

A Flexible tool for Cross-Collection Patent Search

Stefania Marrara and Gabriella Pasi

Department of Informatics, Systems and Communication (DISCo)
University of Milano-Bicocca, Building U14,
I-20126, Milano, Italy
{stefania.marrara, pasi}@disco.unimib.it

Abstract. Prior-art retrieval is a crucial application of Patent Retrieval aimed to determine the novelty of a new invention. In this scenario patent authors require an exhaustive knowledge of all related patents and the search often involves multiple patent collections across the world, which do not share the same document structure or vocabulary. For this reason, despite of the numerous patent search applications already available, we propose in this paper *PatentLight*[1] a search tool that offers novel and flexible functionalities based on both fuzzy logic and IR to help users looking for relevant patents here represented as XML documents. We show some examples of the proposed search tool to inquiry the WIPO and USPTO collections in a flexible way.

Keywords: Patent Search, Fuzzy Logic, Information Retrieval, Flexible Query Language, XML

1 Introduction

Patent Information Retrieval (PIR) is a specialized branch of Information Retrieval, which is aimed to support users, often professionals such as patent attorneys or inventors, in retrieving patents that satisfy their information needs [2].

In this scenario, a crucial application is *prior-art retrieval* [3], which is performed by patent searchers to determine the novelty of a new invention. In fact patent authors require an exhaustive knowledge of all related patents since overlooking a single important patent could lead to detrimental and very expensive consequences, such as patent infringements and litigation.

Today patents are commonly available thanks to collections such as USPTO (United States Patent and Trade Office), EPO (European Patent Office) and WIPO (World Intellectual Property Organization).

Each collection contains several thousands of patents and continues to grow up year by year; this situation poses a serious issue to patent professionals: the cost of filing patents, defining claims and defending a claim of infringement is increasing with time, making the process often too expensive, due to the

complexity in finding relevant patents. In 2010 the estimated cost to find relevant patents was \$1,500 per patent filing [6].

For the above reasons Patent Retrieval stimulates an increasing interest of the scientific community, and it is also considered a complex challenging task since the vocabulary used in patents is often obscure as it contains a lot of specialized or technical words. Often the obfuscation of content is intentional by writers who wish their patents difficult to retrieve; patents contain an intrinsic structure which often include *description*, *claims* or *prior-art* for instance and can be different in different collections. Finally typical queries in patent retrieval include a huge amount of words, often entire claims.

Most Patent Search tools available today are collection dependent. The most known, Google Patents [4] and PatentsSearcher [5,14], are centered on the USPTO collection even if the issue of world-wide patents search is perceived. In fact PatentsSearcher claims to "rely on external services to query international patents and applications" (see www.patentsearcher.com/aboutSearch.jsp), while Google Patents includes the WIPO and EPO collections restricted to US patents only.

Most approaches presented in the literature, based on *keyword extraction* or *query expansion* techniques, proved to produce poor results (see Section 1.1).

Despite this fact, we believe that a traditional keyword-based analysis of XML patents joined with our flexible search approach can be promising with respect to both recall and precision.

The first experimental results produced by [1] on the USPTO collection have motivated us to further investigate in this direction.

In this paper we present the development of our flexible tool PatentLight plus some examples obtained on more than one patent collection from different English language countries. Each document collection stores more or less the same information (i.e., abstract, author names, topic, description, etc.) but with different tree structures and tag vocabulary.

For this task our PatentLight tool [1] has been improved by the introduction of the `similar` constraint on tag names (see Section 2.2).

The approach we propose in this paper relies on the recent outcomes of research in XML Retrieval, overcoming the weaknesses of traditional keyword-based approaches in the patents domain.

1.1 Related Work

Patent IR evolved as a separate branch of IR showing characteristics that drastically reduce the effectiveness of traditional retrieval techniques.

In the last years, several approaches have been proposed, which can be broadly classified into three categories:

- approaches based on *query expansion* techniques to reduce vocabulary mismatch;
- approaches based on *query extraction* techniques to reduce verbose queries

- approaches based on *query translation* techniques, which include approaches for querying multilingual patent collections, and approaches to query patents section by section instead as a whole document.

Because of the peculiarities of patent retrieval w.r.t traditional retrieval (as described in the Introduction), standard IR techniques such as *query expansion* proved not to work effectively with patent queries due to the presence of *noisy* terms in the typical queries.

