●陆 伟 夏立新

基于 OKAPI 的 XML 信息检索实现研究*

摘 要 OKAPI 能实现强大的自由文本检索与评价功能,但要实现基于 XML 的信息检索,却要做适当改造。改造时,一方面尽量不破坏原有系统的体系结构,又要能实现对 XML 文档的索引、查询和表示。为此,必须完成面向 XML 的文档预处理和索引、面向 XML 的文档检索及检索模型的选择。图 3。公式 3。参考文献 10。

关键词 信息检索 OKAPI 系统 可扩展标记语言 检索模型
分类号 G354.4

ABSTRACT Okapi has strong free-text search and evaluation functions. If we want to realize XML-based information retrieval, we will have to make necessary modifications. Therefore, we should complete XML-oriented file pre-processing and indexing and select XML-oriented file retrieval models. 3 figs. 3 formulas. 10 refs.

KEY WORDS Information retrieval. Okapi system. Extensible Markup Language. Retrieval model.
CLASS NUMBER G354.4

长期以来,信息检索研究往往关注于非结构化信息(自由文本),而很少关注文档结构所蕴涵的语义信息。XML 作为半结构化信息的标记语言,不仅仅需要考虑如何从文档中找到相关信息,也需要考虑信息的结构和粒度问题,也就是说要实现内容 + 结构(Content and Structure, CAS)的信息检索(另外一种称为 COS, 即 Content Only + Structure, 这种类型往往应用于 XML 文档的相关反馈研究)[1]。这对传统的信息检索实验系统提出了挑战,因为它们其中的大部分都是构建于自由文本检索的基础之上。要适应基于 XML 的信息检索,必须对原有的系统进行改造和升级。本文将探讨如何在改造和完善自由文本检索系统 OKAPI 的基础上,实现基于 XML 的信息检索。

1 OKAPI 概述

OKAPI 是伦敦城市大学信息科学系的一个信息检索实验系统或者说是系统集的简称。它是在多个研究用户信息搜寻行为和用户系统交互项目的基础上发展而来,这些项目得到了许多机构尤其是大英图书馆的资助。OKAPI 最早起源于 Polytechnic of Central London(今天的威斯敏斯特大学),在那里主要是用于研究 OPAC(Online Public Access Catalogue)以及在线主题检索方面的一些项目。1989 年该系统随着其开发者 Steve Walker 一起迁移到了城市大学的交互系统研究中心,并在这里得到了全面快速发展[2]。该研究中心是由著名的信息检索专家 Stephen Robertson 创办。由于在发展、完善和评价概率模型方面的突出贡献,继 Karen Sparck Jones, Cyril Cleverdon, William Cooper 和 Tefko Saracevic 之后,2000 年 Stephen Robertson 被授予沙尔滕奖。与 SMART 等一起,OKAPI 已成为国际公认的少数几个成熟稳定的信息检索实验系统之一,在历年的 TREC 国际会议上,OKAPI 都取得了出色成绩。

作为一个信息检索实验系统,OKAPI 主要由以下 3 部分组成。

(1)索引器:主要用于对数据集合进行处理,生成数据辞典和倒排文档,倒排文档中包含完整的位置信息。OKAPI 的索引器包含很多可配置的参数,包括词典选择、词干提取、同义词归并、多索引文件的配置等,可以很好满足多种自由文本信息检索实验的需要。通过增加文本分段,还实现了最佳段落检索功能,不过这项功能由于其核心逻辑模型中增加了文档长度这一参数,而显得不是十分重要。OKAPI 也实现了文本域的处理功能,为基于 XML 的信息索引与处理打下了良好基础。美中不足的是,OKAPI 不支持增量索引,当源文档集改变时,必须重新运行索引程序生成索引。OKAPI由 convert_runtime, ixl 和 ixf 3 个命令来创建索引。

