STRUCTURED TEXT RETRIEVAL MODELS

Djoerd Hiemstra
University of Twente
http://www.cs.utwente.nl/~hiemstra

Ricardo Baeza-Yates
Yahoo! Research
http://www.baeza.cl/

SYNONYMS

Retrieval Models for Text Databases

DEFINITION

Structured text retrieval models provide a formal definition or mathematical framework for querying semi-structured textual databases. A textual database contains both content and structure. The content is the text itself, and the structure divides the database into separate textual parts and relates those textual parts by some criterion. Often, textual databases can be represented as marked up text, for instance as XML, where the XML elements define the structure on the text content. Retrieval models for textual databases should comprise three parts: 1) a model of the text, 2) a model of the structure, and 3) a query language [4]: The model of the text defines a tokenization into words or other semantic units, as well as stop words, stemming, synonyms, etc. The model of the structure defines parts of the text, typically a contiguous portion of the text called element, region, or segment, which is defined on top of the text model's word tokens. The query language typically defines a number of operators on content and structure such as set operators and operators like "containing" and "contained-by" to model relations between content and structure, as well as relations between the structural elements themselves. Using such a query language, the (expert) user can for instance formulate requests like *"I want a paragraph discussing formal models near to a table discussing the differences between databases and information retrieval"*. Here, "formal models" and "differences between databases and information retrieval" should match the content that needs to be retrieved from the database, whereas "paragraph" and "table" refer to structural constraints on the units to retrieve. The features, structuring power, and the expressiveness of the query languages of several models for structured text retrieval are discussed below.

HISTORICAL BACKGROUND

The STAIRS system (Storage and Information Retrieval System), which was developed at IBM already in the late 1950's allowed querying both content and structure. Much like today's On-line Public Access Catalogues, it was used to store bibliographic data in records with fields such as *keywords* and *title*, providing structured search, but no overlapping or hierarchical structures nor full text search. At the end of the 1980's, researchers at the University of Waterloo in Canada researched database support for the creation of an electronic version of the Oxford English Dictionary. This resulted in a number of models for querying and manipulating content and hierarchical structure such as the parsed strings model [10], PAT expressions [15], the containment model [5] and generalized concordance lists model [7]. Similar approaches were developed elsewhere, such as the proximal nodes model [13] and the nested region model [11]. The interest in structured text retrieval models has grown since the introduction of XML in 1998, and the emergence of standard data retrieval query languages (see XPATH/XQUERY) for XML data. One might argue that the structured text retrieval approaches such as PAT expressions and proximal nodes mentioned above are predecessors of XPath. The success of XML in turn has influenced the work on structured retrieval models: XIRQL was proposed in 2000 [9] as an information retrieval extension of XML query languages (XIRQL is an extension of XQL, a predecessor of XPath). More recently, in 2004, NEXI (see NEXI) and XQuery and XPath Full-Text (see XQUERY FULL TEXT) have been proposed as query languages for structured text retrieval, as well as examples of structured text retrieval models for the respective query languages [12] and [2].

SCIENTIFIC FUNDAMENTALS

There are several models of structured text retrieval. Since there is no consensus on how to structure a textual database, this entry addresses several modeling decisions following the taxonomy presented in [4]. The entry only addresses models for text databases, and not text retrieval models in relational databases as for instance provided by SQL/MM. Due to the success of XML, today XML retrieval is almost a synonym for structured text retrieval, although XML retrieval only addresses the explicit, single hierarchy case below.

Explicit vs. implicit structure Most models use *explicit structure*, i.e., they define unambiguously what parts of the textual database are for instance "sections". These models require the database to be structured explicitly and unambiguously, or using terminology from markup languages: the models require the database to be well-formed. This allows easy modeling of nested regions, and powerful structural relationships such as the direct ancestor relationship (i.e., child and parent axis steps in XPath). The following query might be used in an explicit structure approach to retrieve sections that contain the word "databases":

```
section CONTAINING "databases"
```

Explicit structure is assumed by amongst others the proximal nodes model [13] and the full match model [2]. In systems that use *implicit structure*, however, structure is not explicitly distinguished from content. In these approaches the database is modeled as a sequence of tokens without distinguishing a word token from a markup token. A structural element should therefore be constructed at runtime by looking up the opening markup tokens, the closing tokens, and to return those regions starting with a opening token and ending with a closing token. The query above would then be formulated as [11] (here, the operator ".." would be pronounced as "following"):

```
( "<section>" .. "</section>" ) CONTAINING "databases"
```