In real practice however, most patent examiners formulate their queries for invalidating claims by selecting high frequency terms from the query-patent claim text, and hence the first approaches proposed in the literature [7, 8] moved their steps from this practice and were based on keyword extraction to reduce queries dimensions, unfortunately achieving results of low quality. More recently, [9] and [10] showed that using the whole patent text with raw term frequency (i.e., simple number of term occurrences in each document) reduces the job complexity and the best results are obtained when terms are taken from all the fields of the query patent.

Other approaches, [11] and [12], used citation extraction to improve the retrieval effectiveness of keyword based IR methods, and this idea is also adopted in [13] that also applies a query expansion technique on segmented queries.

Another important work is [14] which also adopted a query expansion technique based on some *structural* properties of patents such as abstract, description and image descriptions.

The last class of patent retrieval approaches tries to take advantage from the multilinguality of most patent collections, which means that the same patent can be stored in more than one language. Most works are based on natural language processing (NLP) approaches [15–17]. In particular, the most recent [17] uses NLP, and specifically statistical word alignment to translate patent queries from language to language. More generally, query translation has been a popular research mainstream and it is usually realized by means of dictionaries, machine translation systems, ontologies or combinations of these (see [18] for an overview).

Finally a nice survey on users issues and expectations associated to Patent Retrieval is [19]. In this paper authors perform a deep analysis of patent users and their search requirements with respect to current IR systems and applications.

2 FleXy: a Flexible XML Query language

In [24] a flexible extension of the XQuery Full Text language (*FleXy*) by introducing flexible constraints on both XML document structure and content was defined.

A patent search application based on FleXy has been proposed in [1] *Patent-Light*.

In PatentLight the structure-based constraints of Flexy named **below** and **near**, and the content-based flexible constraint **around** where employed. In this

section we introduce the constraint `similar` which applies on tag names, and we show how the combination of content-based and structure-based evaluation of results can improve the effectiveness of PatentLight. Here below a short explanation of the above flexible constraints is given.

2.1 A short description of below, near and around

The constraint `below` retrieves the fragments of an XML document (in this case a patent) that are closer to the path required by the user’s query. The syntax of the `below` constraint follows the standard XQuery axis syntax, and it is specified as: `c/below::t`, where `c` is the *context* node, and `t` is the target node. The best retrieved path is the one in which `t` is direct child of `c`. Others paths, those in which `t` is simply descendant of `c`, will be retrieved but ranked in a lower position w.r.t. the best one. To create the list of results we compute a *path relevance degree* for each retrieved fragment, $w_{c,t}$, computed as $w_{c,t} = \frac{1}{|desc_arc(c,t)|}$ where $desc_arc(c,t)$ is a function that returns the set of descending arcs from `c` to `t` if and only if `t` is a descendant node of `c`.

The flexible constraint `near` retrieves elements that are connected to the *context node* by any path (not only the *descendant* relationship), i.e., also *ancestor* and *sibling* elements are evaluated. For the `near` constraint, the scoring function is defined as: $w_{c,t} = \frac{1}{|arcs(c,t)|}$ where $arcs(c,t)$ is the function that returns the set of arcs that connects the context node `c` to the target node `t` following the shortest path. The `near` constraint syntax is: `c/near::t`, where, as for the constraint `below`, `c` is the context node and `t` is the target node.

`Around` is a flexible constraint which applies to numerical data and its evaluation function is formally defined as the membership function of a fuzzy subset on the considered numerical domain; the membership function expresses the similarity between the retrieved values and the numerical value requested by the user. In the patent domain, the constraint `around` is defined to the aim of analyzing date contents.

The FleXy syntax of `around` is `'tag-date/@date[x around b]'`, where `tag-date` is the attribute having the date value that has to be evaluated, `x` is the date value of the examined patent, and `b` is the date written by the user in his/her query.

The evaluation function of the `around` constraint produces a score in the interval $[0,1]$ based on the date value `b` specified by the user and the date value `x` of the patent. Patents with a date value close to the one specified by the user will receive a higher score (score close to 1) than other patents. The evaluation function of the flexible constraint `around` on the *Date* domain can be defined as fuzzy subset with a triangular membership function centered on `b`.

2.2 The similar constraint to assess tag similarity

`Similar` is a flexible constraint defined on tag names that allows to retrieve fragments with a target node name *similar* to the name used in the user query.

