基于 XML的 PDF文档信息抽取系统的研究 *

宋艳娟 张文德
(福州大学数学与计算机科学学院 福州 350002) (福州大学图书馆 福州 350002)

【摘要】 首先设计了科技论文的 DTD文档,然后分析了 PDF文档的结构。在此基础上,我们介绍了 PDF文档信息抽取系统的设计框架。该框架以上述 DTD为模板,把以 PDF格式表示的科技论文解析转换为有效的 XML文档。

【关键词】 信息抽取 PDF XML 【分类号】 TP392

Research on PDF Documents Information Extraction System Based on XML

Song Yanjuan
(College of Mathematics and Computer Science, Fuzhou Uninversity, Fuzhou 350002, China)
Zhang Wende
(Library of Fuzhou Uninversity, Fuzhou 350002, China)

【**Abstract**】 The article is structured as follows. Firstly, we try to design a DTD of articles of science and technology. Secondly, we analyze the structure of PDF documents. Based on that, we dwell on the design of a PDF information extraction system, which use the above - mentioned DTD as a template, transfer a PDF - formatted scientific and technological article to a valid XML document.

【**Keywords**】 Information Extraction PDF XML

1 引 言

结构化的文档格式 PDF是由美国的 Adobe公司提出的[1]。PDF文件格式以其卓越的特性成为在 Internet上进行电子文档发行和格式化信息传播的理想文档格式。当前,在 Internet中的科技论文以 PDF格式提交变得越来越流行,如万方数据库。但是,PDF重在描述文档的打印格式,并没有描述文档内容的数据结构。这成为制约人们信息检索的瓶颈。因此,对 PDF进行信息抽取的研究显得十分重要。XML是 W3C推荐的数据交换的标准[2]。它的出现推进了网络表达的语言集合。XML是 Internet环境中跨平台的,依赖于内容的技术,是这个时代中处理分布式结构信息的选择工具。XML是面向内容的,因此它能够弥补 PDF文件格式在语义描述方面的不足。

本文首先介绍了描述科技论文的语义框架的 DTD文档,并简要说明了 PDF的文件格式。在此基础上,我们提出了如何将以 PDF格式表示的科技论文解析转换为有效的 XML文档,从而实现格式标注向语义标注转换的设计框架。

收稿日期: 2005 - 05 - 23
收修改稿日期: 2005 - 06 - 07
*本项目是福建省高等学校科技项目 (JA04164)的研究成果之一。

2 PDF文档信息抽取系统设计的工作流程

2.1 DTD (文档类型定义) 的设计

要将 PDF文档中的语义信息较好地表现出来,首要的一步是制定规范 XML文档中元素和标志的规则及相互关系的 DTD文档。

我们参考了广受欢迎的 DocBook元素的子集 Simplified DocBook[3],根据科技论文具有篇章结构和用语规范的特点,分析并选择出以下两类基本的信息:

(1)外部信息元数据 (Articleinfo):描述科技论文外部特征的元数据,包括 Author(作者),Address(作者地址),Edition(出版),Bibliography(参考文献)等。外部信息元数据是用户进行信息检索的重要依据。

```
<! ELEMENT Articleinfo ( authorgroup , edition, bibiography) >
<! ELEMENT authorgroup (address , author + ) >
<! ELEMENT address (department, city, zip , email ) >
<! ELEMENT author (name , birth , sex , degree, research) >
```

<!ELEMENT edition (ediname, pagenums, volumenum, issuenum, pubdate) >

<!ELEMENT bibliography (bibliodiv+) >

<!ELEMENT bibliodiv(title, biblioentry) >

<!ELEMENT biblioentry ((auth_group, title, publisher, date) | ulink) >

<!ELEMENT auth_group (author_name+) >

<!ELEMENT publisher (publishername, address) >

<!ELEMENT department (#PCDATA) >

<!ELEMENT city (#PCDATA) >

… …

<!ELEMENT ulink (#PCDATA) >

<!ATTLIST ulink url CDATA >

(2)内部信息元数据:描述文章语义信息的元数据,包括 Title, Abstract, Keywordset, Section, Para等。利用文章的语义信息进行检索,能在很大程度上提高用户信息检索的效率。