(2)搜索器:在 OKAPI 中此模块称为 BSS(Basic Search System)。它既提供了底层函数来实现权重分

* 本文系国家社会科学基金项目"基于中文 XML 文档的全文检索研究"(批准号 04CTQ005)的阶段性研究成果之一。

配与排序检索,也提供了结果集的布尔和概率操作。OKAPI 系统采用概率模型作为其系统模型的核心。该模型起源于 1976 年 Stephen Robertson 和 Sparck Jones 提出的 IDF(Inverted Document Frequency)模型[3]。并在 TF(Term Frequency) - IDF 模型的基础上,发展成为著名的 BM25 模型。BSS 中包含多个 BM25 模型及其变种,用以实现不同的信息检索目的,如 BM25b, BM250, BM251 等[4]。常用的 BSS 函数命令有 choose 数据集选择,parse 或 superparse 查询词解析,find 或 combine 结果查询或结果集匹配,weight 权重计算以及 show 结果显示等。

(3)其他接口系统。它们中有命令行的,基于 TCL/TK 的 windows 界面,也有基于 PHP 的 web 界面等。这些接口系统主要用来实现系统参数环境配置、系统索引与检索功能调用以及一些面向特定目标的实验系统如相关反馈系统等。总的来说,OKAPI 的体系结构如图 1 所示[5]。

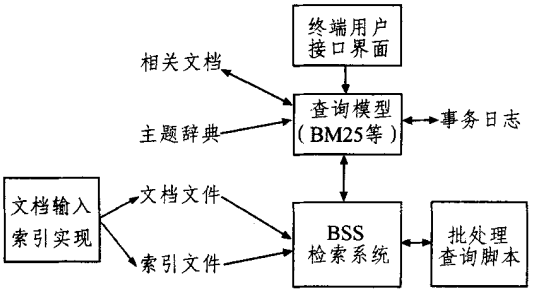

图 1 OKAPI 系统结构

需要强调的是,在一些特定的应用中需要应用词典功能,如相关反馈中的词典提示,自动分类中的词典参照等。OKAPI 的另一个比较有用的功能是事物日志,记载了大量的用户系统交互、试验数据结果集信息等,它可以用做用户行为跟踪分析、信息检索效果评价分析等。

2 OKAPI 环境下 XML 检索的实现

OKAPI 能实现强大的自由文本检索与评价功能,但要实现基于 XML 的信息检索却需要适当改造。当然,OKAPI 也可像实现自由文本检索一样实现对 XML 文档的检索,但是这样一来它却无法提取 XML 标签中所蕴涵的语意信息进行面向 CAS 或 COS 的信息检索。对 OKAPI 的改造,一方面尽量不破坏原有系统的体系结构,又要能实现对 XML 文档的索引、查询与表示。要实现这一目标,必须完成面向 XML 的文档预处理与索引、面向 XML 的文档检索及检索模型的选择。

2.1 XML 的文档预处理与索引

OKAPI 不支持增量索引,因此,对于非 XML 文档,在索引建立前,需要一个文档处理器将源文档文件进行归并,生成一个以. exch 结尾的 exchange file。然后调用 convert_runtime,生成. bib 文件和. bibdir 文件,最后利用索引创建文件 ixl 和 ixf 创建索引文件。如图 2 中,参数文件包含了创建索引所需的各种参数信息;对于非 XML 文件,. bib 文件包含了文档记录和各记录的域长度信息,. bibdir 文件存储着. bib 文件中文档记录的位移信息;Id_j 是生成的索引文件集合,包括数据辞典、倒排文档等。

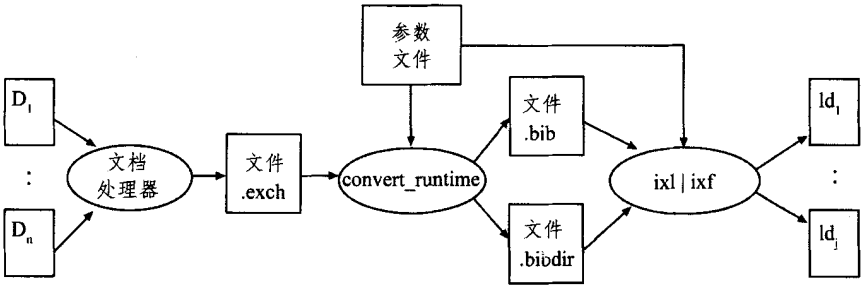

图 2 非 XML 文档的索引过程

对于 XML 文档,同样需要一个文档处理器对 XML 源文件进行预处理,处理后直接生成. bib 文件。该文件只包含了简单解析归并后的 XML 文档记录,并不包含 XML 文档元素位移与长度等信息,该信息与 XML 记录位移信息被一起存储在. bibdir 文件中。XML 文档的预处理还需要一个 XML 文档解析器,对 XML 文档的合法性进行检验,并对文档的结构进行解析。目前,有多种关于 XML 的文档解析器如