So, the `section` element only exists at querying time. Semantically, the query is not different from a content-only query. For instance the query `("all" .. "equal" ) CONTAINING "created"` retrieves regions that start with the word "all", that end with the word "equal" and that contain the word "created", matching for instance the phrase "all men are created equal". Nested elements, or unbalanced tags are handled differently by several approaches. In the Generalized Concordance Lists (GCL) approach [7], nested sections will not recognized by the system (instead two partially overlapped sections will be returned). In the nested region algebra approach [11] nested elements are returned properly. The approach is implemented as sgrep (structured grep).[1] The GCL approach was recently implemented in a research system called Wumpus.[2]

Static vs. dynamic structure The use of implicit structure also implies the use of *dynamic structure*. A system that uses dynamic structure allows operations that define new elements or regions, i.e., elements or regions that were not previously in the database. In XQuery, this is done by element construction, but in some approaches dynamic structure is a natural consequence of the model. As an early example of dynamic structure consider the following bibliographic entry:

> John Doe, "Crime", *Police 6*, 2028.

The entry is explicitly structured by the following grammar that functions as a database schema [10]:

```
entry    :=  author ', ' title ', ' journal ', ' year '.'  ;
author   :=  text ;
title    :=  '"' text '"' ;
journal  :=  text digit+ ;
year     :=  digit digit digit digit ;
text     :=  ( letter | ' ' ) + ;
```

A valid database instance contains data that conforms to the grammar. The instance takes the form of a parsed string or "p-string". Note that the schema does not distinguish the author's first name(s) from his surname, but this might be done at query time by introducing a small grammar fragment `NameG` that parses the author strings into given names and surnames:

[1] http://www.cs.helsinki.fi/~jjaakkol/sgrep.html
[2] http://www.wumpus-search.org

```
NameG  :=  { name       :=  ( givenname ' ' )+ surname ;
             givenname  :=  letter + ;
             surname    :=  letter + ; }
```

The p-strings model provides a simple query language for adding additional grammar fragments. Suppose the bibliographic entry above is `E`, then the following query returns a p-string containing the author element with given name and surname explicitly identified.

`(author` *in* `E)` *reparsed by* `NameG`

The construct might be used to search for all authors with surname "Doe" that wrote a journal paper that mentions "grammar" in the title. The p-strings model uses regular expression matching as a core language primitive, and as such dynamic structure is more easily added than in for instance XQuery or XQuery Full-Text.

Single hierarchy vs. multiple hierarchies Although some fielded search methods use a flat structure to model text, the approaches considered here assume a hierarchical structure of the text database. The systems that use implicit structure introduced above assume a single hierarchy. Interestingly, many approaches assume multiple structural hierarchies on the same textual database. Each hierarchy might serve a different purpose. For instance, one hierarchy might represent the logical structure of the text, dividing it in chapters, sections, subsections, etc., whereas a second hierarchy might represent the lay-out structure in columns and pages; and a third layer might represent the results of a part-of-speech tagger, etc. Inside a single hierarchy, the structural elements are either disjoint or nested inside each other, but across hierarchies elements may partially overlap, i.e., a subsection might start half way a page, and end on the following page.
Some approaches relate single views in one query [13]. An interesting approach is suggested by Alink [1], who introduces additional XPath steps (select-narrow and select-wide) that navigate from one hierarchy to another. For instance, in the following XQuery Full-Text-like query fragment navigates from the paragraph elements to another hierarchy with a `Verb` element that contains "killed", and to a hierarchy with a `person` element that contains "Abraham Lincoln":

```
$doc//paragraph[
  ./select-narrow::Verb ftcontains "killed" and
  ./select-narrow::person ftcontains "Abraham Lincoln" ]
```

The need for multiple hierarchies is for instance addressed in the containment model [5], and the proximal nodes model [13]. In several publications, the hierarchies are called "stand-off annotation" or "offset annotation" to stress that the structural information (or annotations) are modeled separately from the textual data.