① Title (文章标题):最直接地反映了文章的核心内容

<!ELEMENT Title (#PCDATA) >

② Abstract: 论文的摘要

<!ELEMENT Abstract (#PCDATA) >

③ Keywordset: 论文关键词的集合

<!ELEMENT Keywordset (keyword+) >

<!ELEMENT keyword (#PCDATA) >

④ Section: 文章的章节。为了更好地实现文章信息的分类和检索,我们有必要对文章的篇章结构进行分析。文章由章节 Section组成, Section中包含段落 Para子元素,并且可以嵌套 Section。我们工作中很重要的一项任务是完成对章节主题以及段落主题的判断。关于章节的划分,章节主题和段落主题的判断,将在 2.3.4部分阐述。

<!ELEMENT Section (sect_theme, (Section|para+) *) >

<!ELEMENT sect_theme (#PCDATA) >

<!ELEMENT para (para_theme *) >

<!ELEMENT para_theme (#PCDATA) >

<!ATTLIST para id ID #REQUIRED >

2.2 PDF的文件格式[1]

要实现对 PDF文档的语义信息的抽取,必须十分清楚 PDF的文件格式。

(1) PDF的对象

组成 PDF文档的基本元素是 PDF对象 (PDF Object)。PDF支持 7种基本的对象类型: Boolean (布尔型), String (字符串型), Name (名字型), Dictionary (字典型), Number (数值型), Array (数组型), Null (空对象), Stream (流对象)。其中,字典对象 (Dictionary Object)是 PDF文档的主要构成部分。PDF文档中的页面,字库等部分都用字典对象表示。

PDF对象可以分成直接对象 (Direct Object)和间接对象 (Indirect Object)。其中, PDF间接对象是一个被标志过的对象。它由对象标志符,直接对象和关键字 Endobj组成。PDF文档中使用了大量的间接对象和间接引用。

(2) PDF的物理结构

PDF的物理结构 (文件结构)由四部分组成。可用图 1表示。

文件头 (Head)

文件体 (Body)

交叉引用表 (Cross Table)

文件尾 (Trailer)

图 1 PDF的物理结构

①文件尾 (Trailer)中主要包含了交叉引用表的地址,文件体根对象 Catalog的地址以及文档加密等信息。

②交叉引用表 (Cross Table)是为了实现对间接对象的随机存取而特设的地址索引表。

③文件体 (Body)由大量的 PDF间接对象组成。间接对象构成了 PDF文档的具体内容,如字体,页面,表格,图像等。如何对文件体中的间接对象进行处理是我们信息抽取工作的主要工作量。

④文件头 (Head)指明了 PDF文档所遵从的 PDF规范的版本号。例如 %PDF-1.4表示该文档格式符合 PDF1.4规范。

(3) PDF的逻辑结构

PDF的逻辑结构反映了文件体中的间接对象之间的层次关系。它是一种树型结构[4]。树的根节点是 PDF文件的根对象 Catalog。根节点下有四棵子树,分别介绍如下。

①页面树 (Pages Tree):所有的页面对象都是树的叶节点。每一页包含了对该页的内容 (Contents),注释 (Annotations),缩略图 (Thumbnail)的引用。其中, Content stream (内容流)描述的是该页的文本内容。关于内容流的提取,将在下一部分中阐述。

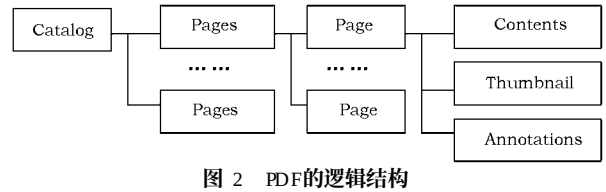

图 2 PDF的逻辑结构

②目录树管理书签 (Bookmark): PDF文档中的 Outline Tree是一个树型层次结构。其中每个节点都是一个书签 Bookmark。书签名 (Bookmark Name)和具体的页面位置一一对应。应用程序能够按照书签名访问文档的内容。关于书签在 PDF文档信息提取中的作用,我们将在 2.3.4中阐述。