DOM、SAX、JDOM 等。本系统中，以 INEX 2005 数据集为样本[6]，采用开源代码库 Expat XML 文档解析器。Expat 是用 C 编写的 XML 文档解析库，应用在 Mozilla 等一些著名开源项目中，其开源软件可以在 http://sourceforge. net/projects/expat/上下载。XML 文档处理器调用 Expat 解析器，根据 XML 文档规则文件. dtd 对 XML 文件进行解析并合并生成. bib 文件。XML 文档的索引过程如图 3 所示。

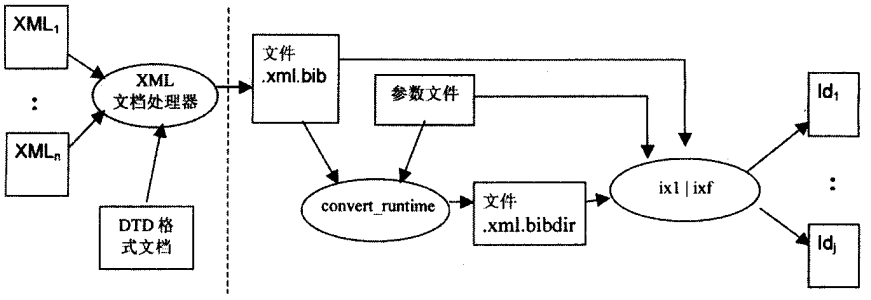

图 3　XML 文档的索引过程

对于自由文本索引，OKAPI 索引文件中包含了索引词在文档中的位置信息；要实现 XML 的结构化检索，则需要根据. bibdir 文件中元素的位置和长度信息结合索引词的位置信息，生成索引词在特定元素中的词频、位移等信息，并存储在倒排文档中。而对于 XML 文档，并不需要对所有元素都建立索引，那样会大大降低系统的性能，如以 INEX 2005 数据集为例，平均每篇文档包含 1000 多个 XML 元素，对每个元素都建立索引完全没有必要。在 OKAPI 中，将需要建立索引的 XML 文档元素存储于参数文件中，索引系统将根据参数设置对指定 XML 元素建立索引。在参数文件中，有两个参数决定了 XML 文档的索引元素信息：r_abbrev 和 f_abbrev。r_abbrev 是 XML 的记录标签，索引程序将根据此标签进行记录读操作；f_abbrev 是 XML 文档的元素标签，索引程序将根据给定元素标签对特定的元素建立索引。另外，nf 参数给定了预建立索引元素的个数。下面给出了对文章号、作者、标题、摘要和正文等元素建立 INEX 2005 索引的参数信息：

```
nf = 5
r_abbrev = < article >                    //文档记录标签
f_abbrev = < fno >                        //文章号
f_abbrev = < fm > < tig > < atl >         //标题
f_abbrev = < fm > < au >                  //作者
f_abbrev = < fm > < abs >                 //摘要
f_abbrev = < bdy >                        //正文
……
```

下面是一个 INEX XML 文件的部分结构：

```
< article >
< fno > A1003 < /fno >
< doi > 10. 1041/A1003s — 1995 < /doi >
< fm >
  < hdr >
    < hdrl >
      < ti > IEEE Annals of the History of Computing < /ti >
      < crt > < issn > 1058—6180 < /issn > /95/$ 4. 00…… < /crt >
    < /hdrl >
    ……
  < /hdr >
  < tig > < atl > About this Issue < /atl > < pn > pp. 3—3 < /pn > < /tig >
  < au sequence = "first" > < fnm > J. A. N. < /fnm > < role > Editor Chief < /role > < /au >
< /fm >
< bdy > …… < /bdy >
< /article >
```

由于考虑到与原有系统的兼容性,即可以在半结构化检索实验中使用原有系统的各种模型如 BM250 最佳段落查找、ADJ 词语相连匹配检索等,改造后的系统可以最多支持 32 个 XML 文档元素,如果不考虑原有系统的功能,为满足特定需要,元素最多可以达到 2 的 16 次幂,这足可以满足各种类型的 XML 索引需要。

