

一种基于多叉树的 HTML 到 XML 的转换方法

张文斌 陈恩红 王 进

(中国科学技术大学 计算机科学系, 安徽 合肥 230027)

摘 要: 当前的 Web 信息大多数都是 HTML 格式的, 由于 HTML 文件中没有严格的结构性, 故很难能用一种有效的方法来检索或提取隐藏其中的数据. 针对 HTML 的这种缺陷, 本文提出了基于多叉树的 HTML 到 XML 转换方法, 把对 HTML 的信息检索问题转化为对 XML 的检索问题, 以便简化下一步的检索问题.

关 键 词: HTML; XML; 多叉树; 信息检索

中图分类号: TP312

文献标识码: A

文章编号: 1000-1220(2003)04-0713-03

A Multi-tree Based HTML to XML Transformation Approach

ZHANG Wen-bin, CHEN En-hong, WANG Jin

(Department of Computer Science, University of Science and Technology of China, Hefei 230027, China)

Abstract: Large volume of current Web information is in HTML format. However HTML file has no strict structures, therefore it is difficult to retrieve or extract its hidden data. To overcome the shortcoming, the paper proposes a multi-tree based HTML to XML transformation approach so that HTML information retrieval problem is transformed into XML information retrieval problem, thus simplifying information retrieval task.

Key words: HTML; XML; multi-tree; information retrieve

1 引 言

当前的 Web 信息大多数都是 HTML^[2] 格式的, 由于 HTML 具有结构简单性和灵活性, 它极大地促进了信息产业的发展, 但是也正是由于 HTML 结构太灵活和自由, 造成了一个致命的缺陷: 难以检索或者提取隐藏其中的数据. 针对 HTML 的这种缺陷, XML 语言应运而生, 它一方面继承了 HTML 的灵活性和简单性, 另一方面又对其存在的问题做了很大的改进, 最重要的就是强制结构的完整性和标签的自定义性. 正因为 XML^[1] 比 HTML 具有更多的优点, 人们普遍认为: XML 最终会取代 HTML 而成为 Web 的通用语言. 此外, 针对 XML 的研究以及支持 XML 的工具也不断涌现. 但是, 毕竟 HTML 的存在已有多年的历史, Web 上的多数信息也是 HTML 格式的, XML 取代 HTML 尚需时日, 研究如何将 HTML 格式转变为 XML 格式具有非常的现实应用价值.

2 HTML 到 XML 的转换方法

2.1 HTML 和 XML 的多叉树表示

对于 HTML 来说, 它的内容是有顺序关系的, 在转换过程中, 必须保持这种有序性, 同时, 还要保证转换前后信息的完整性, 在我们的方法中, 这些条件都可以满足. 从语法上看, HTML 和 XML 并没有本质的差别, 只不过 HTML 采用的标签是预先定义的, 以及 HTML 对标签的匹配没有任何限制, 如果不考虑这些差别, XML 只会比 HTML 多出第一行的

`<?xml version='1.0'?>`, 而其他的内容都是一样的, 如图1所示的一段简单 HTML 文本及相应的 XML 文本. 这样对于一个格式良好 (Well-Formed) 的 HTML 文件来说, 只需在头部加上 XML 的第一行标记, 同时将相应的文件扩展名 .htm 改为 .xml, 就转换成为 XML 文件了. 但对于非格式良好的 HTML, 由于 XML 对标签的匹配限制很严格, 在 HTML 中经常被省略的结束标签 (如 ``, 常略去 ``), 在 XML 中是不允许的故无法通过这种简单的方法来实现转换.

| | |
|---|---|
| <pre><html> <head> <title>HTML 示例 </title> </head> <body> <H1 align="center"> 中国科大</H1> <H2 align="center"> 合肥</H2> </body> </html></pre> | <pre><?xml version='1.0'?> <html> <head> <title>HTML 示例 </title> </head> <body> <H1 align="center"> 中国科大</H1> H2 align="center"> 合肥</H2> </body> </html></pre> |
|---|---|

(a) 一段简单的 HTML 文本 (b) 相应的 XML 格式表示

图1 HTML 文本与相应的 XML 格式表示

Fig. 1 AHTML document and its corresponding XML document

此外, 无论对 HTML, 还是 XML, 都可以用一棵多叉树来表示. 图1中的 HTML 文本及相应 XML 文本, 都可以用图2所示的多叉树来表示. 如果我们能够构造出这样一棵完

整的 HTML 多叉树(下简称 HTML 树),并且在构造 HTML 树的同时,采用一些策略,将 HTML 中不严格的语法消除,最后再由 HTML 树来产生相应的 XML 文件,我们就可以得到格式良好的 XML 文件了,这样,整个转换工作也就会变得十分容易。