Similar is defined as a function whose FleXy syntax is '**similar(x)**', where **x** is the node name we are looking for. The evaluation of the function returns a list of XML fragments with a target node name similar to **x** where the *similarity degree* is number in the interval $[0,1]$ computed as $ws = \frac{1}{1+ed}$ with **ed** = edit distance between the retrieved tag name and **x**.

Fig. 1 shows how the **similar** constraint works on two document fragments, the left one from the USPTO collection, the right one from the WIPO collection.

Although the query *Q1* is looking for a fragment containing the tag name **last-name**, the system is able to retrieve also the patent fragment containing the tag name **orgname** with a *similarity degree* of 0.16.

Note that traditional XML query languages would not retrieve the second fragment in the same situation.

Moreover, to avoid the retrieval of unuseful fragments we can set a threshold value for *ws*. At present we do not evaluate synonyms since this option would include the use of a dictionary or an ontology.

Q1: applicants//similar>Last-Name)[text() contain text «Smith»]

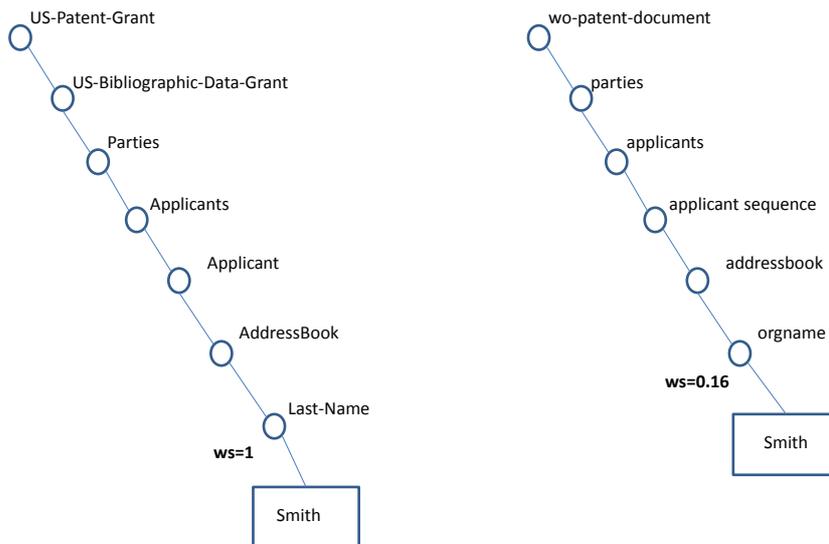


Fig. 1. The flexible constraint **similar**

When a query involves more than one flexible constraint, for instance a flexible axis and the **similar** constraint for the target node name, the overall *relevance*

degree $w_{c,t}$ is computed as a combination between the two scores, $w_{c,t}$ and ws . In principle we prefer a conservative evaluation and therefore we use $w_{c,t} = \min(w_{c,t}, ws)$ but different solutions could be tested.

3 How the flexible constraints works in patent search

This section explains how the flexible constraints introduced in the previous section are used to exploit the patent search task. In particular, the proposed approach has been defined and tested in [1] on the USPTO patent collection that can be freely downloaded on the web. USPTO is the corpus adopted by most patent search applications such as Google Patents and PatentsSearcher. Google Patents has been recently extended to include the WIPO and EPO collections but search is restricted to US patents only.

In any case it was noted that also EPO and WIPO patent documents show more or less the same structure of USPTO, even if with different tags. In this paper we use the `similar` constraint to extend our tests to a cross collections composed by USPTO and WIPO.

The proposed approach allows users to search patents using the formulation of a keyword based query, in addition the user can choose the `similar` tag option to extend the search also to tag names similar to the standard ones.

Subsection 3.1 describes how search results are categorized according to keyword-based queries, while subsection 3.2 shows how the `similar` constraint evaluation changes the original approach in [1].

3.1 Keyword-based Query: the approach

An important functionality of FlexSearch [1] is to categorize patents by exploiting their XML structure. The engine organizes the XML patents into meaningful semantic XML elements covering the main patent information. In this way the categorization process described below can easily capture what the user topical search intent is by identifying the possible interpretations associated with a patent.

By analyzing the patents in the USPTO collection, four categories were identified in [1]: *People*, *Title*, *Description*, and *Claims*.