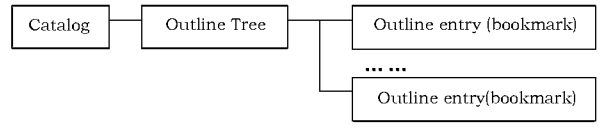

图 3 Outline Tree层次结构

③线索树 (Threads Tree):按树型结构组织文章线索和线索下的文章块。

④名字树 (Name Tree):建立了一个字符串和页面区域之间的关联。

2.3 PDF文档信息抽取的设计框架

(1)提取存放 PDF文档中各页的内容流,并进行解码

应用程序首先访问文件尾,从文件尾中读取交叉引用表的地址和 PDF文件的 Catalog根节点[4]。根据交叉引用表中的地址访问 PDF文档中的间接对象,从而控制整个 PDF文档。具体实现流程如下:

①从文件尾 Trail中找到 PDF根对象。该对象的类型 Type为 Catalog。

②通过 Catalog根节点找到页树节点对象。该对象的 Type为 Pages。

③通过 Pages子节点找到页对象。该对象的 Type为 Page。

④访问页对象 Page中的内容。如果找不到 Contents入口,则说明此页内容为空,不予处理;如果找到 Contents入口,转下一步进行处理。

⑤从 Contents入口得到 Contents后的所有间接对象号,并依次将该对象号记录到该页的内容对象数组 Con_objNo[ ]中。

⑥根据数组 Con_objNo[ ]中的每一个对象号,转到相应的间接对象位置;提取 Filter后的解码名,并将解码名放入数组 Con_encode[ ]。

⑦根据数组 Con_encode[ ]中的解码名,调用 Java的 iText软件包的 PdfReader类中的 Filter解码方法对 Con_objNo[ ]数组中的对象的内容流进行解码。

⑧将 Con_objNo[ ]数组中的所有对象解码后的字符串用指针连接起来,组成该页内容流解码后的字符串 TextStr。

⑨重复以上工作,将各页内容流解码后的字符串用指针连接起来,组成单向链表,写入中间文件中。

注意:

①页树中各节点的组织具有先序深度优先的性质。应用程序采用先序遍历算法读出所有的页面对象和属性,再将这些内容依次写入中间文件中。这样,访问页面节点的次序与页面的实际页码是一致的[5]。

②如果 PDF文档是英文文档,中间文件中解码后的字符串就是原始文档的内容;而对于中文,中间文件中给出的是汉字的编码,需要经过编码的转化才能还原为原文内容。

(2)将 PDF文档的物理结构转化为逻辑结构

从中间文件中,我们可以得到以下几项重要信息:

① Content:每个页面中每一行的文本内容;

② Position:每行的位置 (x, y);

③ Page: 该行所处的页面;

④ Font Type:描述该行中多数文本内容采用的字体类型;

⑤ Font Size:描述该行中多数文本内容采用的字体大小。

由于中间文件描述的只是文档的物理结构,并不具有任何的语义信息[6]。我们这一步的工作便是从中间文件中获取信息,生成按照人们阅读习惯组织的文章的逻辑结构。具体实现分为两步:

①排版分析:该步的目的是将以物理行为单位的中间文件转化为以逻辑行为单位的文件。对单栏排版的文章,逻辑行在一定意义上等同于物理行。而对于多栏排版的文章,要按栏为单位的方式对行进行重组。该步的核心是区分不同栏但同行的字符串。

②逻辑转换:经过排版分析的处理,我们得到了按照文章的物理顺序组织的字符串链表。逻辑转换所做的工作就是将字符串链表改成按照人们阅读文章时的顺序组织的文章的逻辑链表。系统使用聚类算法,依据各字符串之间的对齐方式,将属于同一栏的内容聚集在一起。