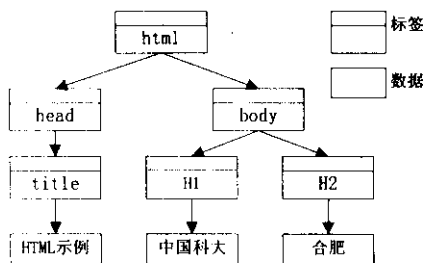


图2 HTML 和 XML 的多叉树表示

Fig. 2 The multi-tree representation of HTML and XML document

2.2 总体流程和数据结构定义

为了便于设计和实现,我们将整个转换过程分为以下几个子过程:

1. 将 HTML 文本以标签为分割点,划分为段的序列,其中和 '<' 和 '>' 之间文本的形成标签段,而 '>' 和 '<' 之间的形成数据段。

2. 先构造出一棵空的多叉树,然后依次处理每个段,并将其插入到相应的节点位置,对于标签段,还处理其中的属性列表。

3. 按树的先根遍历算法,输出到 XML 文件。

为了实现 HTML 的多叉树表示,我们需要定义相应的数据结构。

```
struct SEGLIST    //段列表
{
    int segType;    //标识标签或数据
    char * data;    //标签或数据值
    struct SEGLIST * next;
};
struct ATTRLIST   //属性列表
{
    char * name, * value;
    struct ATTRLIST * next;
};
struct HTMLTREE  // HTML 树(节点)
{
    int nodeType;    //标识标签或数据
    char * data;    //标签或数据值
    struct ATTRLIST * attrSet;
    struct HTMLTREE * lChild, * sibling;
    struct HTMLTREE * parent;
};
```

这里 SEGLIST 用于将整个 HTML 文件转化为段的序列列表,其中 segType 表示该段是标签还是纯数据,对于标签来说,还要区分是开始标签、结束标签还是其他特殊类型的标签(见下)。HTMLTREE 用来存放整棵 HTML 树,从 HTMLTREE 的定义我们可以看出,HTML 树采用树的二叉表示法,lChild 指向第一个儿子,sibling 指向下一个兄弟,parent 指向父节点,nodeType 意义同 SEGLIST 中的 segType,attrSet 则用于存放标签中的属性列表。下面依次介绍每一过程。万方数据

2.3 HTML 分段

HTML 分段过程是依次从 HTML 文件中读入字符,判断其是否为 '<' 或 '>',若为 '<',表示下面输入的是标签开始;若为 '>',则表示标签结束,从上一次出现的 '<' 到本次的 '>' 之间的字符串形成标签段插入到段列表中。而在 '>' 以后,直到出现 '<' 为止,它们中间的字符串就形成数据段插入到段列表中。另外,对于标签段,还要区分一下开始标签和结束标签,这只需要检查 '<' 后面的字符是否为 '/',若不是,则为开始标签(如),否则为结束标签(如),相应的表述如图3所示,其中 stream 为 HTML 输入文件流。

```
while((ch=fgetc(stream))!=EOF)
{
    switch(ch)
    {
        case '<':
            将从上次的 '>' 后到本 '<' 前的数据形成数据段,插入到段表中,同时标示标签段的开始。
            break;
        case '>':
            将从上次的 '<' 后到本 '>' 前的数据形成标签段,插入到段表中,同时标示数据段的开始。
            break;
    }
}
```

图3 将 HTML 划分为段序列

Fig. 3 Partition HTML document into segment sequences

2.4 构造 HTML 树

本子过程是算法的核心,它的处理结果直接影响到 XML 文件的输出。它主要要解决段与段之间的层次和包含关系。为简单起见,我们先假设源 HTML 文件中的标签都是匹配的,且没有其他诸如 script 等特殊的成分。然后,再针对不同的特殊或例外情况作一些修正。先看下面的一段算法。其中 segHead 指向在第一步段列表的表头,每个表节点类型可为:数据段、开始标签和结束标签。

```
struct HTMLTREE * BuildHTMLTree(segHead)
{
    struct HTMLTREE * root=NULL, * CnODE; // 根和当前节点
    p=segHead;
    while(p!=NULL)
    {
        if(p 为开始标签段)
            生成一节点,设置所有的域值,
            若段中含有属性,生成属性列表,
            将该节点作为当前节点最右边的儿子,
            修改当前节点为新节点。
        else if(p 为结束标签段)
            从当前节点顺着 parent 域依次向上找匹配的 begin 标签,
            直到相等,设置该节点为当前节点。
        Else //p 为数据段
            生成一节点,设置所有的域值,
            将该节点作为当前节点最右边的儿子。
        p=p->next;
    }
    return root;
}
```