The same categories are here adopted for the cross-collection due to the structural similarity between the WIPO and the USPTO collections.

Formally, let E be the set of XML elements defined in a patent collection, and Cat be the set of categories, then one or more elements $e_i \in E$ are mapped into each category $c \in Cat$, i.e. $\{e_1, \dots, e_m\} \rightarrow c$. In the application the four identified categories along with the corresponding XML elements are: *People* (the associated elements are `Applicants`, `Agents`, `Assignee`, `Examiners`), *Title* (`title`), *Description* (`Description`), *Claims* (`claims`).

A user specified keyword based query (here below "query terms") is automatically rewritten into four distinct FleXy queries, one for each of the four categories. The structure of each query is predefined in order to search the query terms in pre-established elements as follows:

```

People:
applicants/near::Last-Name[
text() contains text "query terms"]

```

```

Title:
invention-title[
text() contains text "query terms"]

```

```

Descriptions:
Description/below::p[
text() contains text "query terms"]

```

```

Claims:
claims/below::claim-text[
text() contains text "query terms"]

```

The proposed query translation process uses the **near** constraint in the FleXy query related to the category *People*, and the context node is the tag **applicants**; this means that we assume that the applicant role (i.e., the inventor) has more importance in the search with respect to the other roles defined in the patent such as *Agent*, *Examiner*. This choice has been motivated to be coherent with respect to standard patent search applications (i.e., Google Patents, PatentSearcher, etc.).

In case of user queries formulated by the standard textual search area, where a user writes a *name* of a person it is supposed that he/she is interested in finding inventors of patents. However, it is important to notice that by the approach also patents containing the *name* with a different role will be retrieved.

3.2 The evaluation of similar in PatentLight

In this work we improve the PatentLight engine by introducing the possibility to retrieve also fragments with different tag names w.r.t. those expressed by the query.

This feature is useful when we inquiry collections of which we roughly know the internal structure (tag names and node positions) or when we want to apply the same query to a composition of patent collections that contain more or less the same information stored in different tag nodes (for instance the node **orgname** instead of the node **Last-Name**).

In the engine, if the user chose to add the *similar* tag evaluation (in the prototype the user just flags the option in the interface), the set of FleXy queries would change accordingly as shown below:

```

People:
similar(applicants)/near::Last-Name[
text() contains text "query terms"]

```

```

applicants/near::similar>Last-Name)[
text() contains text "query terms"]

```

```

Title:
similar(invention-title)[
text() contains text "query terms"]

```

```

Descriptions:
similar(Description)/below::p[
text() contains text "query terms"]

```

```

Claims:
similar(claims)/below::claim-text[
text() contains text "query terms"]

```

```

claims/below::similar(claim-text)[
text() contains text "query terms"]

```

Note that the `similar` constraint is applied to the relevant nodes of each query and therefore the categories *People* and *Claims* will contain two queries instead of one.

The retrieved fragments are ranked according to two values: the degree of structural relevance based on the evaluation of FleXy constraints ($w_{o_c,t}$), and the degree of relevance obtained by the full-text scoring of the XQuery Full Text language (the prototype in [1] uses the BaseX system [25]). The approach privileges the structural ranking w.r.t. the content based relevance since it was observed that the paragraphs most related to the invention are usually structurally closer to the tag `Description`.

4 PatentLight development and preliminary evaluation

Aim of this work was to evaluate the retrieval capabilities of the flexible language FleXy when applied to patent collections, with particular emphasis on the `similar` constraint able to execute the same queries on different tag vocabularies.

For this reason PatentLight has been developed on top of the BaseX system [25], inheriting its indexing system and query execution engine. In this set of tests we did not use any known collection as the MAREC collection¹ but we created our test collection with XML patents from the USPTO and WIPO collections published in a small time slot (i.e., from 2015-01-01 to 2015-01-15) in order to have heterogeneous document structures and tag names. The final collection is composed by 146.413 XML patents, 82.800 from the WIPO collections and 63.613 from the USPTO collection. The architecture of the system is depicted in Figure 2.