Exact matching vs. ranking Many of the early structured text retrieval models do not consider ranked retrieval results, or if they do only as an afterthought, i.e., by ranking the retrieval results using a text-only query disregarding the structural conditions in the query [5]. A simple but powerful way to take the structure of the results into account is to apply a standard information retrieval model to the retrieved content, and then propagate element scores or aggregate term weights based on the text structure. In several of these approaches to ranking, propagation or aggregation is guided by weighting the paths to elements by so-called augmentation weights [9] or interpolation parameters [14], to model for instance that a title element is more likely to contain important information than a bibliography element. Instead of propagating or aggregating the scores from the leaf nodes, *algebraic* approaches include the ranking functionality inside each operator of the query language [2, 12]. Ranking might also include relaxation of the query's structural conditions, for instance by relaxing complex queries step-wise to simpler queries [3].
In 2002, Fuhr and Lalmas [8] organized the first workshop of the Initiative for the Evaluation of XML Retrieval. The goal of INEX is to evaluate the quality of the retrieved results, and as such the quality of the ranking provided by the system taking both content and structure into account. The initiative provides a large testbed, consisting of XML documents, queries and relevance judgments on the data, where the relevance judgments are human judgments that define if an XML element is relevant to the query or not. With XML databases and extensions of XML query languages becoming a de-facto standard for structured text retrieval, ranking is remaining one of the main research challenges.

KEY APPLICATIONS

Systems based on structured text retrieval models can be applied to any problem that involves semi-structured text databases. Key applications of the approaches described in this section include: Managing and searching electronic dictionaries such as the Oxford English Dictionary [10, 15], managing and searching electronic journals such as the journals of the IEEE [6, 8, 12], searching stageplays such as the collected works of William Shakespeare [7], and searching hard drives for digital forensics [1].

CROSS REFERENCE

AGGREGATION-BASED SEMI-STRUCTURED TEXT RETRIEVAL,
INFORMATION RETRIEVAL MODEL,
NEXI,
PROPAGATION-BASED SEMI-STRUCTURED TEXT RETRIEVAL,
XPATH/XQUERY,
XQUERY FULL TEXT.

RECOMMENDED READING

[1] W. Alink. XIRAF: An XML information retrieval approach to digital forensics. Master's thesis, University of Twente, 2005.

[2] S. Amer-Yahia, C. Botev, and J. Shanmugasundaram. TeXQuery: a full-text search extension to XQuery. In *Proceedings of the 13th international World Wide Web conference*, 2004.

[3] S. Amer-Yahia, L.V.S. Lakshmanan, and S. Pandit. FleXPath: flexible structure and full-text querying for XML. In *Proceedings of the 2004 ACM international conference on management of data (SIGMOD)*, 2004.

[4] R.A. Baeza-Yates and G. Navarro. Integrating contents and structure in text retrieval. *SIGMOD Record 25*(1):67–79, 1996.

[5] F.J. Burkowski. Retrieval activities in a database consisting of heterogeneous collections of structured text. In *Proceedings of the 15th ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 112–124, 1992.

[6] D. Carmel, Y.S. Maarek, M. Mandelbrod, Y. Mass, and A. Soffer. Searching XML documents via XML fragments. In *Proceedings of the 26th ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 151–158, 2003.

[7] C.L.A. Clarke, G.V. Cormack, and F.J. Burkowski. An algebra for structured text search and a framework for its implementation. *The Computer Journal 38*:43–56, 1995.

[8] N. Fuhr, N. Gövert, G. Kazai, and M. Lalmas, editors. *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, ERCIM Workshop Proceedings, 2002.

[9] N. Fuhr and K. Grossjohann. XIRQL: A query language for information retrieval in XML. In *Proceedings of the 24th ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 172–180, 2001.

[10] G.H. Gonnet and F.W. Tompa. Mind your grammar: a new approach to modelling text. In *Proceedings of 13th International Conference on Very Large Data Bases (VLDB)*, pages 339–346. 1987.

[11] J. Jaakkola and P. Kilpeläinen. Nested text-region algebra. Technical report, University of Helsinki, 1999.

[12] V. Mihajlovic, H.E. Blok, D. Hiemstra, and P.M.G. Apers. Score region algebra: Building a transparent XML-IR database. In *Proceedings of the 14th International Conference on Information and Knowledge Management (CIKM)*, pages 12–19, 2005.

[13] G. Navarro and R.A. Baeza-Yates. Proximal nodes: A model to query document databases by content and structure. *ACM Transactions on Information Systems 15*:400–435, 1997.

[14] P. Ogilvie and J. Callan. Hierarchical Language Models for XML Component Retrieval. In *Advances in XML Information Retrieval, Lecture Notes in Computer Science 3493*, *Springer*, pages 224–237, 2005.

[15] A. Salminen and F.W. Tompa. PAT expressions: an algebra for text search. In *COMPLEX 92*, pages 309–332, 1992.