图4 构造 HTML 树

Fig. 4 Construction of a HTML tree

从上面的算法中我们可以看出,程序依次从段列表中读入一段,判断其类型:

- 若为开始标签段,则需要生成一新节点,设置其中所有的域值(如 noteType),然后将其挂到当前节点(cNode)的下面,作为其最右的儿子;若这时根节点(root)为空,则让 root 指向该新节点;若段中含有属性,attrSet 指向生成的属性列表.最后修改当前节点,让 cNode 指向该新节点,表示现在已进入了下一个层次.由于我们是按段的先后次序来处理段,而段的先后关系又是与源 HTML 一致的,加上在生成新的节点时,总是让它作为其双亲最右的儿子,这样就可以保证在对整棵树进行先根遍历时,能保持原来的顺序关系.

- 若为结束标签段,则不需要生成一新节点,因为这表示当前标签层次的结束,这时仅需要重新设置当前节点,让它指向其父节点或祖先节点,表示现在已返回到更高的一个层次.对于所有标签都完全匹配的 HTML 文本,当前节点会指向它的父节点;而对于那些缺少相应结束标签的标签,如对于<H1><P>Hi</H1>序列,由于<P>缺少结束标签</P>,那么算法在扫描到时,就直接将当前节点指到<H1>这个节点,也就是说,即使中间少了匹配的结束标签,算法也会指到正确的位置.

- 若为数据段,这时仅需要将新节点作为当前节点最右的儿子,不需要修改当前节点,因为对于数据段,它不会进入下一个层次.

同时我们也可以看出,构造 HTML 树的过程,有点类似于栈,只不过是按树的层次来表示进栈和出栈;当树往下长的时候表示进栈,往上退的时候代表出栈.

前面我们曾假设 HTML 文件中的标签都是匹配的,且没有其他诸如 script 等特殊成分.现在我们来分析一些特殊情况,对算法进行一些修正:

- 开始标签没有对应的结束标签,对于一些简单的情况,上面的算法已经可以解决了,如上文提到过的<H1><P>Hi</H1>.对于一些比较复杂的情况,有时还需要向前扫描若干段,同时结合 HTML 的语法规则,来判断应该回到那一个层次,也可以在软件中提供几个选择项,让用户自己来指导软件该如何处理.

- 对于 HTML 中含有的 script 脚本,由于其中的表达式中经常含有'>'、'<'等特殊字符,在扫描时若不对这些字符作特殊处理,算法就不能正确运行,所以,必须记录当前是否已进入脚本区,若是,则不将相应的内容作为标签块处理,另外还可以增加段的类型,表示当前段为脚本段,以方便下一步的输出处理.

- 由于 HTML 中经常将多个空格作为一个来处理,同时对于回车、换行或制表符,也经常作为空格来处理,所以在处理数据段时,可以直接将它们合并或转换为一个空格.

- 对于 HTML 中含有的注释,也可以作为一个注释段类型来处理.

- 对于有些 HTML,在其头部经常含有诸如<!DOCTYPE ...>的段,若果这些信息对于目标文档不是很重要,就直接把它删除,比如,在 KPS 算法^[3]中,计算某一个对象的 HTML-PATH,只需要考虑在 HTML 树中此对象到<body>标签间的路径,所以整个 HTML 的头部都可以被删除.另外这些信息也可作为注释段类型来处理.

- 以上列举并没有囊括所有情况,其他的还需要根据实际情况进行特殊处理.

2.5 输出 XML

这个子过程也比较简单,按树的先根遍历算法:先输出根,依次递归调用所有子树,输出到 XML 文件即可.输出过程如图5所示,其中 OutputHTMLTree 输出 HTML 树内容,OutputHTMLTree 先输出 XML 标记行,然后调用 OutputHTMLTree 输出 HTML 内容,stream 为 XML 输出文件流,hTree 为 HTML 树根节点.

```
void OutputHTML Tree(stream, hTree)
{
    if(hTree!=NULL)//非空树
    {
        if(hTree 为标签节点)
            输出标签及属性列表://如<B>
        else
            输出数据;
        对所有 hTree 的子树递归调用 OutputHTMLTree;
        if(hTree 为标签节点)
            输出结束标签://如</B>
    }
}

void OutputHTML TreeAsXML(stream, hTree)
{
    fprintf(stream,"<?xml version='1.0' ?>\n");
    OutputHTMLTree(stream,HTMLTree);
}
```

图5 将 HTML 树输出为 XML

Fig. 5 Output a HTML tree into XML document

OutputHTMLTreeAsXML 过程生成的 XML 文件可以通过 XML 工具(如 IE)来检查其是否具有结构良好性,若不是,则可能需要修改第2步,甚至在第1步中的过程,加入一些需要特殊处理的代码,以产生结构良好的 XML 文件.

