On applying formal grammar and languages, and deduction to information retrieval modelling

Sandor Dominich

Center for Information Retrieval, Department of Computer Science, University of Veszprem, Veszprem, Hungary, 8200 Veszprem, Egyetem u. 10., Email: dominich@dcs.vein.hu

Abstract

The paper applies formal methods (deduction, grammar) to some aspects of information retrieval. A formal definition of information retrieval is given as establishing a measure of a relation between documents and a user model. The user model consists of the query and additional information on user, which is partly stored and partly deduced based on the stored data and a general rule base. The retrieval system should then answer the user model rather than the query. Further, it is shown that the set of documents represented in a normal form is recursive, which makes it possible to design an additional validation processor in order to check the format of new documents being uploaded. It is also shown that the formal correctness of the query does not necessarily imply its positive answerability.

1 User model and deduction

1.1 Information need

Information Retrieval (*IR*) is concerned with the organisation, storage, retrieval, and evaluation of information relevant to a user's information need. The main components of *IR* are as follows: user; information need; request; query; information stored in computer(s); appropriate computer programs.

The user has an information need (i.e., wants to find out something, is looking for information on something; e.g., articles published on a certain subject, books written by an author, banks offering online banking services, travel agencies with last minute offers, etc.). The information need is formulated in a request for information, in natural language. The request is then expressed in the form of a query, in a form that is required by the computer programs (e.g., according to the syntax of a query language). These programs retrieve information in response to a query, e.g., they return database records, journal articles, WWW (World Wide Web) pages, etc.. This is the reason why, mainly in practice, *IR* can also be viewed as a system, and the term Information Retrieval System (*IRS*) is also used. If a user, say *U*, is interested in journal articles and/or authors on, e.g., 'mathematical methods and techniques used in information retrieval' then this is the user's information need; let us denote it by *IN*. The information need *IN* is re–formulated in a form accepted by the search processor (engine); it thus becomes a query, say *Q*.

1.2 Information retrieval without hidden information

Information is stored in computer databases. More generally, information is stored in entities which may be generically referred to as objects *O*, e.g., abstracts, articles, images, sounds, etc.; these are traditionally called documents. The objects should be suitably represented, in such a way that they can be subjected to appropriate algorithms and computer programs. The same holds for queries, too.

The overall aim of an *IR* system is to or try to return information which is relevant to the user, i.e., information that is useful, meaningful.

Thus *IR* may be re–formulated symbolically — or formally — as a 4–tuple yielding retrieved objects as follows:

$$IR = (U, IN, Q, O) \rightarrow R$$

1.3. Implicit (hidden) information

The information need *IN* is more than its expression as a query *Q*: *IN* comprises query *Q* plus additional information about user *U*. This additional information is specific to the user: spoken languages, fields of interest, preferred journals, specialisation, profession, most frequently used queries, etc.. The importance of additional information consists in that it is one factor in the judgment of relevance, when judging whether a retrieved object is relevant or not. For example, the same search term PROGRAM has different meanings for a computer programmer (meaning a text written in the C programming language and solving a differential equation) and for a conference organiser (meaning a structure and sequence of scientific and social events during the conference). The additional information is obvious for the user (he/she implicitly assumes it) but not for the computer. Thus we may term this additional information as being an implicit information *I* specific to the user *U*, and we may write:

$$IN = (Q, I)$$

Thus the meaning of the concept of *IR* can be re–formulated as being concerned with finding an appropriate — relevance — relationship, say $\Re$, between objects *O* and information need *IN*; symbolically:

$$IR = \Re(O, IN) = \Re(O, (Q, I))$$

1.4 Information retrieval with hidden information

In order for an *IR* system to find such a relation $\Re$ it should be made possible to take into account the implicit information *I* as well, and ideally the information which can be deduced (inferred) from *I* to obtain as complete a picture of user *U* as possible. Thus finding an appropriate relation $\Re$ would mean obtaining (deriving, inferring) those objects *O* which match the meaning of the query *Q* and satisfy the implicit information *I*. With these *IR* becomes:

$$IR = \Re(O, (Q, \langle I, |\rightarrow\rangle)$$

where $\langle I, |\rightarrow\rangle$ means *I* plus information derivable (e.g., in some language or logic) or inferred or deduced from *I*. Of course, the relation $\Re$ is established with some (un)certainty *m*; thus:

$$IR = m[\Re(O, (Q, \langle I, |\rightarrow\rangle))]$$

There is a rich literature on user modelling. Based on [4], [5], we give a small example to render a possible meaning of $\langle I, |\rightarrow\rangle$. The user's implicit information *I* may be (stored permanently, and updated as necessary). Consider, for example, the following user:

Identifier: U100
Name: UserOneHundred
Languages spoken: Hungarian,
Age: 24
Computer skills: payroll software
Profession: Secretary

A rule base to deduce additional information from *I* may be:

IF (user has retrieval experience)
THEN (user is skilled AND likes shortcuts AND familiar with Boolean expressions)

IF (user has no OR less retrieval experience)
THEN (user is a beginner AND prefers menus)

IF (user is a child)
THEN (user likes more colours and few text)

IF (user does not speak English)
THEN (do not return hits in the English language)

etc.

Note. Of course, the rules may be weighted, and these may be taken into account when they are applied.

The answering process of a query Q should, then, take into account the above information, too. Let us assume that, for example, the user, our secretary, is given the job to find flights between Budapest and New Orleans to attend the ACM SIGIR MF/IR 2001 conference. Consulting, say, a dictionary, the user enters the following query *Q* to an imaginary Web search engine WWWSearch:

$$Q = \text{Budapest New Orleans ACM SIGIR plane}$$

Then, the imaginary WWW retrieval system WWWSearch should answer the following $(Q, \langle I, |\rightarrow\rangle)$:

Budapest New Orleans ACM SIGIR plane
AND show menus in Hungarian only
AND show Hungarian hits only

Note. It is very probable that our secretary's boss would better search himself.

2 Answerability of formally correct questions

2.1 Documents as a formal language

The theory of formal grammars, languages and automata has been successfully applied to, e.g., compiler theory, linguistics, etc.. Its application to information retrieval (IR) is more recent. [1, 2]

Let $T = \{t_1, ..., t_n\}$ denote the set of index terms. Every document can be represented as a DNF (disjunctive normal forms) of terms; e.g., in the binary vector space model, Boolean model. Let L_{DNF} denote the set of DNFs corresponding to documents, i.e.,

$$L_{DNF} = \{DNF_j \mid j \text{ denotes the } j\text{th document}, j = 1,..., m\}$$

There exists [3] a context-free (type 2) grammar that generates documents represented as above. A context-free grammar consists of a finite set of terminal symbols, a finite set of non-terminal symbols, a finite set of productions, and a start symbol (non-terminal). It generates the set of all sequences of terminals that can be produced from the start symbol by subsequently replacing non-terminals by sequences of terminals and non-terminals when there is a production. Given the following formal grammar *G*:

$$G = (\{s\}; \{t, 1, \wedge, \vee, \neg\}; s; \{s \rightarrow s1 \mid t1 \mid \wedge ss \mid \vee ss \mid \neg s\})$$

The language generated by *G* is denoted by *L*(*G*), and consists of Boolean formulas. The first two productions make it possible to obtain the index terms $t1(=t_1)$, $t11(=t_2)$, $t111(=t_3)$, and so on. The other productions make the construction of disjunction, conjunction and negation of terms possible. Thus, the DNFs (or their equivalents) representing documents will be among the elements of *L*(*G*), and

$$L_{DNF} \subseteq L(G)$$

Note. Of course, in practice, some of the elements of *L*(G) may not exist in the real IR system.