2.2 XML 文档的检索

改造后的 OKAPI 充分考虑了兼容原有系统的功能,索引建立后,就可以使用原有的所有模型和功能进行 XML 文档的非结构化检索。但这并不是改造系统的最终目的,它是要希望能够实现面向 CAS 的 XML 检索。要实现这一目标,有两点需要考虑:其一是检索结果集的获取,其二是检索结果的排序。对于前者,由于 XML 索引文件中存储着索引词在特定元素中的词频信息,因而在给定索引词并限定元素范围的情况下,可以很容易获取检索结果集,我们通过对 bss 中的 find 函数进行适当改造来实现这一点。对于后者,即检索结果的排序,涉及到检索算法模型的选择问题。OKAPI 系统的核心检索算法模型是概率模型 BM25[7],简化后该公式的模型为:

$$w_j(\bar{d},C) = \frac{(k_1+1)d_j}{k_1\left((1-b)+b\frac{dl}{avdl}\right)+d_j}\log\frac{N-df_j+0.5}{df_j+0.5} \quad (1)$$

该公式反映了在文档集 C 中,第 j 个词在文档 $\bar{d}$ 中的权重,其中:d_j 为第 j 个词在文档 $\bar{d}$ 中的词频,dl 为文档长度,$avdl$ 为文档集 C 的平均文档长度,N 为 C 中文档总数,df_j 为 C 中含第 j 个词的文档数量,k_1 和 b 为两个自由变量,可以根据不同的文档集合予以调整,以达到最佳效果。

根据该公式,给定查询语句 q,则文档权重得分为[8]:

$$W(\bar{d},q,c) = \sum_j w_j(\bar{d},C)\cdot q_j \quad (2)$$

以上两公式是针对非结构化条件下的文档检索算法模型。对于半结构化的 XML 文档而言,它们显然不能完全适用。目前,半结构化 XML 的概率检索算法模型可以有以下几种考虑:(1)文档权重得分等于指定元素权重得分;(2)文档权重得分等于各元素权重得分之和;(3)文档权重得分是在各元素词频加权之和后计算所得。对于第一种思路,我们简单地用元素词频 $d_{f,j}$ 代替文档词频 d_j 来计算给定词的文档权重得分,当然也需要进一步将文档长度 dl 和文档平均长度 $avfl$ 替换为文档特定元素长度(即文档元素内词频)fl 和特定元素平均长度(即特定元素总词频/文档总数)$avfl$。对于第二种思路,我们用元素词频 $d_{f,j}$ 代替文档词频 d_j,用文档特定元素长度 fl 和特定元素平均长度 $avfl$ 代替文档长度 dl 和文档平均长度 $avdl$ 计算给定词的元素权重得分 $w_{f,j}$,然后将各元素权重得分相加获得给定词的文档权重得分。对于第三种思路,我们考虑到 BM25 模型的文档适用性,结合元素的重要性,对文档中给定词的元素词频进行加权,并将加权后各元素词频累计相加,计算出给定词在文档的加权词频,然后利用上面的文档权重得分公式计算文档权重得分。需要说明的是,此时文档长度和平均长度因元素加权也应相应调整。另外,前两种方式改变了 BM25 模型实现范围和模式,Robertson 等在其文章中详细探讨了这两种方法的危险性和不足[9]。因而,我们在实际应用中主要是使用第三种方法,其实现公式为[10]:

$$wf_j(\bar{d},C) = \frac{(k'_1+1)d'_j}{k'_1\left((1-b)+b\frac{dl'}{avdl'}\right)+d'_j}\log\frac{N-df_j+0.5}{df_j-0.5} \quad (3)$$