¹ <http://www.ir-facility.org/prototypes/marec>

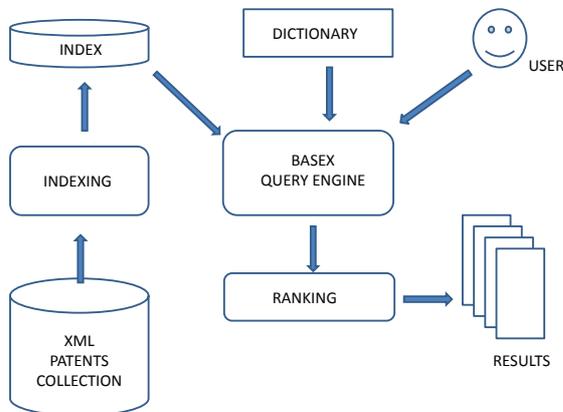


Fig. 2. PatentLight architecture

The main module is the BaseX Query engine, which is in charge of the collection indexing process and querying. During the querying process each class query is executed independently with the support of a dictionary for the **similar** constraint evaluation. One of the main characteristics of the approach is that each query produces a set of results, one for each class (*People*, *Title*, *Descriptions*, *Claims*), which are not merged. For each result two scores are computed, the **overall relevance degree** w_o (see Section 2.2) and the degree of relevance obtained by the full-text scoring of the XQuery Full Text language as implemented by BaseX. The ranking module reorganizes each class of results by first considering w_o and next the degree of content-based relevance as explained in Section 2.2.

To the aim of exploring the usefulness of FleXy on the patent collection we performed several different searches, the most interesting are shown in Figure 3. For each search we started with a very simple query, then we refined it with one more keyword or in one case two. In most cases three keywords were enough to achieve satisfactory results, i.e., a not so large number of retrieved results in each class without losing any of the relevant documents found with the less specific query.

In this set of trials we wanted to compare PatentLight and Google Patents w.r.t the prior-art retrieval task. In this task users really need to find the highest number of relevant patents as possible; in our preliminary evaluations we have carefully checked the first 50 results for each search, and compared them with the results provided by Google Patents for the same query. Figure 3 shows the number of retrieved patents for each query, in parentheses the number of relevant patents found within the first 50 results. PatentLight presents each query results divided into four classes (i.e., *People*, *Title*, *Claims*, and *Descriptions*) depending

Query	Patent Light				Google Patents
	People	Title	Claims	Descriptions	No class
Q1: «Bell»	64(0)	5(1)	98(2)	964(1)	4(0)
Q1.1: «Kettle bell»	0	1(1)	2(2)	3(1)	0
Q2: «gas turbine»	0	215(1)	453(4)	1110(2)	113(2)
Q2.1: «gas turbine compressor»	0	2 (2)	70(2)	341(2)	98(2)
Q3: «Gonzales»	8(1)	0	0	46(1)	40(0)
Q3.1: « Martino Gonzales»	1(1)	0	0	0	0
Q4: «search»	6(0)	279(1)	1770(2)	9487(2)	147(2)
Q4.1: «search engine»	0	25(2)	207(2)	2159(2)	114(0)
Q4.2: «semantic search engine»	0	2(2)	5(2)	139(1)	50(1)
Q5: «transistor»	0	346(1)	2730(1)	8312(1)	199(1)
Q5.1: « low frequency transistor»	0	0	12(4)	910(2)	110(4)

Fig. 3. Preliminary comparative evaluation of the produced results.

on the document section where the query keywords were found, while Google Patents has no classification system and results appear all together in a single list.

As shown in Figure 3, in most cases PatentLight retrieved the same number or a higher number of relevant patents w.r.t. Google Patents within the first 50 results. Moreover the results classification of PatentLight, with in average short lists, was really useful to easily find the relevant documents and discard the unuseful classes as a whole.

See for instance the first query. In this example we were looking for patents about kettle bells. Google Patents found only 4 patents, none relevant. Patent-Light found 5 patents with the word "bell" in the title, one was relevant, 64 with "bell" as person name and hence it was not necessary to check this class of results, 98 with "bell" in the claims sections and 964 in the descriptions. The mere addition of the word "kettle" drastically reduced the number of retrieved results also in the sections "claims" and "description", but the relevant documents were found anyway.

5 Conclusions and Future Work

In this paper we have described the development and preliminary evaluation of PatentLight on a collection of English patents with dishomogeneous structures. The peculiarity of PatentLight is to allow users to specify flexible constraints in their queries. Future work will study the evaluation of synonyms for the tag names used in the queries of the four categories.