(3)外部信息元数据的提取

经过上面几步的预处理,我们得到了描述文章的逻辑链表。接下来,我们要做的工作是对应 DTD文档中的定义,判断 PDF文档的外部信息元数据。

对于第一作者的提取,我们制定的规则如下:

①该字符串的 Position的 y值与已提取出的 Title的 Position的 y值最接近,关于如何提取文章标题 Title将在 2.3.4中阐述;

②该字符串的 Font Size小于 Title的 Font Size。

如果一个字符串同时满足上述两个条件,应用程序则判断其为第一作者,写入元素 Author的子元素 Name中。

对于非第一作者的提取,我们参照如下规则:

①该字符串的 Position的 y值等于第一作者的 Position的 y值;

②该字符串的 Font Size和 Font Type与第一作者的相同。

对于同时满足上述条件的字符串,应用程序同样将这些字符串写入元素 Author的子元素 Name中。其余的为作者对应的单位的地址,名称,邮政编码等信息。

(4)内部信息元数据的提取

遍历整个逻辑结构链表,抽取出文档的内容信息。

①Title文章标题的提取

对于标题的提取,我们参照如下规则:1) 该字符串的 Page为第一页;2) 该字符串的 Position的 y值最大;3)该字符串的 Font Size最大。同时符合上述条件的字符串,应用程序都认为是标题的一部分。

②Section章节信息的提取

如前所述,PDF文档中的 Outline Tree是一个树型层次结构。其中每一个节点都是一个书签 Bookmark。系统使用书签 Bookmark提取章节 Section的信息。具体的实现方法如下:1) Bookmark节点在 Outline Tree中的深度对应转换成 XML文档中章节的层次结构;2)章节的主题 Theme的内容为 Bookmark的文本内容;3)章节中包含的段落以 Bookmark指向文档中的具体位置为依据。

③Para段落信息的提取

对于段落的判断,我们制定的规则如下:1)如果两行文本之间的间距大于平均的行距,则得出“这两行文本分属两个段落”的结论。2)如果行首文本的横坐标大于前一文本行的行首横坐标,则判断该行是一个新段落的开始。

段落中最重要的信息是主题的表示。主题表示的方法常用的有两种:一是摘要形式;二是关键字形式。在系统中,我们采用关键字来表示段落的主题。系统使用了中文信息处理手段[7]提取段落主题。具体步骤如下:

①汉语分词:汉语分词是由计算机自动识别文本中的词边界的过程,可用处理函数 a = F(b)表示。其中,b为汉字字符序列 (b_1 b_2 … b_n),a为汉语词串的组合序列 (a_1 a_2 … a_m)。不同的 F(b),有不同的 a。经过分析和比较,我们采用最大正向匹配算法作为 F(b)。该算法依据一个分词词表和“长词优先”的原则,进行分词。它的基本思想如下:设分词词表中的词由 i个汉字组成,取汉字字符串序列中的前 i

个汉字作为匹配字段,查分词词表。若能匹配,则将这个匹配字段切分出来,填入数组 a[ ]中 ;若不能匹配,则将匹配字段的最后一个字去掉,重复以上过程,直到匹配为止。

②词性标注: 使用专门的工具对分词后的结果 a[ ]中的词进行词性的标注。

③选择关键字: 根据词性标注后的结果, 判断每个段落中的所有的名词。对于这些名词,求出它们的词频。根据香农信息论,区别段落中最有意义的词语应该是那些在段落中出现频率足够高,但在段落集合 (文章)的其他段落中出现频率足够少的词语。我们参考了 TFIDF(Term Frequency Inverse Document Frequency)向量表示法[8],定义的计算词频的公式为:

$$x_i = freq(w_i) \log(N/DF(w_i))$$

其中, $freq(w_i)$表示 w_i在段落中出现的次数; $DF(w_i)$是拥有词语 w_i的段落数目, N表示目标文章中的段落总数。然后,选择词频最高的若干个名词作为该段的关键字, 写入元素 Para的子元素 Para_theme中。