2.2 Automatic formal validation of documents

As known, *G* produces a context-free language, and its acceptor is a (nondeterministic) pushdown automaton (PDA), i.e., a finite automaton with a stack, so that transitions are determined in part by the top of the stack, and may affect the stack. The PDA accepts a language if it has an empty stack or is in a final state when the string (the DNF) terminates. It is also known that a 3-tape Turing Machine (TM) can simulate a 2-stack PDA by having one

tape for the input and one for each stack. Thus, there is a TM that accepts the documents represented in DNF, and hence $L(G)$, and thus L_{DNF}, is recursively enumerable; moreover, it is recursive.

This means that it can be algorithmically decided whether a given Boolean expression of terms (of any general form) is the DNF of a (potential) document, i.e., whether the documet has the correct format. This can be used to validate document representations before being added to the database (of already existing documents).

Note. The automaton is, of course, not able to tell with certainty whether a DNF is an actual representation of a real document. But it can serve to formally assess the format of documents.

2.3 Formally correct questions and answerability

The query, too, being —formally—a document, is generated by G, and thus recognised by the automaton. Unfortunately, whether the query is recognised or not is irrelevant to the retrieval process: even if it is recognised there is no guarantee that there will be documents to answer it. In other words, a (formally) correct question does not necessarily imply effective hits in the retrieval process; the query is answered by a retrieval mechanism of some sort, which is independent of the automaton.

Example. The query Q be a conjunction of two existing terms but there is no document D containing, say, the second term:

$$Q = t_1 \wedge t_3,\ D = t_1 \wedge t_2 \wedge \neg t_4$$

In a classical Boolean retrieval mechanism no documents are returned. However, using some other retrieval model (for example, binary vector space) D may be returned with a rank attached to it (depending on the formula used).

4 Discussion, conclusions

A formal definition of information retrieval was given as establishing a measure of a relation between documents and a user model. The user model consists of the query and additional information on user, which is partly stored and partly deduced based on the stored data and a general rule base. The rules may have attached weights attached to them, which can be used when the rules are applied. The rule base can, of course, contain more rules, and also rules based on, e.g., heuristics gathered from experiments. The retrieval system should then answer the user model rather than the query. Although Web retrieval, for example, is hardly conceivable along this model, it would, however, be useful for a large category of users who do not speak English.

Further, it was shown that the set of documents represented in a disjunctive (or, equivalently, conjunctive) normal form is recursive (being generated by a context=free grammar), which makes it possible to design an additional validation processor in order to check the format of new documents being uploaded. This could be useful when updating the database containing documents. The query too, being formally a document, can be represented in a normal form. Thus its formal correctness can be automatically checked an automaton. Somewhat strangely, the formal correctness of the query does not necessarily imply its positive answerability, i.e., that there will be effective documents as answers to it. In other words, the automaton does not mean nor provide any retrieval mechanism. However, such formal techniques may prove helpful in, e.g., Web retrieval (HTML, XML).

5 Acknowledgements

Research partly supported by research grants OTKA T030194, and AKP 2001-140. Thanks go A. Nagy for helpful discussions.

References

[1] Losada, D.E., and Barreiro, A. (2000). Efficient algorithms for ranking documents. *Technology Letters*, 4(1): 16-24.

[2] Pérez Gutiérrez, M. (2000). The query language: A formal grammar for information retrieval. *Revista Española de Documentación Científica*, 23, 3, 247 - 266.

[3] Arbib, M.A. (1968). *The algebraic theory of machines, languages and semigroups.* Academic Press, New York, London.

[4] Dominich, S. (2001). *Mathematical Foundations of Information Retrieval*. Kluwer Academic Publishers, Dordrecht, Boston, London.

[5] Dominich, S. (1990). Stereotype and database selection. In Haase, V., and Zinterhof, P. (eds., 1990) *Proceedings of the Future Trends in Information Technology '90*, R. Oldenbourg, Wien, Munchen, 33-42.