References

1. S. Calegari, E. Panzeri, G. Pasi, *PatentLight: a Patent Search Application*, Proceedings of IliX 2012, Nijmegen, The Netherlands.
2. Mihai Lupu, Allan Hanbury. Patent Retrieval Foundations and Trends in Information Retrieval. 7(1): 1-97 (2013).
3. W. Magdy, P. Lopez, G.J.F. Jones, *Simple vs. Sophisticated Approaches for Patent Prior-Art Search*, In Advances in Information Retrieval, vol. 6611, Lecture Notes in Computer Science, Springer Berlin Heidelberg, pages 725-728, 2011.
4. www.google.com/patents.
5. www.patentsearcher.com.
6. B. Wiens. *Understanding Patents*. <http://www.benwiens.com/patents.html>, 2010.
7. H. Itoh, H. Mano, and Y. Ogawa. Term distillation in patent retrieval. In Proceedings of the ACL-2003 workshop on Patent corpus processing - Volume 20, pages 41-45, Stroudsburg, PA, USA, 2003.
8. T. Takaki, Query terms extraction from patent document for invalidity search. In NTCIR-5, 2005.
9. X. Xue and W. B. Croft. Transforming patents into prior-art queries. In SIGIR, pages 808-809, 2009.
10. M. Z. Wanagiri and M. Adriani. Prior art retrieval using various patent document fields contents. In CLEF-2010 (Notebook Papers/LABs/Workshops), 2010.
11. A. Fujii, Enhancing patent retrieval by citation analysis. In SIGIR, pages 793-794, ACM, 2007.
12. W. Magdy, P. Lopez, and G. J. F. Jones. Simple vs. sophisticated approaches for patent prior-art search. In ECIR, pages 725-728, 2011.
13. D. Ganguly, J. Leveling, and G. J. F. Jones. United We Fall, Divided WWe Stand: A Study of Query Segmantation and PRF for Patent Prior Art Search. In Proceedings of PaIR'11, pages 13-17, Glasgow, UK, ACM, 2011.
14. V. Hristidis, E. Ruiz, A. Hernandez, F. Fanfan, and R. Varadaraian. Patentssearcher: a novel portal to search and explore patents. In PaIR'10, pages 33 - 38, New York, NY, USA, ACM, 2010.
15. L. S. Larkey. A patent search and classification system. In ACM DL, pages 179 - 187, ACM, 1999.
16. M. Osborn, T. Strzalkowski, and M. Marinescu. Evaluating document retrieval in patent database: A preliminary report. In F. Golshani and K. Makki, editors, CIKM, pages 216 - 221, ACM, 1997.
17. C. Jochim, C. Lioma, H. Schutze, S. Koch, and T. Ertl. Preliminary Study into Query Translation for Patent Retrieval. In PaIR'10, Toronto, Ontario, Canada, pages 57 - 66, ACM, 2010.
18. D. W. Oard, D. He, and J. Wang. User-assisted query translation for interactive cross-language information retrieval. *Inf. Process. Manage.*, 44(1):181 - 211, 2008

19. H. Joho, L. A. Azzopardi, and W. Vanderbauwhede. A survey of patent users: an analysis of tasks, behavior, search functionality and system requirements. In Proceedings of the third symposium on Information interaction in context (IiX '10). ACM, New York, NY, USA, 13-24.
20. A. Trotman and B. Sigurbjornsson. NEXI, Now and Next. In Advances in XML Information Retrieval, vol. 3493, N. Fuhr, M. Lalmas, S. Malik, and Z. Szlavik, Eds., ed: Springer Berlin Heidelberg, 2005, pages 16-40.
21. W3C. XML Path Language 1.0. <http://www.w3.org/TR/xpath/>
22. W3C. XQuery 1.0: An XML Query Language. <http://www.w3.org/TR/xquery/>
23. W3C. XQuery and XPath Full Text 1.0. <http://www.w3.org/TR/xpath-full-text-10/>
24. E. Damiani, S. Marrara, and G. Pasi, A flexible extension of XPath to improve XML querying. In SIGIR, pages 849-850, Singapore, 2008, ACM.
25. C. Grun, S. Gath, A. Holupirek, and M. H. Scholl. XQuery Full Text Implementation in BaseX. In XSym '09, pages 114-128, Berlin, Heidelberg, 2009. Springer-Verlag.