2.4 生成 XML文档

PDF文档经过解码,章节划分,自动分词等处理后,最终的结果就是建立了文本结构树。在此基础上,我们可以实现面向内容的满足已定义好的 DTD的 valid XML文档的生成了。

3 结 语

本文主要描述了 PDF文档信息提取系统的设计框架,并对将 PDF转换为 XML文档过程中的相关技术进行了一些阐述。用户可以对转换后的 XML文档做进一步的操作,从而提高文档自动分类和用户信息检索的效率。

参考文献:

1 Adobe Systems Inc. PDF Reference, Adobe Portable Document Format version 1.4_, 3nd, 2001. http://www.adobe.com/support/downloads/product.jsp?product=44&platform=Windows(Accessed Mar. 8, 2005)

2 Extensible Markup Language 1.0 Second Edition. http://www.w3.org/TR/REC-xml, 2000-10(Accessed Mar. 8, 2005)

3 Simple DocBook. http://www.docbook.org/xml/simple/1.1CR2/(Accessed Mar. 8, 2005)

4 杨道良等. 面向对象的中文 PDF阅读器的设计与实现. 计算机应用, 1999, 19(6): 1-4

5 Introduction to XML, Java, databases and the web Nazmul Idris 1999/06/24. http://www.developerlife.com (Accessed Mar. 8, 2005)

6 Norbert Fuhr. XML Information Retrieal and Information Extraction. http://ls6-www.informatik.uni-dortmund.debibfulltext/ir/Fuhr:02a.pd, 2002 (Accessed Mar. 8, 2005)

7 余锦凤等. 中文信息处理基础教程. 北京:北京大学出版社, 2002

8 李辉,史忠植等. 运用文本领域的常识改善基于支撑向量机的文本分类器性能. 中文信息学报, 2002, 16(2): 7-13

9 Ekkuitte Rusty Harold著,杜大鹏等译. XML实用大全. 北京:中国水利水电出版社, 2001

(作者 E-mail: zhangwd@fzu.edu.cn)

(上接第 5页)

10 A. Shiri Schemas and Ontologies: Building a Semantic Infrastructure for the Grid and Digital Libraries. Workshop Report from E-Science Institute, Edinburgh, 2003

11 P. B. Watry and R. R. Larson. Cheshire 3 Framework White Paper: Implementing Support for Digital Repositories in a Data Grid Environment. The International IEEE-Computer Society Symposium Mass Storage Systems and Technologies, Italy, June 19-24, 2005

12 R. R. Larson and R. Sanderson. Grid-Based Digital Libraries: Cheshire3 and Distributed Retrieval. Joint Conference on Digital Libraries, Denver, Colorado, USA, June, 2005

(作者 E-mail: zyyang@mail.sjtu.edu.cn)

(上接第 9页)

7 邵宁. 网格技术将引领信息技术发展大潮. http://eg1.cn/zhuanti/zhuanti4-3/view.asp?id=130(Accessed Feb. 8, 2005)

8 William K. Cheung, Jiming Liu. ON Knowledge Grid and Grid Intelligence: a survey, Computational Intelligence, Volume 21, Number 2, 2005

9 http://www.w3c.org(Accessed Feb. 8, 2005)

10 I Foster, C. Kesselman, G. Tsudik, S. Tuecke. A Security Architecture for Computational Grids. ACM Conference on Computer and Communications Security 1998: 82-89

11 G. Gheorghiu, T. Ryutov, B. C. Neuman. Authorization for Metacomputing Applications, HPDC 1998: 132-139

12 Minglu Li, Hui Liu, et al ShanghaiGrid in Action: The First Stage Projects towards Digital City and City Grid. Grid and Cooperative Computing: Second International Workshop, GCC 2003, Shanghai, Springer-Verlag Heidelberg press, 2003: 616-623

(作者 E-mail: hongfeng@sjtu.edu.cn)