Information Processing and Management 45 (2009) 298–311

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

On the use of negation in Boolean IR queries

Shmuel T. Klein *

Department of Computer Science, Bar Ilan University, Ramat-Gan 52900, Israel

ARTICLE INFO

Article history:
Received 16 December 2007
Received in revised form 2 December 2008
Accepted 14 December 2008

Keywords:
Boolean queries
Negated keywords
Distance constraints
Concordance
Query processing

ABSTRACT

The negation operator, in various forms in which it appears in Information Retrieval queries, is investigated. The applications include negated terms in Boolean queries, more specifically in the presence of metrical constraints, but also negated characters used in the definition of extended keywords by means of regular expressions. Exact definitions are suggested and their usefulness is shown on several examples. Finally, some implementation issues are discussed, in particular as to the order in which the terms of long queries, with or without negated keywords, should be processed, and efficient heuristics for choosing a good order are suggested.

1. Introduction

Formulating a query in an Information Retrieval (IR) System requires an effort as to the correct choice of the query terms. Finding the right balance between terms that may be too broad and others that are overly restrictive is crucial to assure good retrieval performance, as can be measured by recall and precision. In fact, the formulation of queries is an art, and to be successful, one needs, in addition to mastering the query language syntax, also knowledge about the underlying textual database, its language and peculiarities.

It might be objected that in a time where powerful search engines can freely be used by everybody to access an ever growing pool of information on the internet, the usage of complex query languages, requiring a quite sophisticated user, will be less and less frequent. Even the already existing *advanced search* functions, provided by practically all search engines, are rarely used in practice. But this view of the potential set of people seeking some information is very much biased toward users of the internet, which enjoys growing popularity because it is cheap and easily accessible. In particular, a large part of the users access the internet only occasionally and with very simple queries (Spink, Wolfram, Jansen, & Saracevic, 2001). On the other hand, there are entire communities of users of Information Retrieval systems that are often focused on specific topics. Examples are lawyers and judges wishing to access juridical databases (such as Lexis), physicians and other health professionals interested in various collections of medical information (such as Medline), researchers in the Humanities studying classical texts in different languages (such as the ARTFL project on the *Trésor de la Langue Française*, Bookstein, Klein, & Ziff, 1992), etc. Taken as a part of the full set of search engine users on the internet, these communities might seem relatively small, but in fact they include many thousands of well educated users which are not reluctant to use more sophisticated tools than the most basic queries. Research to provide good query languages can thus be justified.

* Tel.: +972 3 531 8865; fax: +972 3 736 0498.
E-mail address: tomi@cs.biu.ac.il.

0306-4573/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2008.12.003

Indeed, correct query formulation has been a prominent subject in the Information Retrieval literature, and various languages with different application areas have been suggested and studied, see, e.g., Sormunen (2000), Cafarella and Etzioni (2005), or Koubarakis, Skiadopoulos, and Tryfonopoulos (2006) to cite a few recent ones. Studies relating to specific smaller communities can be found in Mason (2006), for Lexis, and in Yoo and Choi (2007), for Medline. The languages used for Database systems (DBS) are usually more involved, permitting a precise description of what is being looked for, but the task solved by a DBS is different from the classical IR task: the underlying text is structured, and meaning is conveyed not just by the words but also by their appearance in specific locations in well defined fields, whereas IR deals with free, unstructured text, and some information need has to be translated into queries which are generally quite fuzzy. XML files have common features with both database and IR systems, and languages have been adapted to treat XML files, ranging from the simplest that could possibly work (O'Keefe & Trotman, 2003) to IR inspired languages, as in (Fuhr & Großjohann, 2004). A recent study of the processing of metrical constraints for XML files can be found in (Klein, 2008).

The present work is a systematic study of the *negation operator* as it appears in its various forms in Information Retrieval applications. In fact, we restrict attention to the Boolean query model, as in Chang, Garcia-Molina, and Paepcke (1999), though several alternatives are available, like the classical vector space model (Salton, Wong, & Yang, 1975), the probabilistic model, and others. By dealing with the impact of negation on the formulation of Boolean queries, this paper complements the work in Widdows (2003), which considers negation in the context of the vector space model: a negative term is implemented as a vector which is orthogonal to the vectors of the positive terms.

The natural approach of most users to query formulation involves the choice of keywords that best describe their information needs. They often overlook the possibility of choosing also a *negative* set, that is, a set of keywords which should *not* appear in the vicinity of some others, thereby achieving improved precision. But the use of negation might sometimes be tricky and is not always symmetrical to the use of positive terms. To bring an example from another IR connected application, user queries are often improved by using *relevance feedback*, adding to the query typical terms that appear in documents that have been judged relevant; the negative counterpart, adding typical terms of documents that have been judged non-relevant as *negated* terms, cannot always be justified, as reported by Dunlop (1997). The meaning of negated terms in Boolean queries therefore needs precise definitions.

Negative keyword sets are, however, not the only application of negation. Distances can also be negative in proximity searches, and individual characters can be excluded when defining query terms using regular expressions. In the next section, several applications of a negation operator are investigated and exact definitions are proposed. Section 3 then deals with the adaptations to the software which are necessary to support the use of negation. In particular, new heuristics are suggested by which the time necessary to evaluate an efficient order of processing the different query terms is reduced from exponential to polynomial.

2. Using negation operators

In this section, the usefulness of negation operators to various applications in different areas of Information Retrieval is studied.

2.1. Negating keywords in a query

To enable a discussion on possible operators in a query language, one has first to define the query language syntax, which we shall do incrementally.

2.1.1. Simplest syntax

Most search engines allow simple queries, consisting just of a set of keywords, such as

$$A_1\, A_2\, \ldots\, A_m, \tag{1}$$

which should retrieve all the documents in the underlying textual database in which all the terms A_i occur at least once. Often, some kind of stemming is automatically performed on all the terms of the text during the construction of the database, as well as online on the query terms (Frakes, 1992; Goldsmith, Higgins, & Soglasnova, 2001), and if the text has not been pre-processed, the system has to replace each A_i by a set $\cup_{j=1}^{n_i} A_{ij}$, where each A_{ij} is a grammatical variant of A_i, and n_i is the number of such variants for keyword A_i; generating these variants is in fact the inverse process of stemming. For example, a typical query could be

`solve differential equation`,

seeking documents in which all these terms appear, but instead of `solve`, one could also accept an occurrence of `solving`, `solves`, `solution`, `solved`, etc., and `equation` could as well appear in plural form.

Negating one or more keywords in the query means that one is interested in prohibiting the occurrence of the negated terms in the retrieved documents. The query is thus extended to the form

$$[-/]A_1\, [-/]A_2\, \cdots\, [-/]A_m, \tag{2}$$