其中 d'_j 代表加权后第 j 个词在文档 $\bar{d}$ 中的词频,dl' 代表加权后的文档长度,$avdl'$ 代表加权后文档的平均长度。

由于篇幅的限制,本文将不对这几种算法模型的具体实现及参数说明详细论述。我们将在专门的文章中对不同模型尤其是第三种模型的运行效果进行评价分析。

3 结论

改造后的 OKAPI 系统实现了非结构化和半结构化的 XML 信息检索,根据参数文件可配置的索引过程使得它可以对当前几大信息检索数据集如 TREC 路透新闻数据集、INEX 数据集等进行索引处理,并可以在此基础上进行各种相关的 XML 检索试验。然而,这只是我们迈出的第一步,文中提到的索引方法还有待进一步改造以适应普适性的 XML 检索需求;我们在系统中采用的几种基于 BM25 的改造模型,其效果还有待于进一步实验去评价和检验;我们的检索

模块在查询语句的表达上还需要进一步融入XQuery、XPath等XML检索语言。我们也将在此基础上研究基于XML的相关反馈、异构XML数据的处理转化与评价问题等。

致谢:

感谢国家留学基金委资助本文第一作者访问伦敦城市大学从事XML检索的相关研究工作,也感谢Stephen Robertson教授和Andrew Macfarlane博士的悉心帮助和指导。

参考文献

1,6 N. Govert and G. Kazai. Overview of the Initiative for the Evaluation of XML retrieval(INEX)2002. Proceedings of the 1st Workshop of the Initiative for the Evaluation of XML Retrieval(INEX). 2002,1 - 17

2 Introduction of Okapi. Available via http://www.dotty.soi.city.ac.uk/~andym/OKAPI-PACK/.

3 Robertson,S. E. and Sparck Jones,K. Relevance weighting of search terms. Journal of the American Society for Information Science,1976,27:129 - 146.

4,5 Robertson,S. E. Overview of The OKAPI Projects. Journal of Documentation,1997,53(1)

7,9,10 Robertson,S. E. and Steve Walker. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval. 1994,345 - 354.

8 Robertson,S. E. ,Hugo Zaragoza and Michael Taylor. Simple BM25 Extension to Multiple Weighted Fields. CIKM 2004. ACM Press. 2004,42 - 49.

陆 伟 博士,副教授。通信地址:武汉大学信息资源研究中心。邮编430072。

夏立新 博士,华中师范大学信息管理系教授,华中科技大学管理学院博士后。通信地址:武汉华中师范大学信息管理系。邮编430079。 (来稿时间:2005 - 12 - 09)

(上接第55页)用于生成包装层的公共代码,如包装层与网络的连接操作等代码放入公共代码库中。

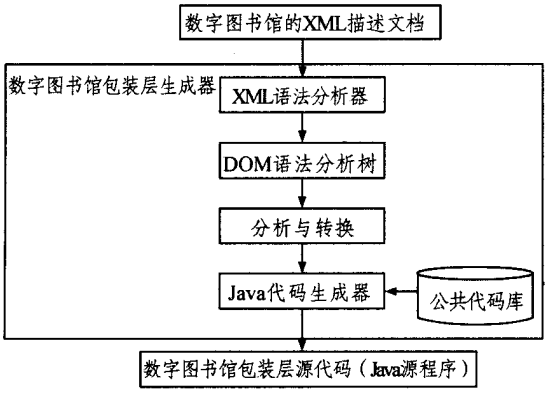

图6 数字图书馆包装层生成器的结构模型

参考文献

1 Yue-Shan Chang, Min-Huang Ho, Wen-Chen Sun, et al. Supporting unified interface to wrapper generator in integrated information retrieval. Computer Standard & Interfaces, 2002, 24(4)

2 Lieming Huang, Matthias Hemmje, Erich J. Neuhold. ADMIRE: an adaptive data model for meta search engines. Computer Networks, 2000, 33:431 - 448

3 Chen-Chuan K. Chang, Hector Garca-Molina. Mind Your Vocabulary: Query Mapping Across Heterogeneous Information Sources. in : Proc. SIGMOD 99, Philadelphia, PA, June 1999. pp. 335 - 346

4 Donald Kossmann. The State of the Art in Distributed Query Processing. ACM Computing Surveys, 2000,32(4)

5 冯少荣. 基于XML和JAVA构建程序生成器. 计算机应用与软件,2005,22(1)

张付志 燕山大学信息科学与工程学院博士,教授。通信地址:河北省秦皇岛市。邮编066004。

肖 阳 燕山大学信息科学与工程学院硕士研究生。通信地址同上。 (来稿时间:2005 - 11 - 08)