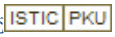
3 结束语

采用基于多叉树的 HTML 到 XML 的转换方法,不仅可以用来有效地解决 HTML 的结构化问题,配合其他的 Web 信息检索方法,如 KPS 算法,还可以用来解决诸如 Web 信息检索、HTML 查询等问题.另外,将 HTML 转化为 XML 后,还可以利用 XML 的 DOM 和 SAX 编程接口,以方便对 HTML 的信息抽取.当然,由于 HTML 的格式灵活多变,本方法尚不能完全解决 HTML 转化过程中所有可能的问题,更细致的工作还有待于进一步深化和研究.

References:

- 1 Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation 6 October 2000[EB/OL]. <http://www.w3.org/TR/REC-xml>.
- 2 Raggett D, Le Hors A and Jacobs I. Hypertext markup language 4.0 Reference Specification[EB/OL]. December 1997. <http://www.w3.org/TR/REC-html40>.
- 3 Guan T, Wong K F. KPS: a Web information mining algorithm [J]. Computer Networks, Elsevier, 1999, 31:1495~1507

一种基于多叉树的HTML到XML的转换方法

作者: 张文斌, 陈恩红, 王进
作者单位: 中国科学技术大学, 计算机科学系, 安徽, 合肥, 230027
刊名: 小型微型计算机系统 
英文刊名: MINI-MICRO SYSTEMS
年, 卷(期): 2003, 24(4)
被引用次数: 6次

参考文献(3条)

1. Extensible Markup Language (XML) 10 (Second Edition) W3C Recommendation 6 October 2000
2. RAGGETT D; Le Hors A; Jacobs I Hypertext markup language 40 Reference Specification 1997
3. Guan T; Wong K F KPS: a Web information mining algorithm[外文期刊] 1999(11-16)

本文读者也读过(10条)

1. 黄晓斌 HTML向XML转换的研究[期刊论文]-现代图书情报技术2003(1)
2. 李昕, 李丽萍, 常革新, LI Xin, LI Li-ping, CHAN Gge-xin 基于XML的文档的动态产生[期刊论文]-辽宁工程技术大学学报2006, 25(1)
3. 张靓 基于XML的WEB数据抽取与存储的研究[学位论文]2005
4. 方睿, 韩斌, 陈灵 WordML文档转换器研究[期刊论文]-计算机应用2006, 26(z1)
5. 王富强, 王默玉, WANG Fu-qiang, WANG Mo-yu 数据岛在HTML的嵌入显示[期刊论文]-电脑与信息技术2005, 13(1)
6. 李青山, 陈平, Li Qingshan, Chen Ping 一种基于内容的HTML到XML转换策略[期刊论文]-计算机工程与应用2001, 37(9)
7. 黄伟, 刘娟, HUANG Wei, LIU Juan 一种基于DOM树的HTML转换为XML的方法[期刊论文]-电脑知识与技术(学术交流)2006(7)
8. 李雪竹, LI Xue-zhu 一种基于XML的Web数据抽取的实现[期刊论文]-科学技术与工程2008, 8(9)
9. 温津伟, 罗四维, 王宝静 多类模式识别的动态多叉树算法研究与实现[期刊论文]-计算机研究与发展2003, 40(1)
10. 连瑞梅, Lian Ruimei 基于XML的数据格式转换的研究[期刊论文]-科技广场2007(11)

引证文献(6条)

1. 袁玲, 邓晓燕 一种基于多叉树的检索方法及其应用[期刊论文]-计算技术与自动化 2011(3)
2. 谭新良, 蔡代纯 基于XML文档检索的搜索引擎设计[期刊论文]-计算机科学 2007(3)
3. 彭涛, 曾燕, 代晓红, 胡飞 基于语义分层迭代法的网页挖掘技术[期刊论文]-重庆工商大学学报(自然科学版)2007(5)
4. 胡飞 基于标记树的Web页面区域划分和搜索方法[期刊论文]-计算机科学 2005(8)
5. 刘依依 基于转码及信息提取的WAP代理服务器的研究与设计[学位论文]硕士 2005
6. 张彦 面向XML的搜索引擎研究[学位论文]硕士 2005

本文链接: http://d.wanfangdata.com.cn/Periodical_xwxjsjxt200304023.aspx