<td>Government</td>
<td>3.4</td>
<td>2.0</td>
</tr>
<tr>
<td>11</td>
<td>Performing or fine arts</td>
<td>5.4</td>
<td>1.1</td>
</tr>
</tbody>
</table>
Shen et al. mapped the results of a search engine to the set of categories in the Open Directory Project. Their results: F-score of about .45 on 63 categories.

The table below shows the results for five queries:

<table>
<thead>
<tr>
<th>Query</th>
<th>Top category</th>
<th>Second category</th>
</tr>
</thead>
<tbody>
<tr>
<td>chat rooms</td>
<td>Computers/Internet</td>
<td>Online Community/Chat</td>
</tr>
<tr>
<td>lake michigan lodges</td>
<td>Info/Local &amp; Regional</td>
<td>Living/Travel &amp; Vacation</td>
</tr>
<tr>
<td>stephen hawking</td>
<td>Info/Science &amp; Tech</td>
<td>Info/Arts &amp; Humanities</td>
</tr>
<tr>
<td>dog shampoo</td>
<td>Shopping/Buying Guides</td>
<td>Living/Pets &amp; Animals</td>
</tr>
<tr>
<td>text mining</td>
<td>Computers/Software</td>
<td>Information/Companies</td>
</tr>
</tbody>
</table>

Queries: Languages & Properties, Modern Information Retrieval, Addison Wesley, 2010 – p. 53
Broder et al presented a method for classifying short, rare queries into a taxonomy of 6,000 categories.

This is an important problem because rare or infrequent queries are approximately half of all queries.

Training data: a commercial taxonomy containing many documents assigned to each category.

They classified the results of a query in the classifier, and then used a voting algorithm to classify the query.
Ambiguous queries are those queries that can have two or more distinct meanings.

For instance, a query can be related to politics but also to news.

There are few papers for ambiguity detection.

Song et al. estimated that about 16% of all queries are ambiguous.
Query Sessions and Missions

One important problem when analyzing queries is determining user query sessions.

Early work used fixed limits of time to define sessions, but this definition presents two problems:

- the session might be longer
- the goals of the user in the session might be more than one

Hence, it is good to distinguish:

- time based sessions: queries of a user in a same session
- missions: sequence of queries with the same goal

Research missions: an additional problem is that missions can span more than one session.
Query Session Boundaries

One alternative to determine sessions more accurately is to establish a **maximum inactivity time**

Different authors have found different thresholds ranging from five to sixty minutes

**Query missions** are a sequence of reformulated queries that express a same need

The detection of missions is a hard problem and has been approached by several researchers
Query Properties

Query Difficulty
Another important characteristic of queries is their intrinsic difficulty

For example, one word queries are simpler than phrase queries

There are two different ways to measure difficulty

Post-retrieval predictions mechanisms: run the query and analyze its answer set

Pre-retrieval mechanisms: evaluate the difficulty without executing the query
Post-retrieval Algorithms

- Post-retrieval algorithms are more diverse as they have more information at hand.
- **Clarity Score** relies on the difference between the language models of the collection and of the top retrieved documents.
- The intuition is that the top ranked results of an unambiguous query will be topically cohesive.
- Term distribution of an ambiguous query is assumed to be more similar to the collection distribution.
Yom-Tov et al compared the ranked list of the original query with the ranked lists of the query’s terms

Intuition: for well performing queries, the results do not change considerably if only a subset of query terms is used

Aslam et al: a query is considered to be difficult if different ranking functions retrieve diverse ranked lists

If the overlap between the top ranked documents is large across all ranked lists, the query is deemed to be easy
Zhou and Croft investigated two approaches: Weighted Information Gain and Query Feedback

**Weighted Information Gain** measures the change in information about the quality of retrieval between:

- an imaginary state that only an average document is retrieved (estimated by the collection model), and
- a posterior state where the actual search results are observed.
Post-retrieval Algorithms

**Query Feedback** frames query prediction as a communication channel problem.

- The input is query $Q$, the channel is the retrieval system, and the ranked list $L$ is the noisy output of the channel.
- From $L$, a new query $Q'$ is generated, a second ranking $L'$ is retrieved with $Q'$.
- The overlap between $L$ and $L'$ is used as prediction score.

*Hauff et al.* provides an evaluation of these techniques and propose a new technique named **Improved Clarity**.
Pre-retrieval Algorithms

Pre-retrieval algorithms must rely on the collection statistics of the query terms to predict query difficulty. For example, they either take into account:

- the frequencies of the query terms in the collection, such as **Averaged IDF** or **Simplified Clarity Score**, or
- the co-occurrence of query terms in the collection, such as **Averaged Pointwise Mutual Information** (PMI).

*Kwok et al.* proposed measures based on the inverse document frequency and the collection frequency.

Queries with low frequency terms are predicted to achieve a better performance.
He et al evaluated a number of algorithms, including Query Scope and Simplified Clarity Score

**Query Scope** uses the number of documents in the collection that contain at least one of the query terms

**Simplified Clarity Score (SCS)** relies on term frequencies:

$$SCS(q) = \sum_{k_i \in q} P_{ml}(k_i | q) \times \log_2 \left( \frac{P_{ml}(k_i | q)}{P_{coll}(k_i)} \right)$$

where

- $P_{ml}(k_i | q)$ is the maximum likelihood estimator of term $k_i$ given query $q$
- $P_{coll}(k_i)$ is the term count of $k_i$ in the collection divided by the total number of terms in the collection
**Averaged PMI** measures the **average mutual information** of two query terms in the collection:

\[
AvPMI(q) = \frac{1}{|(k_i, k_\ell)|} \sum_{(k_i, k_\ell) \in q} \log_2 \left( \frac{P_{coll}(k_i, k_\ell)}{P_{coll}(k_i)P_{coll}(k_\ell)} \right)
\]

where \(P_{coll}(k_i, k_\ell)\) is the probability that terms \(k_i\) and \(k_\ell\) occur in the same document.
Pre-retrieval Algorithms

Pre-retrieval predictors usually have a lower accuracy compared to post-retrieval predictors.

This is because the information available to them is much more general and much more sparse.

Nevertheless, two pre-retrieval predictors achieve performance equivalent to post-retrieval predictors.

Both approaches are computationally intensive:

- The technique proposed by He et al. requires clustering.
- The technique proposed by Zhao et al. requires determining the \( tfidf \) distribution of all documents.

However, both techniques are not efficient enough to be useful for large collections.